Faculty of Science and Engineering
Department of Computing

OPTIMAL REASONING OF GOALS IN AGENT-ORIENTED SYSTEMS

Chitra Muniyappa Gounder Subramanian

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

July 2017

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made. This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Chitra Muniyappa Gounder Subramanian
Faculty of Science and Engineering
Department of Computing
Curtin University
July, 2017

Acknowledgements

The completion of my thesis has been a long, but fruitful, journey. I am grateful to many individuals for their help and guidance over the past four years.

First of all, I would like to express my deepest thank to my supervisor, Associate Professor Aneesh Krishna, and co-supervisors Associate Professor Arshinder Kaur, and Dr. Raj Gopalan, for providing me with the wonderful opportunity to complete my thesis. I especially thank Dr. Aneesh Krishna, for his unwavering faith in my ability, and his motivation and encouragement through the most difficult times. I have been very fortunate and honoured to work with him and have his guidance. I am also very grateful to Dr. Arshinder Kaur for her invaluable comments and suggestions throughout this research and the writing of the thesis.

I am grateful to Associate Professor Mihai Lazarescu, Head of the Department and Esther Yew for their administrative support for my conference trips.

My gratitude and acknowledgement to my family. I thank my husband, Vadivelu Rangasamy, who has always encouraged my pursuit of knowledge. He continued to give his support and encouragement during this journey. Special thanks to my children Mukesh Kumar and Haritha, for their love, patience and understanding during the course of this research work. I would like to thank my parents who have given me tremendous encouragement and support during these years.

A lot of thanks to my friends Sunitha, Sreenithya, Dang and Illaiyaraja for the wonderful time we shared. There are many who have helped me in one or the other way during my study, and I would like to thank them all.

Credits

Portions of the material in this thesis have previously appeared in the following publications:

1. Chitra M Subramanian, Aneesh Krishna, Raj P. Gopalan and Arshinder Kaur (2015). Quantitative Reasoning of Goal Satisfaction in the i^* framework, The 27^{th} International Conference on Software Engineering and Knowledge Engineering (SEKE 2015), Pittsburgh, USA.

2. Chitra M Subramanian, Aneesh Krishna and Arshinder Kaur (2015). Reasoning about Goal Satisfaction for early Requirements Engineering in the i^* framework using Inter-actor Dependency, The 19^{th} Pacific Asia Conference on Information Systems (PACIS 2015), Singapore.

3. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2015). Optimal Reasoning of Goals in the i^* Framework. In Software Engineering Conference (APSEC), 2015 Asia-Pacific (pp. 346-353). IEEE.

4. Chitra M. Subramanian, Aneesh Krishna, Arshinder Kaur (2016) Sensitivity Analysis of the i^* Optimisation Model. JSW 11(1): 10-26.

5. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2016). Optimal Goal Programming of Softgoals in Goal- Oriented Requirements Engineering. The 20th Pacific Asia Conference on Information Systems (PACIS), Taiwan June 27- July 1.

6. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2017). Game Theory based Requirements Analysis in the i^* framework. The Computer Journal. (Accepted on 25^{th} October 2017)

Abstract

Requirements Engineering (RE) is one of the crucial and demanding phase of software engineering. The success or failure of a software project or a system depends on RE activities. RE states the intention of a software system, the functionalities it must accomplish and the definition of the software constraints that must be designed and implemented. The activities involved in RE are elicitation, analysis, specification, validation, and management of requirements. Many approaches ranging from traditional to goal-oriented RE have been proposed for modeling and analysing requirements.

Goal-Oriented Requirements Engineering (GORE) considers the system-to-be and its environment as a group of active components called agents. These components are the stakeholder's, devices and softwares involved in the system. GORE is appropriate for analysing requirements (also known as goals) particularly with regard to non-functional requirements (also known as softgoals) in the early stages of the software development cycle. There are number of benefits associated with the analysis of goal models apart from modeling. One such benefit is the evaluation of alternative design options based on non-functional requirements. The critical problem in goal analysis is: given a set of alternative options for a goal, which alternative gives best satisfaction in terms of the softgoals of the system. Although many approaches have been proposed, goal reasoning is still a significant challenge in RE.

Among the various GORE framework, the i^* framework which is Goal- and agent-oriented modeling framework is appropriate for representing the social elements of the system domain and is suitable for reasoning of goals particularly at the requirements level. The survey of goal analysis procedures for the i^* framework shows that these are basically qualitative in nature. We have developed a novel approach based on fuzzy numbers for quantitative reasoning of goals in the i^* framework for early requirements engineering. The fuzzy-based quantitative approach was enhanced by using an operation research technique namely

optimisation. Optimisation was used to deal with incomplete or unobtainable information about requirements. Furthermore, we have proposed a novel approach based on game theory to manage requirements of a conflicting nature. A tool has been developed to implement the proposed approaches and it was evaluated using OpenOME, an existing tool for the i^* framework. The evaluation results have shown that fuzzy-based optimal goal analysis is better when choosing from alternative options of goals in comparison with respect to the OpenOME tool.

Contents

Acknowledgements **iii**

Abstract **vii**

List of Figures **xiii**

List of Tables **xv**

1 Introduction **1**

1.1 Research Questions 3

1.2 Research Methodology 4

1.3 Research Outcomes and Main Contributions 6

1.4 Thesis Structure 7

2 Literature Survey **9**

2.1 Introduction to Requirements Engineering 10

2.2 Goal-Oriented Requirements Engineering 11

2.2.1 KAOS Framework 12

2.2.2 Non-Functional Requirements Framework 12

2.2.3 Goal-Oriented Requirements Language 13

2.2.4 Attribute Goal-Oriented Requirements Analysis Method . 14

2.2.5 i^* Framework 14

2.2.6 TROPOS 15

2.3 Goal Analysis 16

2.3.1 Qualitative Analysis 17

2.3.2 Quantitative Analysis 19

2.3.3 Optimisation in Goal Analysis 24
2.3.4 Game Theory in Requirements Engineering 25
2.4 Summary of the Research Review and Purpose of this Research . 26
2.5 Chapter Summary 28

3 Inter-actor Quantitative Reasoning of the *i Framework 33**
3.1 Overview of Fuzzy Numbers 34
3.2 Overview of the *i** Framework 36
3.2.1 The Strategic Dependency Model 37
3.2.2 The Strategic Rationale Model 38
3.3 The Quantitative, Fuzzy based Goal Analysis using Inter-Actor Dependency 40
3.4 Simulation and Evaluation of the Approach with Case Studies . . 45
3.5 Chapter Summary 53

4 Optimal Reasoning and Sensitivity Analysis in the *i Framework 55**
4.1 Need for Optimisation in Quantitative Analysis of the *i** Framework 57
4.2 Multi-Objective Optimisation 61
4.3 Optimal *i** Framework 62
4.3.1 Multi-Objective Optimisation Algorithms 68
4.3.2 Encoding the Optimisation Problem for Weights of the Leaf Softgoals 68
4.4 Application of Optimisation to a Case Study 70
4.4.1 Evaluation of the Approach using a Case Study 72
4.5 Sensitivity Analysis 77
4.5.1 Sensitivity Analysis of the *i** Framework 80
4.5.2 Analysis with LAS Case Study 82
4.6 Chapter Summary 84

5 Optimal Goal Programming of Softgoals in the *i Framework 85**
5.1 Optimisation Model based on SoftGoals in the *i** Goal Model . . 87
5.2 Multi-Objective Goal Programming (MOGP) 91
5.3 Formulation of the *i** Framework as a Multi-Objective Goal Model 92

5.4 Evaluation of Alternatives with Softgoal Optimisation 95
5.4.1 Deriving Objective Functions for the Actors 96
5.4.2 Obtaining Weights by MOGP 98
5.4.3 Evaluation of the Approach 100
5.5 Chapter Summary 103

6 Softgoals with Opposing Nature 105
6.1 Motivating Example: Supply Chain Management System 106
6.2 Challenges and Motivation 109
6.3 Game Theory Model for Goal Modeling 111
6.3.1 Introduction to Game Theory 111
6.3.2 Game Theory for Multi-Objective Optimisation of i^* Goal Model 113
6.4 Illustration of Game-Theory-Based Goal Analysis using i^* Framework 121
6.4.1 Generation of the Objective Functions using Gauge Variables 122
6.4.2 Pay-off Matrix and Alternative Selection 124
6.4.3 Evaluation of the Approach 128
6.4.4 Actor with $n(> 2)$ Top Softgoals 135
6.5 Chapter Summary 135

7 Tool Validation 137
7.1 State of the Art 138
7.1.1 REDEPEND-REACT 138
7.1.2 J-Prim 138
7.1.3 DesCARTES 139
7.1.4 OpenOME 139
7.1.5 Optimal Goal Analysis of the i^* framework 139
7.2 OpenOME: Qualitative Analysis of the i^* Framework 140
7.3 Optimal Goal Analysis Tool: Quantitative Analysis of the i^* Framework 143

7.4 Empirical Evaluation (Controlled Experiment) to Evaluate the Usability of the Optimal i^* Tool 144
7.4.1 Research Objectives 145
7.4.2 Experimental Design 147
7.4.3 Execution 150
7.4.4 Results and Analyses 150
7.5 Discussion 156
7.5.1 Threats to Validity 156
7.6 Chapter Summary 157

8 Contributions, Limitations and Future work 159
8.1 Summary of contributions 160
8.2 Limitations and Future work 162
8.2.1 Additional parameters for the i^* evaluation 162
8.2.2 Implementation Issues 162

Bibliography 165

List of Figures

3.1 Membership function of TFN 35
3.2 Example of Strategic Dependency model 37
3.3 A Simple SR Model: Youth Counselling Example (adapted from Horkoff and Yu (2009)) 39
3.4 Inter-actor Dependency 40
3.5 SR Model: Youth Counseling Example (adapted from Horkoff and Yu (2009)) 41
3.6 Membership function for Impact 43
3.7 Quantitative Analysis of Goals for alternative *Use CyberCafe/Portal/ChatRoom* in Kids Youth Counseling case study 47
3.8 Meeting Scheduling System case study 49
3.9 Comparison Graph for Kids Youth Counseling case study 51
3.10 Comparison Graph for Meeting Scheduling System case study . . 52

4.1 SR diagram for LAS (adapted from You (2004)) 58
4.2 Optimisation Framework 64
4.3 Crossover for Optimal weights 70
4.4 Optimal softgoal scores for the alternative option *ByDatabase or Network* 73
4.5 An SR Model of Telemedicine System (adapted from (Yu, 2001)) 74
4.6 A Goal Analysis with optimal weights for alternative option *Provider Centred Care* 76
4.7 Graphical representation of scores comparisons with and without optimisation 78
4.8 A model for the sensitivity analysis of optimisation 80
4.9 Perturbation of fuzzy number 81
4.10 Sensitivity Analysis of the softgoal *optimal[mobInst]* 83

4.11 Sensitivity Analysis of softgoal Timeliness[mobilisation] 83

5.1 Directed Graph representation of a SR diagram 87
5.2 Scheme of the Multi-Objective Goal Programming Softgoal Optimisation and alternative selection 95
5.3 SR Model for Youth Counseling (adapted from Horkoff and Yu (2009)) 96
5.4 An Strategic Rationale Model: Telemedicine system (Adapted from Yu (2001)) 99
5.5 Telemedicine Comparison of Softgoals optimisation scores with Leaf softgoals Optimisation scores 103
5.6 Youth Counseling Comparison of Softgoals optimisation scores with Leaf softgoals Optimisation scores 104

6.1 Supply Chain Network 106
6.2 SD model for Supply Chain Network 108
6.3 Simplified SD model for Supply Chain Network 108
6.4 Simplified SR model for Supply Chain Network 110
6.5 Tool Result for actor Manufacturer for alternative VMI 128
6.6 Tool Result for actor Manufacturer for alternative Traditional . . 129
6.7 Graphical representation of proportional values of alternatives for SCM 130
6.8 Simplified SR model for Telemedicine (adapted from Yu (2001)) . 131

7.1 SR Model: Telemedicine Example (adapted from (Yu, 2002)). . . 141
7.2 OpenOME: Goal Analysis with first alternative 142
7.3 OpenOME: Goal Analysis with second alternative 143
7.4 Optimal Tool: Goal Analysis (First Alternative) 145
7.5 Optimal Tool: Goal Analysis (Second Alternative) 146
7.6 Boxplot of the actual time consumption 151

List of Tables

2.1 Analysis Approaches in Software Engineering,- Strengths and Limitations . . . 29
2.2 Goal Analysis Approaches,- Strengths and Limitations . . . 30
2.3 Comparison of Goal Analysis Approaches(cont..) . . . 31
2.3 Comparison of Goal Analysis Approaches . . . 32

3.1 Fuzzy values for hard goal and softgoal correlation. . . . 42
3.2 Leaf softgoal scores for all three actors of Kids Youth Counseling case study . . . 48
3.3 Top Softgoals Satisfaction scores for Kids Youth Counseling (* indicates goal selection) . . . 48
3.4 Top Softgoals Satisfaction scores for the actor MeetingParticipants (* indicates goal selection) . . . 50
3.5 Top Softgoals Satisfaction scores for the actor MeetingInitiator (* indicates goal selection) . . . 50
3.6 Top Softgoals Satisfaction scores for the actor MeetingScheduler (* indicates goal selection) . . . 51
3.7 Scores Comparison for Kids Youth Counseling (with and without inter-actor dependencies) . . . 53
3.8 Scores Comparison for Meeting Scheduling System (with and without inter-actor dependencies) . . . 53

4.1 Weights of the leaf softgoals of the actor Resource Allocator . . . 72
4.2 Top softgoal scores for all the alternative options in LAS case study 73
4.3 Optimised weights of leaf softgoals of the Telemedicine goal model 75
4.4 Top SoftGoals Satisfaction Scores for Telemedicine Goal model (* indicates goal selection). . . . 76

4.5 Top softgoal scores of Patient (for different weights) 77
4.6 Top softgoal scores of Healthcare Provider (for different weights) . 77
4.7 Scores Comparison for Telemedicine System (with and without Optimisation) 77

5.1 Optimal weights for the Kids Youth Counseling case study 100
5.2 Optimal weights for the Telemedicine case study 100
5.3 Scores of Top softgoals of the Kids Youth Counseling case study (* indicates selection) 101
5.4 Scores of Top softgoals of the Telemedicine case study (* indicates selection) 101
5.5 Scores Comparison for the Youth Counseling case study 103
5.6 Scores Comparison for the Telemedicine case study 103

6.1 Objective functions values for SCM goal model 123
6.2 Satisfaction Percentage of Top Softgoals for the actor Manufacturer(SCM goal model)(* indicates options selected) 127
6.3 Satisfaction Percentage of Top Softgoals for the actor Retailer(SCM goal model)(* indicates options selected) 127
6.4 Objective functions values for Telemedicine goal model 131
6.5 Satisfaction Percentage of Top Softgoals for the actor Patient (Telemedicine goal model) (* indicates options selected) 132
6.6 Satisfaction Percentage of Top Softgoals for the actor Healthcare Provider(Telemedicine goal model)(* indicates options selected) . 133
6.7 Weights of SCM and Telemedicine goal models 134
6.8 Top Softgoals scores of SCM goal model (* indicates options selected) 134
6.9 Top Softgoals scores of Telemedicine goal model(* indicates options selected) 134

7.1 Key Elements of Controlled Experiment 147
7.2 Average time consumption for OpenOME tool and Optimal tool . 151
7.3 T-test values for null hypothesis H_T0 151
7.4 Chi-Square test values for null hypothesis H_I0 152
7.5 Chi-Square test values for null hypothesis H_A0 153
7.6 Chi-Square test values for null hypothesis H_F0 153
7.7 Chi-Square test values for null hypothesis H_P0 154

7.8 Chi-Square test values for null hypothesis H_S0 155
7.9 Chi-Square test values for null hypothesis H_U0 155

List of Abbreviations

RE	**R**equirements **E**ngineering
GORE	**G**oal-**O**riented **R**equirements **E**ngineering
NFR	**N**on-**F**unctional **R**equirements
KAOS	**K**nowledge **A**cquisition in **A**utomated **S**pace
GRL	**G**oal-**O**riented **R**equirement **L**anguage
AGORA	**A**ttributed **G**oal-**O**riented **R**equirements **A**nalysis
MOGP	**M**ulti-**O**bjective **G**oal **P**rogramming
SD	**S**trategic **D**ependency
SR	**S**trategic **R**ationale
AHP	**A**nalytical **H**ierarchy **P**rocess
PDDL	**P**lanning **D**omain **D**efinition **L**anguage
HTN	**H**ierarchical **T**ask **N**etwork
UML	**U**nified **M**odeling **L**anguage
OCL	**O**bject **C**onstraint **L**anguage
MCDA	**M**ulti **C**riteria **D**ecision **A**nalysis
TOPSIS	**T**echnique for **O**rder of **P**reference by **S**imilarity to **I**deal **S**olution
PRFGORE	**P**rioritization of **R**equirements using a **F**uzzy-based approach in **G**oal-**O**riented **R**equirements **E**licitation
ENFR	**E**xtended **N**on-**F**unctional **R**equirements
TFN	**T**riangular **F**uzzy **N**umber
LSG	**L**eaf **S**oft**G**oal
SG	**S**oft**G**oal
LAS	**L**ondon **A**mbulance **S**ystem
GA	**G**enetic **A**lgorithm
NSGA	**N**on-dominant **S**orting **G**enetic **A**lgorithm
SPEA	**S**trength **P**areto **E**volutionary **A**lgorithm
LP	**L**inear **P**rogramming

MOLP	**M**ulti-**O**bjective **L**linear **P**rogramming
SCM	**S**upply **C**hain **M**anagement
VMI	**V**endor **M**anaged **I**nventory
GT	**G**ame **T**heory
MOP	**M**ulti-**O**bjective o**P**timisation
TS	**T**op **S**oftgoal
DesCARTES	**D**esign **C**ase tool for **A**gent-oriented **R**epositories **T**echniques, **E**nvironment and **S**ystems
AUML	**A**gent textbfUnified **M**odeling **L**anguage
OME	**O**rganisation **M**odelling **L**anguage
RQ	**R**esearch **Q**uestion

Chapter 1

Introduction

With the advancements in Internet and web technology, most software systems progress towards open, distributed, decentralized and integrated environments. Users can now access various services (e-commerce, e-learning, e-banking, e-healthcare, e-forecasting, etc) from anywhere, at any time and in any form. There is a demand for new features in the software systems enabling them to adapt to environmental changes and high expectations of the users. This is attainable with the emergence of new technology namely, agent-oriented technology. Software agents are software systems that are autonomous and social; they communicate, coordinate, and cooperate with each other to accomplish goals. Agent-oriented software engineering has received a great deal of attention in agent research (Yu, 2001).

When developing any software system, one has to go through the entire software development cycle:- requirements, design, construction, validation and verification, deployment, maintenance, evolution, legacy, re-use, etc. For agent-based software technology, an important but challenging task is requirements engineering. Requirements engineering emphasizes "*the use of systematic techniques to ensure the completeness, consistency, and relevance of the system requirements*" (Chung et al., 2012). In requirements engineering, all stakeholders, users, developers, and analysts comprehend each other's interests. Furthermore, the requirements analyst examines alternative options and makes decisions about the systems to be implemented. The requirements of a software system are classified as either functional requirements (goals) or non-functional requirements (softgoals). Functional(or behavioural) requirements are associated with the function of the system or its components. Non-functional requirements are the criteria for checking the system's operation rather than specific behaviour. Non-functional

requirements like usability, integrity and security have more impact on software systems than do the functional requirements (Pohl, 2010).

In RE literature, various methods have been proposed to model requirements, namely conceptual entity-relationship modeling, structured modeling, object-oriented, use case and goal-oriented approaches. Goal-oriented requirements engineering (GORE) is complementary when compared to traditional approaches (Mylopoulos et al., 1999). Compared with other approaches, GORE is appropriate for requirements analysis with regard to non-functional requirements (NFR) early in the software development cycle and is appropriate for the evaluation of alternatives. The GORE approach is used to reveal, examine and express stakeholder's goals that lead to software and system requirements. The other approaches are more suited to requirements analysis in the later stage of the software cycle and target the traceability between requirements and implementation. The popular GORE frameworks are Non-Functional Requirements (NFR) framework (Chung et al., 2012), Knowledge Acquisition in Automated Space (KAOS) (Dardenne et al., 1991), i^* framework (Yu, 2011), Tropos (Bresciani et al., 2004), Goal-Oriented Requirement Language (GRL) (Amyot et al., 2010) and Attributed Goal-Oriented Requirements Analysis (AGORA) (Kaiya et al., 2002).

In addition to modeling analysts use goal models to determine whether goals are being met, to evaluate design alternatives, to choose the system design, analyse risk and prioritize the requirements. A goal may have different design options and, for implementation, the best design has to be selected. Hence, when evaluating design alternatives, analysts explore various design alternatives for a goal and select the best ones using several evaluation criteria. Softgoals in goal models are used as evaluation criteria in existing quantitative and qualitative approaches(Mylopoulos et al., 1992). During the process of goal model evaluation, the qualitative or quantitative values are propagated either using forward propagation (from the bottom softgoals to the top softgoals) or backward propagation (from the top softgoals to the bottom softgoals). The satisfaction levels of softgoals are assessed based on the selected design alternative. The design alternative that gives best satisfaction in terms of the top softgoals is selected.

Qualitative approaches use qualitative labels such as *denied*, *partially denied*, *satisfied*, *partially satisfied*, *unknown*, and *conflict* in the propagation algorithm. The drawback of qualitative approaches is the ambiguity in decision making. Ambiguity arises when two alternatives have the same label and when a goal receives an *unknown* or *conflict* label. It is essential to deal with conflicting requirements as these are one of the three main problems associated with software

development (Mairiza et al., 2014).

Quantitative approaches use numbers instead of qualitative labels to overcome the problems of qualitative analysis. It is also crucial to assign definite numbers to stakeholders' requirements because requirements analysis may involve distinct stakeholders having diverse preferences regarding the same requirements. This is explained by the fact that different stakeholders have different levels of knowledge, training and skills (Wang and Xiong, 2011). Moreover, in reality, linguistic terms such as *low cost*, and *high profit* are generally used by the stakeholders to communicate their requirement preferences. It is challenging task to represent these terms with definite numbers. Hence, there is a need for a novel approach to goal analysis.

Compared with KAOS and NFR frameworks, frameworks such as i^*, GRL and Tropos goal model shows different actors and their dependencies. Each actor depends on other actors for goal accomplishment. These interdependencies are also influential in the decision-making regarding alternative design options (Bresciani et al., 2004; Amyot et al., 2010; Yu, 2011). The existing literature has also discussed the actor dependencies and formalisation (Morandini et al., 2007).

In practice, due to incomplete or unobtainable information, the input data used to evaluate the requirements are imprecise. The inputs used in the analysis are subjected to the requirements analyst. Moreover, in real-life RE problems, one has to deal with multiple-goal models in which there can be more than one goal which is important to address RE problems. The priority of these goals may be different but it is important that they be considered simultaneously. These goals might be either conflicting or congruent. Furthermore, another problem with goal models is scalability. As the size of the goal model increases, it is difficult to decide and assign values to goals and, hence, the decision making is a challenging task (Yu, 2011; Letier and Van Lamsweerde, 2004; Heaven and Letier, 2011). Therefore, there is a need for automating the goal analysis procedure.

1.1 Research Questions

The following research questions were raised:

1. How to represent the linguistic description of requirements (by stakeholders) in goal analysis?

2. How to effectively represent the subjective preference of quantitative values used in the goal analysis?

3. How to effectively implement requirements of opposing nature?

4. What type of tool can assist in the process of goal analysis?

The research is carried out using the i^* framework. The reason for selecting the i^* framework is that it is a goal- and agent-oriented modeling framework that can be efficiently used to model and analyse the relationships among the strategic entities in a social network (Yu, 2011), which includes human organizations and other types of social structures. The i^* framework has also been used in business process modeling and redesign (Yu and Mylopoulos, 1995) and for software process modelling (Yu and Mylopoulos, 1994). Moreover, in the existing RE literature, there is no work on the quantitative analysis of goals in the i^* framework. Hence, in this research, the i^* framework is preferred as a modeling tool for the goal analysis. The research includes: fuzzy-based procedures for evaluating goals; the evaluation of goals using inter-actor dependencies in the i^* framework; procedures for optimising the i^* goal model to avoid the subjective preferences during goal analysis; game-theory-based procedure for handling the optimisation of softgoals with opposing objective functions and finally, the development of a tool for the analysis of goals in the i^* framework.

Research Objectives The main objectives of this research are:

- Use of fuzzy numbers to represent linguistic description of requirements
- Use of optimisation technique to avoid subjective preferences in goal analysis
- Use of game-theory-based procedure to effectively implement requirements of opposing nature
- Development of a tool to assist in the process of goal analysis

1.2 Research Methodology

To address the limitations of the existing qualitative and quantitative approaches, we explored the use of fuzzy numbers for goal analysis with the aim of supporting decision making during the RE process. The linguistic representation of stakeholders' requirement preferences can be easily expressed in Fuzzy Logic (Zadeh, 1965, 1975).

- **Fuzzy numbers to represent linguistic terms of requirements:** First, we have developed a fuzzy-based quantitative approach for finding softgoals satisfaction using the inter-actor dependencies in the i^* framework. The approach used fuzzy concepts to capture requirements, which can be stated in linguistic terms. Fuzzy logic helps in converting the linguistic terms to quantitative numbers.

- **Optimisation for incomplete or unobtainable information about requirements:** Next, in order to handle incomplete or unobtainable information about the requirements and scalability issue of a goal model, we have developed an approach using optimisation, an operation research technique. In the proposed fuzzy-based, inter-actor goal analysis procedure, the weights of the leaf softgoals are assigned by the analyst and this is subject to the analyst's preference. To avoid this subjective preference and to minimize the analyst's interaction, we have proposed an optimisation-based approach. In the optimisation approach, based on the leaf softgoals of a goal model, the weight of the leaf softgoals are found and, in turn these are used in the analysis of the goal model. The optimisation model has been expanded to include sensitivity analysis, in order to produce useful information as the input data for the requirement analyst.

  The optimisation approach was further enhanced by developing an optimisation model that is based on all the softgoals and leaf softgoals of a goal model, presenting a complete optimisation model for an i^* framework in a more generalized form. The reason for this improvement is that the choices made from a range of alternatives are based upon the propagation of values throughout the entire hierarchy of softgoals. Therefore, an optimal model was developed by taking into consideration all of the softgoals within the hierarchy.

- **Requirements of conflicting nature:** Incredibly, most of the real-world business problems encounter the simultaneous optimisation of many competitive objective functions (Zou et al., 2011). In existing requirements engineering literatures (Heaven and Letier, 2011), goal analysis procedures have taken into consideration objectives of maximising type (all objectives of the same type). A game-theory-based approach was proposed in order to select an alternative design for a system with goals of opposing objective functions in an i^* goal model. The reason for the selection of the game

theory concept is that it is used to find optimal solutions when there is conflict, with the assumption that players are rational and behave in their own interests (Kelly, 2003). The non-functional requirements with opposing objective functions of the system are treated as players in a game and the alternative design options of the system are considered as strategies of the game.

1.3 Research Outcomes and Main Contributions

This thesis makes several contributions to the current state-of-art RE literature.

- **Fuzzy based quantitative analysis of the i^* framework:** Firstly, we developed a fuzzy based quantitative method for goal analysis of an i^* framework using the inter-actor dependencies among the actors. Fuzzy values are defined in terms of goal/softgoal contribution (or impact) to softgoal relations. The alternative options are selected based on the percentage of their contribution of satisfaction to the top softgoals. A simulation-based implementation was developed in VC++ to evaluate the feasibility of the approach.

- **Optimisation of the i^* framework based on leaf softgoals:** Secondly, we developed an optimisation approach for the i^* goal analysis. In the fuzzy-based quantitative goal analysis, the weights of the leaf softgoals are assigned by the analyst. Different analysts have different preferences regarding leaf softgoals and, hence, different satisfaction values are obtained for the top softgoals for a selected alternative. To avoid such subjective preferences, a multi-objective optimisation method is used to find the weights of the leaf softgoals, which are then used in the goal analysis. The feasibility of the approach was tested by simulation developed in VC++ integrated with MATLAB. The built-in genetic algorithm of MATLAB was used for optimisation of the approach.

- **Optimisation of the i^* framework based on all softgoals in a Strategic Rationale model:** Thirdly, the optimisation approach was enhanced to include all the softgoals in the hierarchy. The reason for this is that an alternative selection is made by propagating the score value through all the

softgoals in its propagation path. A complete optimisation model for the i^* framework was developed using the Multi-Objective Goal Programming (MOGP) approach. To test the feasibility of the approach, a simulation was developed in Java Eclipse integrated with the IBM Cplex optimisation tool. The choice of Java Eclipse instead of VC++ (as in the previous two approaches) is a step towards tool development for optimal goal analysis of the i^* framework.

- **Analysis of goals with opposing objective functions:** Fourthly, our research has addressed the problems of goals with opposing objective functions (one with maximum value and the other with minimum value). The game-theory-based approach was used to handle the goals with opposing objective functions. A tool has been developed in Java integrated with the IBM Cplex optimisation tool for the game-theory-based goal analysis in the i^* framework.

- **Evaluation of the Optimal goal analysis tool:** Finally, our last contribution is the empirical evaluation of the Optimal tool to assess the efficiency and effectiveness of our approaches. This evaluation is done for the purpose of comparison with the OpenOME tool, an open tool for the i^* goal analysis.

1.4 Thesis Structure

The remainder of this thesis is structured as follows:-

- Chapter 2: "*Literature Survey*" reviews the existing research work related to this thesis. This includes an overview of Goal-Oriented Requirements Engineering (GORE) frameworks. This chapter also provides background information which details the approaches available for goal analysis namely qualitative and quantitative approaches and also those that use optimisation for goal analysis.

- Chapter 3:"*Fuzzy-based Inter-actor Quantitative Reasoning*" provides a description of our fuzzy-based quantitative approach to address the ambiguity and conflict problems that arise in the qualitative approaches for the i^* framework.

- Chapter 4:"*Optimal Reasoning and Sensitivity Analysis*" describes how optimisation and sensitivity analysis are applied to the i^* framework.
- Chapter 5:"*Optmal Goal Programming of Softgoals*" provides a description of our complete optimal framework for i^* framework.
- Chapter 6:"*Softgoals with Opposing Objective Functions*" presents in detail how the game-theory-based approach was used for handling the softgoals of opposing objective functions.
- Chapter 7:"*Evaluation*" presents, discusses and analyses the results of an empirical evaluation of our Optimal goal analysis tool for the purpose of comparison with the existing OpenOME tool.
- Chapter 8:"*Conclusion*" revisits our research problems, reiterates our contributions and discusses the limitations of our approach. The chapter concludes with suggestions for future work.

Chapter 2

Literature Survey

This chapter gives some insight into the problem that this research addresses. The first part of this chapter provides some background on Requirements Engineering (RE) and Goal-Oriented Requirements Engineering (GORE). In Section 2.3, we explain the different approaches proposed for performing goal analysis. In Section 2.4, we present the purpose of this research work.

2.1 Introduction to Requirements Engineering

Requirements refer to a set of specifications or a description of what a system has to do. They define for the software developer what needs to be developed. There are different kinds of requirements: those that specify a feature of a system; those that provide a detailed specification of a system behaviour, may represent a general property of a system, or specify a specific constraints of a system, or algorithmic information to system engineers and so on. In general, requirements are classified as functional requirements, non-functional requirements or domain requirements. Functional requirements define the function of the system or its components. Non-functional requirements are the criteria for checking the system's operation rather than its specific behaviour. Non-functional requirements such as usability, integrity and security have more impact on software systems than do the functional requirements (Pohl, 2010). The domain requirements are inherited from the features of an application domain or define the rules and regulations that have to be applied to that domain.

Requirements Engineering (RE), an early phase of the software development life-cycle involves activities such as goals elicitation to be accomplished by the conceived system, the operationalization of such goals according to service and constraint specifications, the allocation of tasks (for the fulfillment of requirements) to agents such as humans, devices and software, and the evolution of these requirements over time. However, the RE process may vary depending on the application domain, the people involved and the organisation developing the requirements. However, RE follows a systematic approach to achieve a complete and consistent set of requirements for the achievement of goals. Obtaining the correct requirements is an important activity in software engineering process. Obtaining the correct and quality requirements is a challenging and critical task. The recent literature surveys on requirements have shown that RE is a primary area of concern in software engineering research and practice (Van Lamsweerde and Letier, 2004).

Currently, the term 'goal' is used in requirements engineering technology. A goal is defined as an objective to be achieved by the system under consideration (Van Lamsweerde, 2001, 2000). Goals are obtained from stakeholders, disclosed in requirements documents, analysis of similar or existing systems, on elaborating other goal models and so on. Goals may be specified at different levels of abstraction varying from high-level, strategic concerns to low-level, technical concerns. Goals also include different types of concerns: behavioural goals

(functional requirements) prescribe the services to be provided by the system and softgoals (non-functional requirements) are concerned with quality of service such as accuracy, performance, security and so forth.

In software engineering, goals have been used to model early requirements and non-functional requirements (Giorgini et al., 2002). In RE literature, a variety of methods have been proposed to model requirements, namely conceptual entity-relationship modeling, structured modeling, object-oriented, use case and goal-oriented approaches. Goal-oriented requirements engineering (GORE) is complementary when compared to traditional approaches (Mylopoulos et al., 1999). Compared with other approaches, GORE is appropriate for requirements analysis with regard to non-functional requirements (NFR) early in the software development cycle and is suitable for alternatives evaluation. The GORE approach is used to reveal, examine and express stakeholders' goals that lead to software and system requirements. The other approaches are more appropriate for requirements analysis later in the software cycle and target the traceability between requirements and implementation.

2.2 Goal-Oriented Requirements Engineering

Goal-oriented requirements engineering (GORE) is involved with the application of goals for eliciting, elaborating, structuring, specifying, analysing, negotiating, documenting, and modifying requirements (Van Lamsweerde, 2004). Goals are the stakeholders' objectives or requirements to be accomplished by the system under consideration. Here the word *"system"* refers to the software-to-be together with its environment (Van Lamsweerde and Letier, 2004). The goals are represented using AND/OR structures that show how the goals are refined or abstracted. Goals may also specify the functional or non-functional requirements and are categorized from high-level ones to lower-level ones. System goals relate to the application's specific safety, fault tolerance and survivability properties that are to be achieved for high assurance systems. Goal modeling and reasoning are particularly crucial for such high assurance systems.

The popular GORE frameworks are: Non-Functional Requirements (NFR) framework (Chung et al., 2012), Knowledge Acquisition in Automated Space (KAOS) (Dardenne et al., 1991), i^* framework (Yu, 2011), Tropos (Bresciani et al., 2004), Goal-Oriented Requirement Language (GRL) (Amyot et al., 2010) and Attributed Goal-Oriented Requirements Analysis (AGORA) (Kaiya et al.,

2002). In the following section, we give a brief description of each framework.

2.2.1 KAOS Framework

KAOS (Van Lamsweerde and Letier, 2004) is a requirement engineering methodology designed by van Lamsweerde and others in 1990 in collaboration with the University of Oregon and the University of Louvain. It assists the requirement analyst to construct requirements models and to develop requirements documents from KAOS models. Objectiver is a tool designed to support KAOS. With KAOS, the system goals are obtained by understanding the existing technical documents, consulting current and future users, and examining the current systems, and so on. The collected goals are structured into directed, acyclic graphs by the analyst. In the graph model, each goal (except the roots: the top-most strategic goals) is rationalized by at least another goal, and each goal (except the leaves, the bottom goals) is refined as a collection of subgoals. The goals at the top of the graph represent business or strategic goals and the bottom represents the system requirements. These goals are expressed in stakeholders' words. The identification of goals is done using either a top-down or bottom-up approach. The analysis is carried out as follows: analysts first determine the intermediate goals and then by asking " why do we want that?" they look for higher-level reasons for each new goal. More specific goals emerge from asking "how shall we attain that objective?". In general, the KAOS goal model can be considered as a collection of interrelated goal diagrams for handling a particular problem.

2.2.2 Non-Functional Requirements Framework

The Non-Functional Requirements Framework (NFR) (Mylopoulos et al., 1992) is a framework for representing and using non-functional requirements during the development process. The functional and non-functional requirements together determine the complexity of a software system. The non-functional requirements, which are informally stated during requirement analysis are challenging to implement during software development and difficult to validate once the final system has been built. The NFR framework is used as a tool for formal representation of non-functional requirements. The formal representation of non-functional requirements is defined using two approaches: the product-oriented approach and the process-oriented approach. In product-oriented formalization, formal definitions of non-functional requirements are developed in order to evaluate the degree

to which a software system meets its requirements. On the other hand, in the process-oriented approach, techniques are developed to justify design decisions during the software development process. Design decisions either positively or negatively influence the specific non-functional requirements. These dependencies are used to verify whether or not a software system indeed meets a certain non-functional requirement.

The non-functional requirements, also known as softgoals are difficult to express, although they contribute to the global qualities of a software system. The qualities could be precision, performance, accuracy, security, and usability of a given system. These softgoals are then usually streamlined through refinement methods to reveal a hierarchy of goals and softgoals. The tree structure can then be evaluated to resolve the extent to which a set of non-functional requirements is supported by a particular design.

2.2.3 Goal-Oriented Requirements Language

Goal-Oriented Requirements Language (GRL) (Amyot et al., 2010) is a modeling language that is used for decision making and rationale documentation. In the goal graph, the AND decomposition represents system functionality and the OR decomposition represents the different ways of performing these goals. The system stakeholders and their non-functional requirements are captured by actors. In GRL, the goal diagram incorporates a scenario notation for qualitative and quantitative attributes. It also separates GRL elements from their graphical representation which facilitates a consistent and scalable representation of multiple views/diagrams of the same goal model. For quantitative analysis, GRL uses numbers in the range ([-100..100]) to denote contributions, satisfaction values, and the importance of intentional elements to their containing actor. There are also qualitative scales for contributions { Break, ..., Make }, satisfactions { Denied,..., Satisfied }, and importance { None, ..., High }. However, for quantitative analysis, it is hard to assign suitable values to goals. The tool jUCMNav has features to handle this situation. GRL syntax is based on the i^* language and the goal diagram represents stakeholders' high-level business goals and non-functional requirements. The stakeholders' important beliefs (facts) are also shown in the goal diagram.

2.2.4 Attribute Goal-Oriented Requirements Analysis Method

Attribute Goal-Oriented Requirements Analysis Method (AGORA) (Kaiya et al., 2002) is an extended version of the AND-OR goal graph; stated precisely, it is an attributed AND-OR goal graph. The extended part includes:

- Nodes and edges with attribute values: These attributes include details of structural characteristics of the graph plus the variables that help to estimate the quality of the requirements specification.
- Rationale: It specifies the reasons for the decomposition of goals into sub-goals. It is attached to an attribute as well as to a node and an edge

To construct an AGORA graph, the following steps are used.

1. Choosing customer's needs as initial goals
2. Goals decomposition and refinement into sub-goals
3. Selection of goals from the alternatives of decomposed goals
4. Analysing the conflict of goals

2.2.5 i^* Framework

The i^* framework introduced by Yu (2011) models the social elements of a system and can be used in the early stages of requirements analysis. The Strategic Dependency (SD) model and the Strategic Rationale (SR) model are the two types of diagrams employed in modeling. The Strategic Dependency diagram represents stakeholders' relationships, while the Strategic Rationale diagram represents the internal intentional relationships of stakeholders. An SD model is a graph in which the nodes represent the actors and the links represent the interdependency between the actors. Goal, softgoal, task and resources are the intentional elements. A dependency can be any one of the intentional elements. An SD model is a higher level of abstraction representing the actors' dependency upon each other. An SD model targets external relationships and does not disclose the details of the internal structure. In the SD model, actors are represented by circles, hard goals by ovals, softgoals by cloud symbols, resources by rectangles, and tasks by hexagonal shapes.

An SR model assigns the intentional elements' goals, tasks, resources and softgoals to actors. It describes how actors achieve their goals. Intentional elements are linked by MEANS-END relationships, TASK decomposition and softgoal contributions. An SR model can be viewed as a graph that shows the decomposition of high-level goals into lower level goals by the MEANS-END / TASK decomposition. In means-end relationships, a mean node can represent a softgoal or a task, and an end node can be a goal, a softgoal, a resource or a task. A means-end links a task to a goal, implying that a particular method is used to achieve a goal. Task decomposition shows the sub-goals, resources and softgoals that are to be carried out to ensure the success of a task. A softgoal contribution can be any of the following types: *help*, *make*, *some+*, *some-*, *hurt*, or *break*.

The early analysis of the i^* model not only answers the 'what' and 'how', but also the 'why' questions involved in software development. Hence, it performs a refined analysis of software dependencies and encourages a uniform treatment of the system's functional and non-functional requirements.

2.2.6 TROPOS

Tropos (Bresciani et al., 2004) is an agent-oriented software methodology. It borrows the i^* modeling framework proposed by Eric Yu. The idea of actor, goal and dependencies are used to model early and late requirements. The Tropos methodology was intended to assist with all analysis and design activities in the software development activity. Requirement analysis comprises two phases: early requirements analysis and late requirements analysis. In the early requirements phase, the domain stakeholders are identified by the analyst who models them as actors with their dependencies on other actors and resources. In the late requirement phase, the new actors and their dependencies are added, thereby extending the model. All the functional and non-functional requirements of the system are defined. The system specifications resulting from the requirements phases are addressed in the architectural and detailed design phase. Following the detailed design specification, the implementation activity is carried out step-by-step in order to map between the implementation platform constructs and the detailed design.

In the following section, we briefly describe some of the goal analysis procedures proposed for various GORE frameworks in the existing RE literature.

2.3 Goal Analysis

Goal models are developed to support qualitative or formal reasoning of goals during requirements engineering. A goal model is an AND/OR graph depicting how higher-level goals are satisfied by lower-level ones and how lower-level goals contribute to the satisfaction of higher-level ones (Van Lamsweerde, 2009).

In addition to modeling, analysts use goal models to find the level of goals satisfaction, to evaluate design alternatives, to choose the system design, analyse risk and decide the requirements' prioritization. During alternative design evaluation, analysts explore different design alternatives and select the best ones using several evaluation criteria. Softgoals in goal models are used as evaluation criteria in existing qualitative and quantitative approaches (Mylopoulos et al., 1992). Many qualitative and quantitative reasoning approaches have been proposed in RE literature to support the goal analysis (Liaskos et al., 2011; Amyot et al., 2010; Franch, 2006; Horkoff and Yu, 2009; Miller et al., 2014; Waters et al., 2015; Ashamalla et al., 2017; Burgess and Krishna, 2009; Burgess et al., 2009). During the evaluation process, the qualitative or quantitative values are propagated from the bottom softgoals to the top softgoals in a goal model.

In qualitative analysis, qualitative weights such as *positively* and *negatively* are used for describing the contribution of goals to softgoals. Qualitative labels such as *satisfied*, *denied*, *partially satisfied* and *partially denied* are assigned to the nodes in a goal model. These labels are propagated through link paths to find satisfaction of goal achievement. In qualitative label propagation, there is a chance of getting one or more alternative options with the same label, in which case decision making is difficult. Moreover, in some cases, decision making is uncertain when an option has a U (Undetermined) label. To address the limitations of qualitative reasoning, a measurable specification for goals, namely quantitative reasoning, has been proposed in RE literature. In the quantitative approach, quantitative estimations are used to represent the contribution of goals to softgoals and quantitative labels are propagated through link paths to find the level to which the goal has been satisfied. Since the development of the concept of a goal model, considerable amount of research work has been undertaken on reasoning related to goal achievement using qualitative and quantitative labels(Liaskos et al., 2011; Amyot et al., 2010; Franch, 2006; Horkoff and Yu, 2009; Goncalves and Krishna, 2016b; Affleck et al., 2013, 2015).

2.3.1 Qualitative Analysis

2.3.1.1 Reasoning about Alternative Requirements

Van Lamsweerde (2009) initially proposed a qualitative reasoning approach for alternative evaluation. The qualitative procedure is as follows: first, the contribution of each alternative to the various softgoals is qualitatively assessed. Then these contributions are propagated upwards in the softgoal graph through refinement marked as + or ++ according to the strength of the positive contribution and conflict links marked as - or - - according to the severity of the negative contribution. The propagation is carried out recursively until the top-level softgoals receive a single label. The disadvantages of this approach are

1. The propagation rule sometimes turns labels into 'inconclusive'
2. The labels and link weights have no clear meaning in terms of system-specific phenomena.
3. It provides a rough evaluation of goals.

To address these limitations, Lamsweerde came up with a lightweight quantitative alternative evaluation system by integrating the notions of softgoals and goals into the KAOS framework. The contributions of alternatives to all the leaf softgoals are assessed using quantitative estimations. The leaf softgoals are assigned different weights to show their relative importance. Based on system phenomena, each option is scored against the leaf softgoals. A weighted matrix is used to collect the weights and scores for overall comparison. In his approach, Lamsweerde used variables such as gauge variable, ideal target value, and maximum acceptable value, for each softgoal. These values are obtained from the specification of the system. So, in order to design a goal model using this method, a thorough knowledge of the specification of the system is required. Another problem with this approach is that it may be difficult to apply to complex and large systems. The limitation of this approach is the ambiguity that arises when two or more goals receive the same label, creating confusion in decision making.

2.3.1.2 Exploring Alternatives during Requirements Analysis

Mylopoulos et al. (2001) proposed an approach to explore alternatives and to evaluate their feasibility and desirability with regard to business-goals-based goal-oriented analysis technique. The analysis is done using an AND/OR decomposition graph and consists of five steps:

1. Goal analysis
2. Softgoal analysis
3. Softgoal correlation analysis
4. Goal correlation analysis and
5. Evaluation of alternatives.

The approach is explained with meeting scheduler case study. Once the graph showing the goal and softgoal decomposition has been constructed using the first four steps, the evaluation of functional goal decomposition is done in terms of softgoal hierarchies. Evaluation is done by choosing a set of softgoals that altogether satisfy all given goals or at least support the best overall satisfaction for top-level softgoals.

2.3.1.3 Reasoning with Goal Models

Giorgini et al. (2002) presented a formal reasoning of goals in goal models using a qualitative formalization and label propagation algorithm. The work aimed to model a framework for goals to incorporate qualitative goal relationships and also to include contradictory situations. This is done by introducing goal relationships labelled "+", "-" to represent a goal's positive and negative contribution towards satisfying another goal. This labelling required a precise semantics representation of the new goal relationships. This is accomplished in two different ways: labelling the propagation algorithm with a qualitative formalization, and labelling the propagation algorithm with a quantitative formalization. Both algorithms have been implemented in Java. Two sets of experiments were carried out: the first one for the qualitative label propagation algorithm, and the second one for the quantitative label propagation algorithm. In the qualitative algorithm, a set of labels was assigned to some of the goals and events and the consequences of other nodes were noted. A steady state was reached after, at most, five iterations. In the quantitative algorithm, the relationships "+", "-" and "-s" are assigned numeric weights. Using the propagation algorithm, the final values are computed for each goal/event. These results are confirmed by the results obtained from the qualitative algorithm. However, more precise conclusions about the final value of the goals/events are obtained with the numeric approach. In addition, quantitative semantics for new relationships are based on the probabilistic model. This work requires sound mathematical knowledge, as it uses first order logic.

2.3.1.4 Evaluating Goal Achievement in Enterprise Modeling - an Interactive Procedure and Experiences

Horkoff and Yu (2009) proposed a qualitative analysis of goal- and agent-oriented models to comprehend the problem domain during the early phase of requirement engineering. In addition to understanding the problem domain, they also introduced an interactive evaluation procedure for alternatives evaluation which require customer intervention. The alternatives may be a system alternative or a process design choice or alternative courses of actions, capabilities, and commitments. For easy understanding and manual analysis, an informal procedure in terms of the i^* framework has been presented. The analysis starts with the question "How effective is an alternative with respect to model goals?". The intentional elements are assigned a degree of satisfaction or denial using a set of qualitative labels. Using the propagation algorithm, these qualitative labels are propagated through the model links. When multiple conflicting or partial values arise, human judgement is used to find the satisfaction or denial of a softgoal. Each actor's final satisfaction and denial values are analysed using the original question. The analysis checks whether a design choice is satisfied ("good enough") or not, and allows further model analysis and refinement. An experimental study has been undertaken where this procedure has been applied to several case studies. The experimental results showed that the procedure provided a better understanding of the model and domain. However, the experiment suffered from several issues of validity, including the small number of participants. However, the main drawback with their approach is the ambiguity of the decision-making when one or more goals receive the same label.

2.3.2 Quantitative Analysis

2.3.2.1 Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering

Letier and Van Lamsweerde (2004) proposed a more accurate, but heavy weight approach based on the interpretation of numbers in terms of probability. Lamsweerde et al. (2004) presented a method for determining the partial degrees of goal satisfaction and for quantifying the impact of system alternatives on high-level goals that are partially satisfied. Both qualitative and quantitative reasoning methods are used to evaluate alternatives with respect to degrees of goal satisfaction. Bayesian Networks concepts are used for making predictions about

softgoals. A semi-formal but precise way of determining partial goal satisfaction is represented by objective functions and quality variables. Quality variables are domain-specific goal-related random variables defined for a specified sample space. Again, objective functions are also domain-specific goal-related quantities to be maximized or minimized. Using five heuristics, the appropriate objective functions and quality variables are specified correctly for alternative design evaluation. Probabilistic extension of temporal logic is used to specify objective functions more precisely at the optimal formal layer. The propagation rules that relate the quality variables of subgoals to the quality variables of parent goals are used to compute the objective function values for each alternative. The actual calculation of objective functions is done through ad-hoc use of mathematical software. This may turn out to be difficult for complex refinement equations. Dedicated tools to perform such computations more effectively should be provided. The author also states that the framework should be extended to handle uncertainties on parameter estimations by using confidence intervals. This approach is difficult to apply to a complex system.

2.3.2.2 Evaluating Goal models within Goal-oriented Requirements Language

Amyot et al. (2010) developed a hybrid approach by combining the qualitative and quantitative approaches to perform an analysis of the GRL model to evaluate the satisfaction levels of the actors and the intentional elements. The algorithms are illustrated using the example of a telecommunications system. The evaluation is done by attaching satisfaction values to a subgroup of intentional elements and, by means of a propagation algorithm, these values are propagated through decomposition, contribution and dependency links to other intentional elements. This is carried out at different times for a different subset of intentional elements by assigning different intentional strategies for the same GRL model. In the evaluation, two evaluation values namely, one that is qualitative and one that is quantitative (integer value in range [-100...100]), are selected. Initially, these two attribute values are set to None and 0 respectively for all other intentional elements. The jUCMNav tool, an Eclipse-based editor for URN models, implemented three evaluation algorithms, namely qualitative evaluation, quantitative evaluation and hybrid evaluation. The tool also implemented three propagation algorithms, namely forward propagation, backward propagation and mixed propagation. In mixed propagation, the evaluation starts from intentional elements

that are neither roots nor leaves. Some of the criteria that are checked during evaluation are: actor satisfaction, cycles, automation, conflicts, strategy consistency, and evaluation overriding. This work does not address the generation of a good goal model, which is often linked with the requirements elicitation and analysis process. Moreover, when stakeholders' requirements are vague, it is difficult to assign exact numeric numbers to requirements using quantitative analysis.

2.3.2.3 On Eliciting Contribution Measures in Goal Models

Liaskos et al. (2012) explained the application of a mathematical method, namely Analytic Hierarchy Process (AHP), to quantitatively prioritize goals-based stakeholder input. It aimed to examine the similarity between goal hierarchies and criteria hierarchies, by which AHP organizes priority elicitation. This requires that the goal model satisfy certain structural features. The approach uses five steps to perform priority elicitation. The formation of the criteria hierarchy is the first step. The objective to be achieved is the top element of the hierarchy and is called the 'decision goal'. In the second step, the relative importance of the sibling criteria or alternatives is obtained by comparing the elements at each level of the hierarchy. These comparison values are stored in an $n * n$ comparison matrix. In the third step, the local weights are obtained by transforming the comparison matrices into weighted priority profiles of the involved items. Thus, each element in the hierarchy tree obtains a real number ranging from 0 to 1, and this represents the contribution of each element to its parent. In the next step, the local weights are aggregated to obtain global weights for the decision alternatives. The global weights give the ranking and the suitability of each alternative. The higher the ranking of an element, the greater is its suitability. An experimental study has also been conducted to show the feasibility of the approach. The limitation of this approach is that it requires certain structural features to be satisfied by the model for goal analysis.

2.3.2.4 Representing and Reasoning about Preferences in Requirements Engineering

Liaskos et al. (2010, 2011) suggested a framework to indicate preference requirements and their prioritization. This framework is used to determine the specifications that achieve mandatory requirements while best satisfying preference requirements and priorities. It differentiates mandatory goals from preference goals and also offers a method of determining alternative ways to achieve mandatory

goals and the fulfilment or non-fulfillment of preference goals. By using a quantitative requirement prioritization method such as the Analytic Hierarchy Process (AHP) weights of importance are assigned to preference goals to obtain an optimized preference function. The alternative plans for fulfilling mandatory goals and optimized preference functions are obtained by means of a powerful preference-based planner. The formal planning domain definition language (PDDL) 3.0 was used to specify Artificial Intelligence planning problems and to help in the definition of hierarchical task networks (HTN). This HTN and PDDL-based reasoning tool is useful for exploring alternative designs during early requirements stages, supporting priority elicitation activities, improving domain understanding and model accuracy and supporting customization of software systems. This is accomplished by integrating the high-level design descriptions obtained through the tool into configurations of various points in the software itself. The disadvantage of this approach is that it cannot be applied to larger frameworks.

2.3.2.5 On the Quantitative Analysis of Agent-Oriented Models

Franch (2006) presented an analysis of agent-oriented models with an emphasis on the quantitative aspect. To explain his approach, he used the i^* language to construct agent-oriented models. The conceptual model of the i^* is represented using Unified Modeling Language(UML) and Object Constraint Language (OCL) is used to express the framework. The quantitative method is used to determine the structural indicators that are used to define the structural metrics. The structural metrics are used to measure properties of an i^* model such as actors, dependencies and other elements. They have illustrated with an example how the indicators are used to obtain the properties of a system. This method requires some sort of expert judgement when deciding the alternative to be selected. This method is not completely quantitative in nature since it requires some degree of qualitative reasoning to obtain accurate information.

2.3.2.6 Utilizing TOPSIS: A Multi-Criteria Decision Analysis Techniques for Non-Functional Requirements Conflicts

Mairiza et al. (2014) proposed a Multi Criteria Decision Analysis (MCDA) for resolving the conflicts in NFR decision analysis for sureCM framework ie., an integrated experimental-based framework for NFRs conflict management and Analysis. The evaluation and analysis of alternative design solutions are performed using MCDA. This approach also finds the best design solution that best satisfy

the conflicting NFRs using MCDA. A goal-based technique in MCDA, namely TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution), is used to find an alternative that is closest to the ideal solution. This research does not provide any evaluation of the approach.

2.3.2.7 Applying Fuzzy Preferences Relation for Requirements Prioritization in Goal-Oriented Requirements Elicitation Process

Sadiq and Jain (2014) presented a method for requirements prioritization, called Prioritization of Requirements using a fuzzy-based approach in Goal-Oriented Requirements Elicitation (PRFGORE). It applied the concepts of α-level weighted F-preference relation, a fuzzy-based Analytical Hierarchy Process (AHP) for group decision making and a binary sort tree method to obtain a list of the prioritized requirements. The AHP pair wise comparison is used for assigning weights to goals/softgoals and locates the list of prioritized requirements using the binary sort tree method. Fuzzy preference relation is used to combine experts' preferences with group preferences. The approach has been demonstrated with a small number of requirements and criteria. This approach requires sound mathematical knowledge for the prioritization of goals.

2.3.2.8 Optimal Requirements-Dependent Model-Driven Agent Development

Goncalves and Krishna (2016b) proposed a quantitative approach for operationalization in Extended Non-Functional Requirements framework (ENFR). To determine the most appropriate operationalization, the preferences of operationalization and the progressive value of its children within the ENFR model are used. The authors have proposed a method for finding an optimal path with regard to the complexity time and space that has to be followed for any combination of operationalization at any particular time. A simulation was created and tested using a banking system case study. The authors extended their work (Goncalves and Krishna, 2016a, 2015) by incorporating change management in agents. The decision path is used to find the agents that are affected when any changes (like change in softgoal weight, change in contribution values) occur in an agent. An optimisation model was proposed based on probability, and an evaluation was carried out. The evaluation results indicated that this approach cannot be applied directly to the ENFR. This process requires changes to be made to the original model in order to incorporate the proposed concepts.

2.3.3 Optimisation in Goal Analysis

2.3.3.1 Simulating and Optimising Design Decisions in Quantitative Goal Models

Heaven and Letier (2011) extended their previous work by applying multi-objective optimisation to the KAOS goal model for exploring the alternative design options. In their earlier work, a formal semantics and a set of heuristics for the goal models were presented. However, this did not automate the model analysis and is not applicable to a model with very a large number of design alternatives. In this research, these limitations are addressed by providing an automated technique for the goal model with a large number of design alternatives and identification of optimal alternatives among them. To simulate the complete set of alternative designs in the given goal model, a stochastic simulation model is generated. The input to the simulation is a particular set of design choices and a sample size. The behaviour of that particular design is simulated using probability distributions and equations of the quantitative goal model. Also, the goal satisfaction levels for the simulation are obtained. Optimal design choices are identified using a multi-objective optimization component. A MATLAB simulation-based illustration was presented using the London Ambulance Service goal model. The drawback of this approach is that there is no systematic technique for specifying the objective functions.

2.3.3.2 Non-functional Requirements Framework: A Mathematical Programming Approach

Affleck et al. (2013, 2015) proposed a linear programming optimisation model for the NFR framework. This approach aimed to minimise the operationalization. The optimisation model is an extended version of their original work (Affleck and Krishna, 2012). The initial work presented a quantitative extension to the NFR framework to support the decision-making process. In the quantitative NFR framework, weights are assigned to the leaf softgoals in the softgoal graph and to links between leaf softgoals and operationalization. Using these weights, the operational scores, leaf softgoal scores, and the optimal and actual attainment scores are calculated. The operation selection is done based on operational scores. In the extended version, the linear programming method is used to mathematically express an objective function in terms of decision variables. The constraints are the restrictions to be satisfied by these decision variables. To a linear solver, the

objective functions, variables and constraints are given as an input to compute the results. The aim is to find an optimal set of operationalizations that produces the maximum satisfaction of leaf softgoals. It is also presented as a minimization or maximization problem along with the proof. Also, a procedure for the propagation of the leaf softgoals to the root softgoals in the goal graph was included in the extended model. A simulation was implemented using C# and LPSolve. The simulation showed that the process was effective in graphs where there is a large number of relationships between softgoals and operationalization. To do so, it used single-objective optimisation to select the minimum number of operations that maximises the overall satisfaction of non-functional requirements. Additionally, sensitivity analysis is performed to help developers find the quantitative input values. Sensitivity analysis finds the extent of input data for which there is no change in the final value. The developers are alerted if the potential value exceeds the range given, and take action accordingly. The limitation of the approach is that values assigned to leaf softgoals are subjective and it is difficult to assign correct values to the leaf softgoals.

2.3.4 Game Theory in Requirements Engineering

2.3.4.1 Game Theory Perspective on Requirement-Based Engineering Design

Yazdania et al. (2017) proposed a non-cooperative game theory for requirement-based engineering design to improve the suboptimal performance of a software project. During software development, the complex systems are hierarchically divided into subsystems. The system-level requirements are also decomposed into subsystem-level requirements to be satisfied by each design team. The resources shared by the subsystems are seen as system-level requirements. The entire system performance depends on the effective allocation of these resources. Since each design team wants to satisfy its own requirements specification, the system performance will be affected by poor decomposition of system-level requirements. It also restricts the design alternatives. This may not result in an optimal design outcome. To improve the optimal results of sub-designs, the author applied a non-cooperative game theory for the theoretical analysis of the performance of requirements-based design engineering. While this approach uses game theory for requirements design engineering, it cannot be applied to goal analysis.

2.3.4.2 Applying Game Theoretic Approach to Goal-driven Requirements Trade-Off Analysis for Self-Adaptation

Lee and Lee (2015) proposed a game-theory-based approach to handle conflicts among requirements in a self-adaptive software environment. The authors state that the satisfaction degree of quality requirements is considerably affected when self-adaptive software is adapting to changes in the environment. To analyse the trade-off obtained from requirements in an adaptive environment, game theory is applied. The approach is explained using a N-tier client-server architecture system. A goal model in a normal form is constructed to analyse conflicts of adaptive actions. The author also states that the model has to be refined to include the dynamics correctly. Although this approach is based on goal-driven requirements, it does not deal with goal analysis.

2.4 Summary of the Research Review and Purpose of this Research

In software engineering, the two main approaches for data analysis are the qualitative approach (Dybå et al., 2011) and the quantitative approach (Lázaro and Marcos, 2006). Qualitative approaches use text and picture representation of data in the analysis. Qualitative methods are used to study the complexities of human behaviour namely, motivation, communication and understanding. These methods were devised by educational researchers and other social scientists. However, in quantitative analysis, the numerically-presented data obtained from a sample are used and examined using statistical methods. These approaches reduce the complexity of the problem by identifying independent and dependent variables, and by eliminating irrelevant variables. Both qualitative and quantitative approaches have their own strengths and limitations as shown in Table 2.1.

From the review of the goal analysis literature, we find that both qualitative and quantitative approaches are used. Qualitative analysis methods use qualitative labels such as *satisfied, denied, partially satisfied, partially denied, unknown* and *conflict* in the propagation algorithms to find the satisfaction of goals. Also, in some researches, the symbols "+", "-" are used to represent a goal's positive and negative contribution towards the satisfaction of another goal. Even though, the qualitative algorithms provide easier, more simple and cheaper ways

of performing goal analysis, they have several limitations. The main drawback of qualitative approaches is the ambiguity that arises when two or more alternatives contribute to the same label, thereby creating problems for decision making. Moreover, there is confusion when a goal receives an *unknown* or *conflict* label. The second limitation is that the labels and link weights have no clear meaning in terms of system-specific phenomena. Several approaches require system knowledge and sound mathematical knowledge. Some approaches are difficult to apply to complex and large systems.

The problems of qualitative analysis were overcome by the quantitative analysis methods. Numeric representations, probabilistic interpretation of numbers, Analytic Hierarchy Process (AHP) representation of goals were used in quantitative goal analysis. Despite overcoming the limitations of qualitative analysis, quantitative analysis also has its own drawbacks. The probabilistic extension of temporal logic requires strong mathematical knowledge, and the AHP method requires certain structural features to be satisfied by the goal model. The main issue with quantitative analysis is that it is difficult to assign definite numbers to stakeholders' requirements as requirements analysis may involve various stakeholders with diverse preferences regarding the same requirements. Moreover, in reality, terms such as *low cost*, and *high profit* are generally used by the stakeholders to communicate their requirement preferences. These linguistic terms are challenging to represent by qualitative labels as used in qualitative analysis and also difficult to represent by definite numbers as used in quantitative analysis. Hence, there is a need for a method to deal with linguistic terms of requirements. This dissertation aims to fill this gap by using fuzzy numbers for the linguistic representation of stakeholders' requirements.

Further more in real-world situations, the input data used to evaluate the requirements are imprecise due to incomplete or unobtainable information. The inputs used in the analysis are also subjective from the perspective of the requirements analyst. Moreover, most of the real-world business problems encounter the simultaneous optimisation of many competing objective functions. The literature review reveals that the existing goal analysis approaches do not address these problems in their goal analysis procedures. To address these research gaps, we have proposed methods based on multi-objective optimisation and goal programming to handle imprecise requirements and to avoid subjective preferences. To address the competing objective functions or requirements with opposing objective functions, a method based on game theory was proposed in this thesis.

Also from the literature review of goal analysis, we infer that to date there has

been no research work on the quantitative analysis of goals in the i^* framework. Hence, the i^* framework is chosen as a framework for implementing our novel approaches.

A summary of the above research approaches with their strengths and limitations are shown in Tables 2.2 and 2.3.

2.5 Chapter Summary

In this chapter, we provided an overview of requirements engineering, goal-oriented requirements engineering and goal analysis procedures by briefly presenting the relevant definitions, and key concepts, and emphasizing its significance in software development. We have explained the two major approaches used for goal analysis. We have also reviewed various approaches that have been proposed in RE literature to address qualitative and quantitative goal analysis. We have also discussed the advantages and disadvantages of those approaches in order to highlight the gaps that our work intends to address. In the next chapter, we begin describing our fuzzy-based quantitative work with a description of the i^* framework.

Table 2.1: Analysis Approaches in Software Engineering,- Strengths and Limitations

Approach	Based on	Strengths	Limitations
Qualitative Approach (Dybå et al., 2011)	♦ Use data that are described as text, pictures, not numbers. ♦ These methods were used to study the complexities of human behaviour namely, motivation, communication and understanding. ♦ These methods were devised by educational researchers and other social scientists.	♦ Help the researcher to investigate the complexity of the problem instead of abstracting it away. ♦ Rich and more informative results are obtained. ♦ Help to find responses to queries that contain variables that are hard to quantify (principally human characteristics such as motivation, perception, and experience). ♦ Used to answer the *"why"* questions.	♦ Usually more labour-intensive and exhaustive ♦ Qualitative results often are considered softer, or fuzzier than quantitative results, especially in software engineering communities. ♦ Results are more hard to summarize or simplify.
Quantitative Approach (Lázaro and Marcos, 2006)	♦ Use numerical form of data obtained from a sample and are examined over statistical methods. ♦ Reduces the complexity of the problem by identifying independent and dependent variables, and by eliminating inadequate variables.	♦ Results can be generalized. ♦ Results are of great accuracy. ♦ Informations can be summarized. ♦ Avoids personal bias.	♦ Results are limited namely numerical descriptions, no human perception. ♦ Laboratory environment does not correspond to real world. ♦ Cannot be used in all circumstances.

Table 2.2: Goal Analysis Approaches,- Strengths and Limitations

Approach & Ref.	Concept used	Strengths	Limitations
Qualitative Methods (Van Lamsweerde, 2009) (Giorgini et al., 2002) (Horkoff and Yu, 2009) (Mylopoulos et al., 2001)	♦ Used Qualitative labels like satisfied, partially satisfied, denied, partially denied, unknown, conflict in the propagation algorithms. ♦ Or used labels like ++, +, – or - in the propagation algorithms.	♦ Its easy and simple to perform the propagation algorithm . ♦ It provides a quick means of goal evaluation.	♦ Sometimes the label turn into unconclusive in the propagation rule. ♦ The labels and link weights have no clear meaning in terms of system specific phenomena. ♦ Some approaches requires knowledge about system being developed. ♦ Sometimes, the approaches are difficult in complex and large systems. ♦ Formal reasoning of qualitative approaches require a strong mathematical knowledge. ♦ Leads to ambiguity in decision making when one or more goals receive same label.
Quantitative Methods (Mairiza et al., 2014) (Amyot et al., 2010) (Liaskos et al., 2012) (Liaskos et al., 2010) (Liaskos et al., 2011) (Franch, 2006)	♦ Approaches used numeric values, or probabilistic interpretation of numbers, or Analytical Hierarchy Process (AHP) to quantify goals, or used multi criteria decision analysis, or fuzzy based AHP in the goal analysis process. ♦ Used single objective, multi-objective optimisation for goal analysis.	♦ Avoids ambiguity in decision making.	♦ It's difficult to assign exact numbers as stakeholders requirements are sometimes vague. ♦ Probabilistic extension of temporal logic requires extensive mathematical knowledge. ♦ The AHP methods requires certain structural features to be satisfied by the model for goal analysis. ♦ Certain approaches cannot be applied to larger frameworks.

Table 2.3: Comparison of Goal Analysis Approaches(cont..)

Approach	Authors (Year)	Analysis Type	Involve Optimisation	Tool Support	Sensitivity Analysis	Game Theory
KAOS						
Exploring alternatives during requirements analysis	Mylopolus et al. (2001)	Qualitative	No	No	No	No
Reasoning about alternative requirements	Lamsweerde (2009)	Qualitative	No	No	No	No
Reasoning about partial goal satisfaction for requirements and design engineering	Lamsweerde et al. (2004)	Quantitative	No	No	No	No
Simulating and optimising design decisions in for quantitative goal models	Willam et al. (2011)	Quantitative	Yes	Presents a Simulation	No	No
NFR						
Utilizing TOPSIS: A multi criteria decision analysis techniques for non-functional requirements conflicts	D.Zowghi et al. (2014)	Multi Criteria Decision Analysis	No	No	No	No
Optimal Requirements Dependent Model-Driven Agent Development	Joshua et al. (2016b)	Quantitative	No	No	No	No
Non-functional requirements framework: A mathematical Programming approach	Affleck et al. (2015)	Qualitative	Yes	No	Yes	No
GRL						
Evaluating Goal models within Goal-oriented Requirements Language	D.Amyot et al. (2010)	Quantitative, Quantitative and hybrid	No	yes	No	No

Table 2.3: Comparison of Goal Analysis Approaches

Approach	Authors	Analysis Type	Involve Optimisation	Tool Support	Sensitivity Analysis	Game Theory
		i^*				
Evaluating goal achievement in enterprise modeling - an interactive procedure and experiences	Horkoff and Yu (2009)	Qualitative	No	Yes but involves human interaction	No	No
On eliciting contribution measures in goal models	Liaskos et al. (2012)	Analytic Hierarchy Process	No	No	No	No
Representing and reasoning about preferences in Requirements Engineering	Liaskos et al. (2011)	Analytic Hierarchy Process (AHP)	No	No	No	No
On the quantitative analysis of agent-oriented models	Franch (2006)	not completely quantitative in nature since it requires some degree of qualitative reasoning	No	No	No	No

Chapter 3

Inter-actor Quantitative Reasoning of the i^* Framework

The previous chapter gave some insight into the problem which this research attempts to address. The previous chapter briefly explained the concepts relevant to goals and goal-oriented requirement engineering. In addition, we presented an overview of the research area of goal analysis with a special focus on the approaches and techniques used for goal analysis. Furthermore, we reviewed the use of the optimisation techniques applied to goal analysis. In this chapter, we discuss how to develop a fuzzy-based method for quantitative analysis of the i^* framework. In Section 3.1, we introduce the concept of fuzzy numbers. An overview of the i^* framework is given in Section 3.2. In Section 3.3, we present our approach to provide a quantitative fuzzy-based reasoning of goals in the i^* framework. Section 3.4 describes a simulation in VC++ and presents an evaluation of our work using case studies in the existing literature.

Some of the material in this chapter has previously appeared in the following publications:

1. Chitra M Subramanian, Aneesh Krishna, Raj P. Gopalan and Arshinder Kaur (2015). Quantitative Reasoning of Goal Satisfaction in the i^* framework, The 27^{th} International Conference on Software Engineering and Knowledge Engineering (SEKE 2015), Pittsburgh, USA.

2. Chitra M Subramanian, Aneesh Krishna and Arshinder Kaur (2015). Reasoning about Goal Satisfaction for early Requirements Engineering in the i^* framework using Inter-actor Dependency, The 19^{th} Pacific Asia Conference on Information Systems (PACIS 2015), Singapore.

The success of any software system depends on the accurate requirement framing from the stakeholders, users and customers of the software. Requirement elicitation, an early phase of requirements engineering (RE) involves collecting requirements from the stakeholders, users and customer of the software. Sometimes these requirements are uncertain, vague or imprecise and therefore cannot be used for the analysis or modeling of requirements. To illustrate, to express a requirement quantitatively, a stakeholder may use phrases such as *"about twenty"* which is a practical, if vague, approach. These uncertainties in the requirements have to be represented in some form during analysis or modeling. In addition to this issue, another concern is aggregating the requirements of various stakeholders. Since a diverse group of stakeholders with conflicting requirements are involved in a software development, it is difficult to aggregate the requirements from several stakeholders. These vague, uncertain, inappropriate or conflicting requirements can be easily expressed in linguistic terms which are conveniently represented by fuzzy numbers. Due to their appropriateness for expressing uncertainty, fuzzy numbers and fuzzy values are widely used in many engineering applications. In the following section, an overview of fuzzy numbers is provided.

3.1 Overview of Fuzzy Numbers

Fuzzy set theory, introduced by Zadeh, is a technique used to handle the imprecision and vagueness associated with human decision making (Zadeh, 1965). A fuzzy number is considered as a quantity whose value is imprecise rather than definite as in the case of single valued numbers. The fuzzy numbers express the real world problem more practically than do single valued numbers. In general, a fuzzy set is defined as *"A collection of objects with graded membership between 0 and 1"* and represent a fuzzy set 'A' as $\{(x, \mu A(x))|x\varepsilon X, 0 \leq \mu A(x) \leq 1\}$, where μA(x) is a membership function. The membership function describing a fuzzy number helps to represent some vague idea related to that fuzzy number. The membership function is selected in a subjective way. Hence, different persons may select different membership functions to represent the same idea. It also depends on the context in which it is used. Fuzzy membership functions are continuous for many fuzzy numbers. But discontinuous membership functions are also used in certain specific cases. Examples of membership functions that are used to represent fuzzy numbers are Triangular, Trapezoidal and Gaussian functions. Triangular fuzzy numbers are used in the proposed approach because

starting with a triangular membership function is the simplest approach. Moreover, triangular fuzzy numbers represent fuzzy numbers, whereas trapezoidal fuzzy number represents fuzzy intervals. We give a brief introduction to triangular fuzzy numbers in the following paragraphs.

Triangular fuzzy numbers (TFN) are in the form $\bar{A} = (a1, a2, a3)$ and are diagrammatically presented as in Figure 3.1.

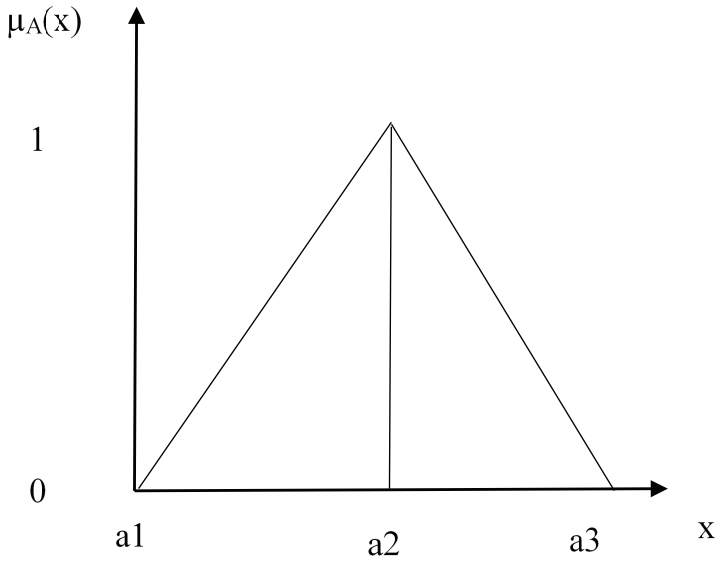

Figure 3.1: Membership function of TFN

The parameter $a2$ is the value where the membership function of a fuzzy number is 1.0, $a1$ is the left distribution of the confidence interval and $a3$ the right distribution of the confidence interval of the fuzzy number $\bar{A}$. The membership function for the interval is given as follows:

$$\mu\bar{A}(x) = \begin{cases} (x - a1)/(a2 - a1), a1 \leq x \leq a2 \\ (a3 - x)/(a3 - a2), a2 \leq x \leq a3 \\ 0, otherwise \end{cases}$$

Fuzzy logic supports many functions for performing fuzzy arithmetic. A few of the arithmetic operations that are performed on TFN are addition, subtraction, multiplication, division, and α - cut (Gani and Assarudeen, 2012).

Let $\bar{A} = (a1, a2, a3)$ and $\bar{B} = (b1, b2, b3)$ are two fuzzy numbers then,

Addition:

$$\bar{A} + \bar{B} = (a1 + b1, a2 + b2, a3 + b3)$$

Subtraction:

$$\bar{A} - \bar{B} = (a1 - b3, a2 - b2, a3 - b1)$$

Division:

$$\bar{A} \div \bar{B} = (\min(a1 \div b1, a1 \div b3, a3 \div b1, a3 \div b3), a2 \div b2, \max(a1 \div b1, a1 \div b3, a3 \div b1, a3 \div b3))$$

Multiplication:

$$\bar{A} \times \bar{B} = (\min(a1 \times b1, a1 \times b3, a3 \times b1, a3 \times b3), a2 \times b2, \max(a1 \times b1, a1 \times b3, a3 \times b1, a3 \times b3))$$

α - cut : is the set of elements whose membership values exceed the threshold level α.

$$\bar{A}_\alpha = \{x \mid \mu\bar{A}(x) \geq \alpha\}$$

A crisp interval of a fuzzy number $\bar{A}$ can be obtained by α - cut operation. Thus

$$\bar{A}_\alpha = [(a2 - a1)\alpha + a1, a3 - (a3 - a2)\alpha]$$

Having given a brief introduction to fuzzy numbers, we present an overview of the i^* framework in the following section.

3.2 Overview of the i^* Framework

Among the Goal Oriented Requirements Engineering (GORE) frameworks, the i^* framework captures the social elements of the system and can be used for reasoning, especially at the requirements level (Eric, 2011). In the i^* framework, a process or a system is comprehended by considering the intentional, strategic view of the process or the system. The intentional actor is the central entity to be modelled and the actors has motivations, intents and rationales behind its actions. Properties such as goals, beliefs, ability and commitment are used to characterise the intentional aspects of the actor. An actor is said to be a strategic actor when it is concerned about its structural relationships with other actors, in addition to performing its goal. It consists of two models:

- a Strategic Dependency (SD) model for depicting a specific configuration of dependency relationship between organisational actors and
- a Strategic Rationale (SR) model for depicting the rationales of that actor.

3.2.1 The Strategic Dependency Model

The Strategic Dependency (SD) model is intended to capture the intentional structures of the system, instead of usual non-intentional and non-strategic process models of activities and entities. The SD model is represented by a set of nodes and links, where the nodes represent the actors and the links represent the dependency. The depending actor is called the depender and the term dependee is used for the actor who is depended upon. Dependum is the term used to represent the object around which the dependency relationship centres. There are four types of dependency: goal dependency, task dependency, resource dependency and softgoal dependency. The SD model can be used to analyse the dependency relationships, namely who depends on whom for what, directly or indirectly. It can also be used to find who the stakeholders are and what their stakes are. The validation of the model is done by comparing answers to different questions to determine whether they comply with what is expected intuitively. An example of an SD diagram is shown in Figure 3.2. In this figure, actors are represented by circles, hard goals by ovals, softgoals by cloud symbols, resources by rectangles and tasks by hexagonal shapes.

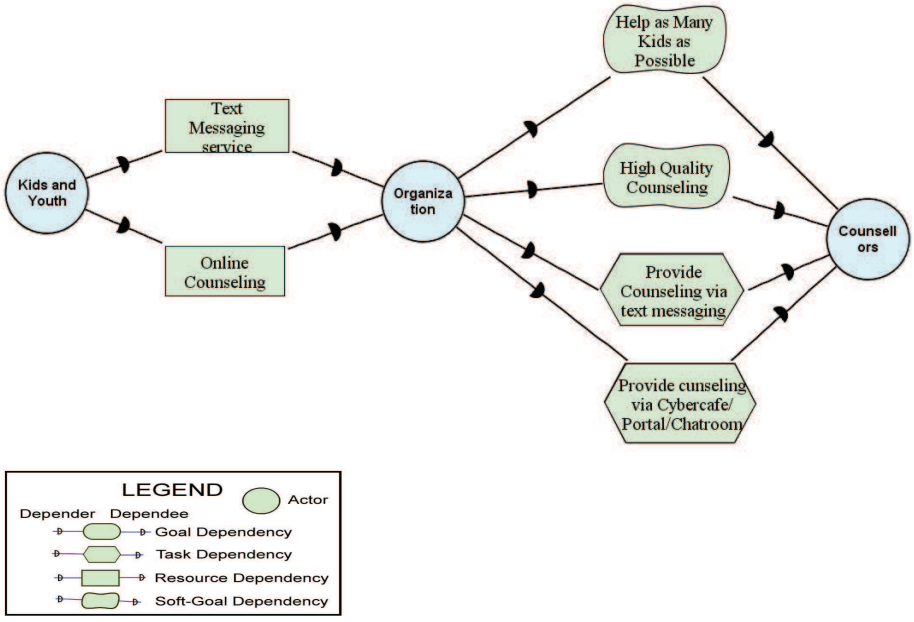

Figure 3.2: Example of Strategic Dependency model

3.2.2 The Strategic Rationale Model

The Strategic Rationale (SR) model explicitly describes the rationales of the system in terms of its elements and their relationships. The SR model is a graph with a set of nodes and links that depict structural representations of rationales. There are four types of nodes: goals, resources, softgoals, and tasks. The relationships between the nodes are denoted by means-end links and task-decomposition links. In means-end relationships, a mean node can represent a softgoal or a task, and an end node can be a goal, a softgoal, a resource or a task. A means-end links a task to a goal, indicating that a particular method is used to achieve a goal. A task node is connected to its component nodes by task-decomposition links. For the four elements of the model, there are four types of task decomposition links: subgoal, subtask, resource and softgoal. Task decomposition shows the sub-goals, resources and softgoals that are to be carried out to ensure the success of a task. A softgoal contribution can be any of the following types: *help, make, some+, some-, hurt,* or *break*.

In addition to modeling, the SR model can also be used in a number of computer-supported analysis design activities by considering their formal representations. An analysis can be done at the actor level to determine whether an actor knows how to do something, whether it will work, how well it will work and why the agents believe it will work. Systematic exploration of alternatives is also done during the design phase. An instance of the SR model is shown in Figure 3.3. The figure shows three actors: *Kids and Youth*, *Organisation* and *Counsellors*. The detailed intentional elements are shown for the actor *Kids and Youth*. There are inter-actor dependencies between the actors. The actor *Organisation* depends on actor *Counsellor* through softgoal dependency, *HighQualityCounselling*.

Apart from modeling, goal models assist the requirement analyst to assess the satisfaction of goals, to determine the high-level requirements and to assess design alternatives (Amyot et al., 2010). Many approaches which include both quantitative and qualitative analysis procedures have been proposed to assess the satisfaction of goals (Amyot et al., 2010; Horkoff and Yu, 2009; Van Lamsweerde, 2004). Both the KAOS and NFR the goal models show the goals' decomposition and there is no actor dependency. However, the i^*, Tropos and GRL models, include inter-actor dependencies. The existing literature has discussed the actor dependencies and formalisation of these dependencies (Morandini et al., 2007). However, in the current RE literature, these dependencies have not been considered in relation to calculating the goal satisfaction. This research presents

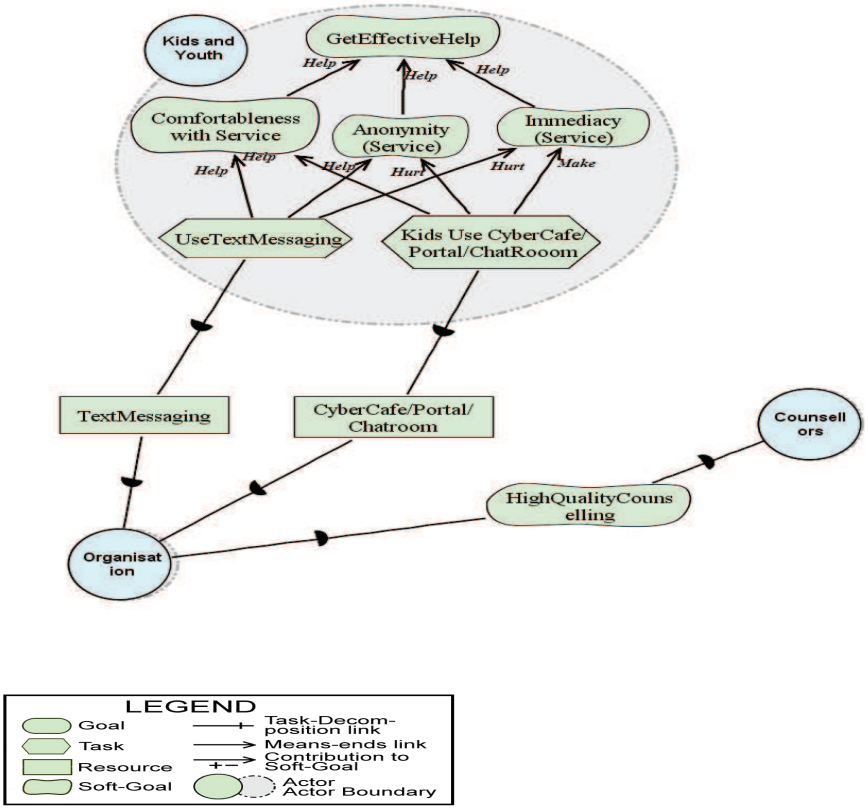

Figure 3.3: A Simple SR Model: Youth Counselling Example (adapted from Horkoff and Yu (2009))

an approach for finding softgoal satisfaction using the inter-actor dependencies of the i^* framework. It uses fuzzy concepts to capture requirements, which are stated in linguistic terms. Fuzzy logic helps to convert the linguistic terms in a quantitative manner.

The i^* frameworks are useful for qualitatively representing and analysing how the stakeholders' goals influence each other (Horkoff and Yu, 2009). However, it lacks any method of quantitative support for goal analysis. In the next section, a quantitative-based goal analysis approach using inter-actor dependencies for the i^* framework is described.

3.3 The Quantitative, Fuzzy based Goal Analysis using Inter-Actor Dependency

The i^* framework involves an enormous number of actors performing distinct functions, interacting with each other to accomplish individual and common goals. To perform common goals, tasks are distributed among the actors to work in parallel. The inter-actor dependencies characterise this arrangement. Goal analysis has to be performed by considering these dependencies amongst the actors. To illustrate the approach, we use the goal model shown in Figure 3.4.

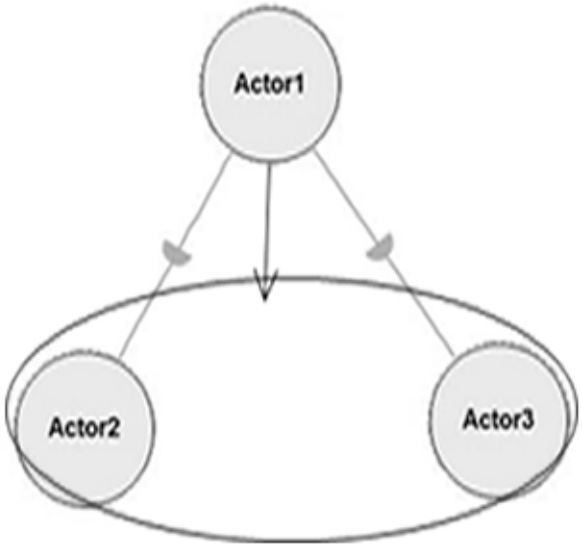

Figure 3.4: Inter-actor Dependency

In this example, Actor1 depends on Actor2 and Actor3 for its goal accomplishment. The goal analysis for Actor1 results in the following three cases:
Case 1: Goals analysis using the dependency from Actor2 only
Case 2: Goals analysis using the dependency from Actor3 only
Case 3: Goals analysis using the dependencies from Actor2 and Actor3 simultaneously.

The results obtained from these three cases can be analysed to find the impact of one or more actors on another actor in terms of goal accomplishment and satisfaction. For understandability and explanation of the approach, the Youth Counselling case study as shown in figure 3.5 has been used throughout the

following section. The steps i to v are carried out for each alternative to find its satisfaction percentage for the top softgoals. The top softgoals satisfaction percentages for each alternative are compared in order to select an alternative that provides the best satisfaction.

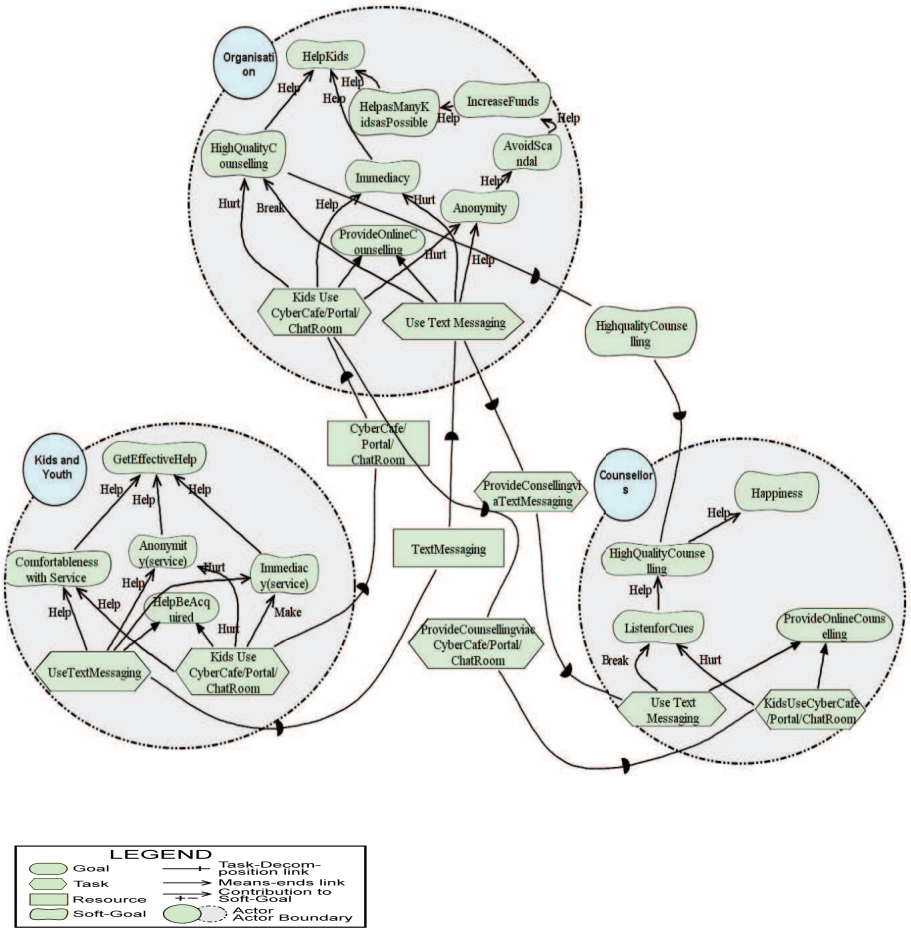

Figure 3.5: SR Model: Youth Counseling Example (adapted from Horkoff and Yu (2009))

i) **Assigning weights to leaf softgoals:** The leaf softgoals (the softgoals that are lower in the hierarchy are called leaf softgoals) are assigned values from 0 to 100 based on their relative importance in percentage, and the weight of a leaf softgoal is represented by ω_L.

Youth Counselling : The leaf softgoals (LSG) *HighQualityCounseling*, *Immediacy* and *Anonymity* of the actor *Organisation* are assigned weights 50%, 70%, and 30% respectively based on their relative importance.

ii) **Fuzzy weights for the correlation between goals and softgoals:** The contributions of goals or tasks to softgoals described by *make*, *help*,*some+*, *some-hurt*, and *break* are expressed as fuzzy numbers. Both triangular and trapezoidal fuzzy numbers are simple to implement and fast for computation. Triangular fuzzy numbers are used in the proposed approach because starting with a triangular membership function is the simplest approach. Moreover, triangular fuzzy numbers represent fuzzy numbers, whereas trapezoidal fuzzy numbers represent fuzzy intervals. Table 3.1 shows our representation of the contributions of goals/tasks to leaf softgoals (LSG). The fuzzy values and its membership function for the softgoal contribution are shown in Figure 3.6. The correlation between goal and softgoal is represented as $\bar{C}_{A*L}$, where A is an alternative option that is selected and L is a leaf softgoal.

Youth Counselling : For the actor, *Organisation*, correlations between the alternative option *Use Text Messaging* and the leaf softgoals *HighQualityCounseling*, *Immediacy* and *Anonymity* are *break, hurt* and *help* respectively. Based on Table 3.1, the correlation links are assigned (0, 0, 0), (0,0.16, 0.32) and (0.48, 0.64, 0.8) respectively. Similarly, The correlation links between the task, *Use Cyber Cafe/Portal/Chat Room* and the leaf softgoals *HighQualityCounseling*, *Immediacy* and *Anonymity* are *hurt, help* and *hurt* respectively. Based on Table 3.1, the correlation links are assigned weights (0,0.16, 0.32) (0.48, 0.64, 0.80), and (0, 0.16, 0.32) respectively.

Table 3.1: Fuzzy values for hard goal and softgoal correlation.

Name	Fuzzy Contribution
Make	(0.64, 0.80, 1)
Help	(0.48, 0.64, 0.80)
Some+	(0.32, 0.48, 0.64)
Some-	(0.16, 0.32, 0.48)
Hurt	(0, 0.16, 0.32)
Break	(0, 0, 0.16)

iii) **Calculation of leaf softgoal score:** Let us represent the leaf softgoal score as $\bar{S}_{L(A)}$, it is calculated from the variables: weights of the leaf softgoals and the impact of leaf softgoals for the selected alternative. It also takes into account any dependencies on the other actors. The dependency link is considered as *'MAKE'* contribution. If the dependency score and dependency impact are denoted by $\bar{S}_d$ and $\bar{I}_d$ correspondingly and there are *'n'* dependencies, then the

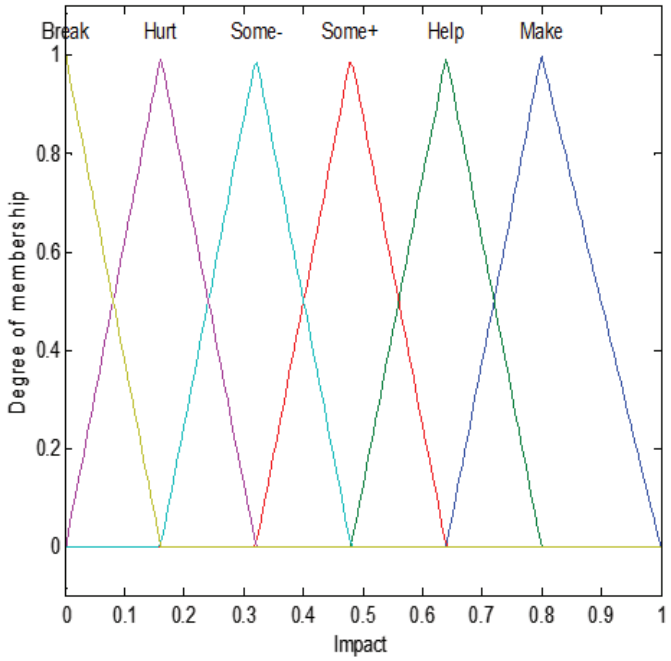

Figure 3.6: Membership function for Impact

equation for the score calculation of a leaf softgoal is given by the equation 3.1 below:

$$\bar{S}_{L(A)} = \bar{C}_{A*L} * \omega_L + \sum_{i=1}^{n} \left(\bar{S}_{di} * \bar{I}_{di}\right) \tag{3.1}$$

Youth Counselling: To illustrate, let us consider the score calculation of the leaf softgoal *HighQualityCounseling* for the alternative *Use CyberCafe/Portal/ ChatRoom* in the actor *Organisation*. The leaf softgoal *HighQualityCounseling* of actor *Organisation* depends on the leaf softgoal *HighQualityCounseling* of actor *Counsellor*. So, in score calculation the leaf softgoal *HighQualityCounseling* of actor *Organisation*, the score of *HighQualityCounseling* of actor *Counsellor* is used. If the calculated score of *HighQualityCounseling* of actor *Counsellor* is (0.0, 0.031, 0.0768), then the score equation for *HighQualityCounseling* of actor *Organisation* is given by:

$$\begin{aligned} \bar{S}_{HighQualityCounseling(CyberCafe/Portal/ChatRoom)} = \\ (0, 0.16, 0.31) * 0.5 + (0.64.0.8, 1) * (0.0, 0.031, 0.0768) \\ = (0, 0.1, 0.23) \end{aligned}$$

The multiplication operation is performed using fuzzy multiplication.

iv) **Propagation of leaf softgoal scores to find satisfaction of the softgoals:** The LSG scores are propagated backwards to find the scores of the softgoals that are higher in the hierarchy. Softgoals are the recipients of multiple contribution links. The score is calculated in two steps. In the first step, the score of its children are multiplied by their impact links. In the second step, the combined effects of all the children are calculated by using the addition operation. Let us represent the softgoal (SG) score by $\bar{S}_{SG}$. If the goal for which the score calculated is dependent on other actor goals, then the dependency score is also taken into consideration for score calculation. The score of a softgoal is given by equation 3.2 below:

$$\bar{S}_{(SG)} = \sum_{i=1}^{n} \left(\bar{C}_{SCi} * \bar{S}_{LCi|SCi}\right) + \sum_{i=1}^{m} \left(\bar{S}_{di} * \bar{I}_{di}\right) \tag{3.2}$$

where LC represents leaf softgoal child and SC represents softgoal child, $\bar{C}_{SCi}$ is the correlation link between a softgoal and its i^{th} child, $\bar{S}_{LCi|SCi}$ is the score of its i^{th} child, $\bar{S}_{di}$ is the score of its i^{th} dependent, $\bar{I}_{di}$ is the i^{th} dependent impact, 'n' is the number of its children and 'm' is the number of dependencies.

Youth Counselling: Let us consider the score calculation of the softgoal *HelpKids* for the alternative option *Use CyberCafe/Portal/ChatRoom* in the actor *Organisation*. The softgoal *HelpKids* has three children namely *HighQuality-Counseling*, *Immediacy*, and *HelpasManyKidsasPossible*. The softgoal *HelpKids* is not dependent on any other actor goals. Hence, the scores calculation involves scores of its children only. If the scores of the softgoals *HighQualityCounseling*, *Immediacy*, and *HelpasManyKidsasPossible* are (0, 0.1, 0.23), (0.336,0.448,0.56) and (0,0.012,0.049) respectively, then the score equation of *HelpKids* is given by

$$\begin{aligned}\bar{S}_{HelpKids(UseCyberCafe/Portal/ChatRoom)} &= (0, 0.1, 0.23) * (0.48, 0.64, 0.8) \\ +(0.336, 0.448, 0.56) * (0.48, 0.64, 0.8) &+ (0, 0.012, 0.049) * (0.48, 0.64, 0.8) \\ &= (0.16, 0.358, 0.67)\end{aligned}$$

v) **Selection of alternative with the highest score:** The scores are propagated backwards until the softgoals that are the top of the hierarchy are reached. The softgoals that are at the top of the hierarchy are called top softgoals. These top softgoal scores for each alternative are then compared and the best alternative is selected for implementation, thereby supporting the analyst in the decision-making process. This is done from each actor's perspective. To obtain

a quantifiable result, the scores are defuzzified by applying the α - cut operation and using an optimal index λ (Deng and Molla, 2008). The λ indicates the degree of confidence and it can take values $\lambda = 0$ for a pessimistic index, $\lambda = 0.5$ for a moderate index and $\lambda = 1$ for an optimistic index.

Youth Counselling : In the actor *Organisation*, the defuzzified score of the top softgoal *HelpKids* for the alternative option *UseCyberCafe/Portal/ChatRoom* is 87% and for the alternative option *UseTextMessaging* it is 29%. On comparison of these two values, the alternative option *UseCyberCafe/Portal/ChatRoom* has a higher value than the alternative option, *UseTextMessaging*. The option *UseCyberCafe/Portal/ChatRoom* provides the best satisfaction level and hence is selected for the actor *Organisation*.

3.4 Simulation and Evaluation of the Approach with Case Studies

The validation of the fuzzy-based quantitative approach is important as it plays a major role in critical decision making. The validation of the approach is done based on its ability to help in the decision-making process. To facilitate this, a simulation for fuzzy based inter-actor goal analysis was developed in Visual C++. A Strategic Rationale (SR) model was considered as a collection of directed graphs with each graph corresponding to an actor. Directed graphs represent the softgoal interdependencies and the inter-actor dependencies. The directed graph was implemented using linked list representation. Inputs to the goal analysis were lists with each list representing each actor's softgoal hierarchy and the alternative option for which softgoals satisfaction is calculated. The output of the simulation is each actor's top softgoals satisfaction for the given alternative option. The pseudo code for inter-actor goal analysis of the i^* framework is given in **Algorithm 1**.

We evaluated the fuzzy-based inter-actor goal analysis using two distinct case studies from the existing RE literature : Youth Counselling (Horkoff and Yu, 2009) and Meeting Scheduler (Letier and Van Lamsweerde, 2004). The computer-based Youth Counselling (Figure 3.5) provides a friendly, confidential service for young people who are in need of counselling. It supports phone counselling for youth but is primarily concerned with reaching more youth via the Internet. In the Youth Counselling example, the two different alternative tasks for all the three actors are

- Kids Use CyberCafe/Portal/ChatRoom

Algorithm 1 Pseudo code for Goal Analysis using inter-actor dependencies in i^* Framework

INPUT :
i) Set of interconnected graphs representing the softgoals interdependencies and actor dependencies.
ii) A task/goal of each actor and their impacts with the leaf softgoals.

OUTPUT :
The top softgoals satisfaction percentage for each actor.

// **compute leaf softgoals scores**
for all graph in the given set of graphs **do**
The leaf softgoals are assigned weights to reflect their relative importance
Compute the leaf softgoals score by multiplying its weight with impact of the given goal.
end for
// **compute softgoals score in backward propagation**
while a graph with score not calculated **do**
if graph is independent and scores are not calculated **then**
while top softgoal not reached **do**
Compute the softgoals score by adding all its children multiplied score with its impact
end while
else
// **depends on other graphs**
if leaf softgoal depends on other actors **then**
for each graph it depends add the depended score to this leaf softgoal score
end if
while top softgoal not reached **do**
Compute the softgoals score by adding all its children multiplied score with its impact
end while
end if
end while

- Kids Use TextMessaging

In this case study, an analyst has to select an alternative that achieves good satisfaction for softgoals *GetEffectiveHelp*, *Happiness* and *HelpKids* of actors *Kids and Youth*, *Counsellor* and *Organisation* respectively.

By using the first alternative *Kids Use CyberCafe/Portal/ChatRoom*, goal analysis was performed using steps 'i' through 'v' presented in Section 3.2; the computed scores of the softgoals are shown in Figure 3.7. In this case study, by considering only the softgoal dependencies, it is apparent that the existence of the Case 1: actor *Organisation* depends on actor *Counsellor*.

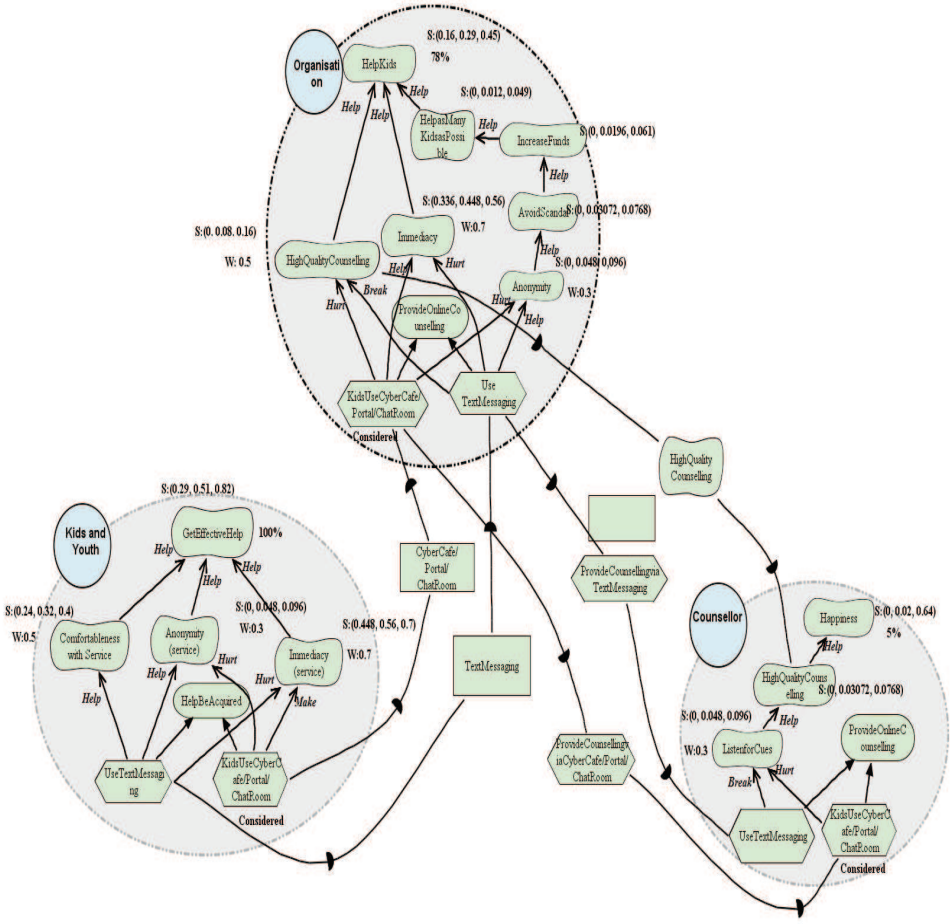

Figure 3.7: Quantitative Analysis of Goals for alternative *Use CyberCafe/Portal/ChatRoom* in Kids Youth Counseling case study

As seen from Figure 3.7, the first alternative option, *Kids Use CyberCafe/Portal/ChatRoom*, was estimated to achieve satisfaction scores of 100%, 5% and 78% for the top softgoals *GetEffectiveHelp (Kids and Youth)*, *Happiness (Counsellor)* and *HelpKids*

(Organisation) respectively. To analyse the estimated values, these values are compared with the satisfaction values from the second alternative element: *Kids UseTextMessaging*. The scores of softgoals for both the alternative options are given in table 3.2, and table 3.3. From table 3.3, we can see that the satisfaction percentage of *GetEffectiveHelp (Kids and Youth)*, *Happiness (Counsellor)* and *HelpKids (Organisation)* are 83%, 1% and 29% respectively for the second alternative option *Use TextMessaging*. By comparing the scores of the top softgoals for both alternatives, it can be seen that the first alternative, *Kids Use CyberCafe/Portal/ChatRoom*, outperforms the *Kids UseTextMessaging* in terms of the relative weights assigned to each softgoal.

Table 3.2: Leaf softgoal scores for all three actors of Kids Youth Counseling case study

Actor	LSG	Score	
		UseTextMessaging	Use CyberCafe/Portal/ChatRoom
Kids and Youth	Comfortableness	(0.24, 0.32, 0.4)	(0.24, 0.32, 0.4)
	Anonymity	(0.144, 0.192, 0.24)	(0, 0.048, 0.096)
	Immediacy	(0, 0.112, 0.224)	(0.448, 0.56, 0.7)
Counsellors	ListenforCues	(0, 0, 0.048)	(0, 0.048, 0.096)
Organization	HighQualityCounselling	(0, 0, 0.08)	(0, 0.08, 0.16)
	Immediacy	(0, 0.112, 0.224)	(0.336, 0.448, 0.56)
	Anonymity	(0.144, 0.192, 0.24)	(0, 0.048, 0.096)

Table 3.3: Top Softgoals Satisfaction scores for Kids Youth Counseling (* indicates goal selection)

Actor	Top Softgoals	Alternative option scores		Defuzzified scores	
		UseTextMessaging	Use CyberCafe/ Portal/ChatRoom	UseTextMessaging	Use CyberCafe/ Portal/ChatRoom
Kids and Youth	Get Effective Help	(0.18,0.39,0.69)	(0.29,0.51,0.82)*	0.83(83%)	1 (100%)
Counsellors	Happiness	(0,0,0.03)	(0,0.02,0.64)*	0.01(1%)	0.052(5%)
Organization	Help Kids	(0.007,0.10,0.37)	(0.16 ,0.29,0.45)*	0.29(29%)	0.78(78%)

To check the feasibility of our approach, we have demonstrated the goal analysis procedure with another case study. The second case study is the Meeting Scheduling System (Figure 3.8). A computer-based Meeting Scheduling System should effectively organise meetings by finding appropriate dates and locations for invited participants. All potential information about the participants is obtained by the meeting initiator. The intended participants may express their requested constraints by email or the requested information may be obtained by accessing their electronic agenda.

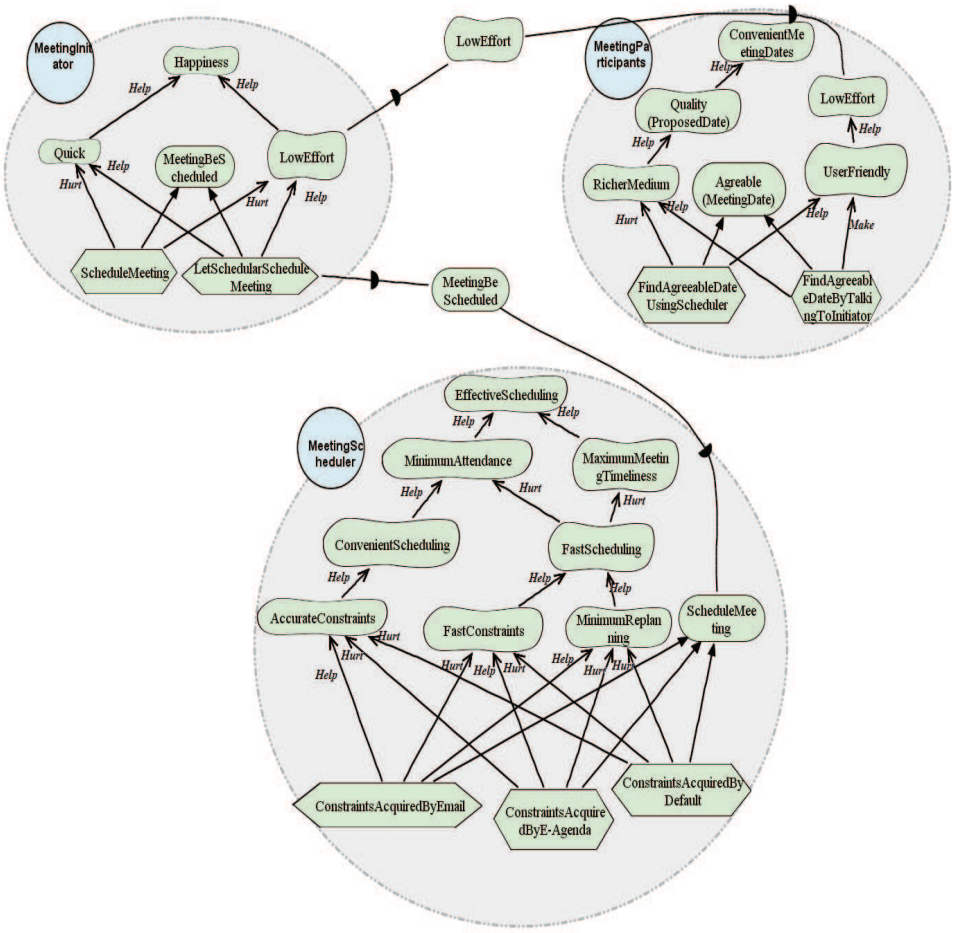

Figure 3.8: Meeting Scheduling System case study

This example is different from the previous case study. In the Kids Youth Counseling case, all the actors have the same type of alternatives and the same number of alternatives. However, in the Meeting Scheduling system, each actor has a different number and different types of alternatives. The selected alternative options are different for each actor and goal analysis is performed in accordance with each alternative point of view.

In the Meeting Scheduling system, the softgoal *LowEffort* of the actor *MeetingInitiator* has a softgoal dependency on *LowEffort* of the actor *MeetingParticipants*. Hence, the goal analysis for the actor *MeetingParticipants* is carried out first so that its score values can be used in the goal analysis of the actor *MeetingInitiator*. By using the goal analysis procedure described in Section 3.3, the scores of the softgoals of the actor *MeetingParticipants* are computed. Table 3.4 shows the scores of the top softgoals of the actor *MeetingParticipants*. From Table 3.4,

it can be seen that the alternative *FindAgreeableDateByTalkingToInitiator* of the actor *MeetingParticipants* contributes 29% and 85% to the top softgoals *ConvenientMeetingDates* and *LowEffort* respectively. Similarly, the alternative *FindAgreeableDateUsingScheduler* of the actor *MeetingParticipants* contributes 8% and 68% to the top softgoals *ConvenientMeetingDates* and *LowEffort* respectively. The alternative option *FindAgreeableDateByTalkingToInitiator* has better satisfaction than the other alternative option *FindAgreeableDateUsingScheduler* and hence it is selected for the actor *MeetingParticipants*.

Table 3.4: Top Softgoals Satisfaction scores for the actor MeetingParticipants (* indicates goal selection)

Top Softgoals	Alternative option scores		Defuzzified scores	
	FindAgreeableDate UsingScheduler	FindAgreeableDate ByTalkingToInitiator	FindAgreeableDate UsingScheduler	FindAgreeableDate ByTalkingToInitiator
ConvenientMeetingDates	(0, 0.0328, 0.1024)	(0.055, 0.13, 0.256)	0.08 (8%)	0.286 (29%)*
LowEffort	(0.184, 0.327, 0.512)	(0.246, 0.409, 0.64)	0.675 (68%)	0.852 (85%)*

Next, the scores of the actor *MeetingInitiator* are calculated. The scores of the top softgoals of the actor *MeetingInitiator* are shown in Table 3.5. From the Table 3.5, it can be seen that the alternative *ScheduleMeeting* of the actor *MeetingInitiator* contributes 76% to the top softgoal, *Happiness*. Similarly, the alternative *LetSchedulerScheduleMeeting* of the actor *MeetingInitiator* contributes 100% to the top softgoal *Happiness*. The alternative option *LetSchedulerScheduleMeeting* has better satisfaction than the other alternative option *ScheduleMeeting*; hence, it is selected for the actor *MeetingInitiator*.

Table 3.5: Top Softgoals Satisfaction scores for the actor MeetingInitiator (* indicates goal selection)

Top Softgoals	Alternative option scores		Defuzzified scores	
	ScheduleMeeting	LetSchedulerSchedule Meeting	ScheduleMeeting	LetSchedulerSchedule Meeting
Happiness	(0.088, 0.342,0.742)	(0.314, 0.611, 1)	0.758 (76%)	1 (100%)*

As the actor *MeetingScheduler* does not depend on any other actor, the goal analysis is performed from the perspective of each alternative. The scores of the top softgoals of the actor *MeetingScheduler* are shown in Table 3.6. From Table 3.6, it can be seen that the alternative *ConstraintsAcquiredByEmail* of the actor *MeetingScheduler* contributes 36% to the top softgoal *EffectiveScheduling*. Simi-

larly, the alternative *ConstraintsAcquiredByE-Agenda* of the actor *MeetingScheduler* contributes 22% to the top softgoal *EffectiveScheduling*. And the alternative *ConstraintsAcquiredByDefault* of the actor *MeetingScheduler* contributes 14% to the top softgoal *EffectiveScheduling*. The alternative option *ConstraintsAcquiredByEmail* has better satisfaction than the other two alternative options *ConstraintsAcquiredByE-Agenda* and *ConstraintsAcquiredByDefault*. Hence, *ConstraintsAcquiredByEmail* is selected for the actor *MeetingScheduler*.

The graphical representation of the comparison of scores for Kids Youth Counseling is shown in Figure 3.9 and for the Meeting Scheduler System in Figure 3.10.

Table 3.6: Top Softgoals Satisfaction scores for the actor MeetingScheduler (* indicates goal selection)

Top Softgoals	Alternative option scores			Defuzzified scores		
	ConstraintsAcquired ByEmail	ConstraintsAcquired ByE-Agenda	ConstraintsAcquired ByDefault	ConstraintsAcquired ByEmail	ConstraintsAcquired ByE-Agenda	ConstraintsAcquired ByDefault
EffectiveScheduling	(0.032, 0.136, 0.409)	(0, 0.074, 0.3014)	(0, 0.0419, 0.2032)	0.357 (36%)*	0.224 (22%)	0.143 (14%)

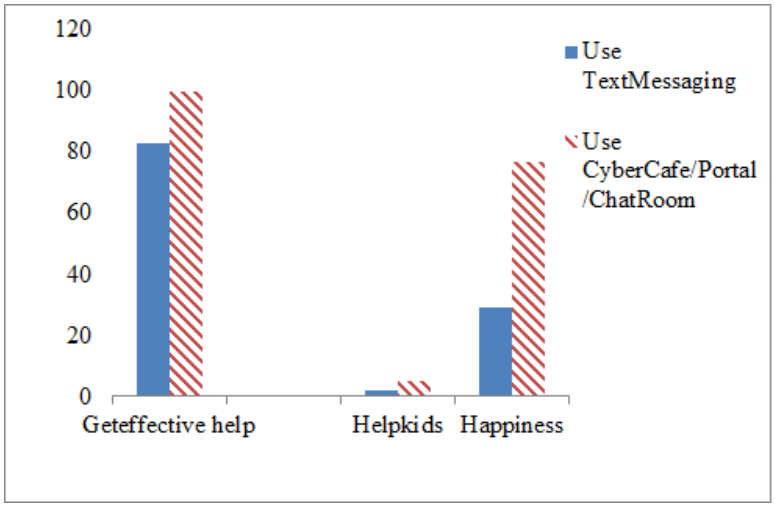

Figure 3.9: Comparison Graph for Kids Youth Counseling case study

To check the effectiveness of the proposed approach, the estimated values from inter-actor dependencies were compared with values obtained from without using inter-actor dependencies. Tables 3.7 and 3.8 demonstrate the comparison of scores for the Kids Youth Counseling and Meeting Scheduling System case studies. In the first case study, Kids Youth Counseling, the only softgoal that is affected by interaction is *HighQualityCounseling* of the actor *Organisation*. From table 3.7, it can be seen that by performing goal analysis without using inter-actor dependencies, the satisfaction of *Kids Use CyberCafe/Portal/ChatRoom*

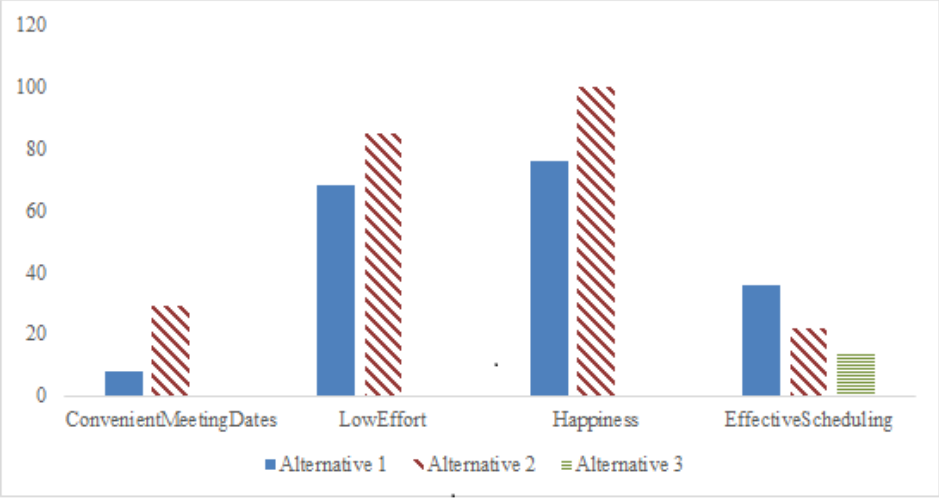

Figure 3.10: Comparison Graph for Meeting Scheduling System case study

and *Kids Use TextMessaging* are 73% and 26% respectively. With inter-actor dependencies, the scores for *Kids Use CyberCafe/Portal/ChatRoom* and *Kids Use TextMessaging* are found to be 77% and 29% respectively. By analysing the above computed scores, it can be seen that, the proposed inter-actor dependency approach gives improved scores for softgoals than for those without inter-actor dependencies. Similarly, from table 3.8 it can be observed that for the Meeting Scheduling System (MSS), the scores obtained by the inter-actor dependency approach are better than those for the without inter-actor dependencies.

For Youth Counseling Service with single softgoal dependency *HighQuality-Counseling* between the actors *Organisation* and *Counsellor*, the softgoal *Help-Kids* satisfaction is found to be increased by 4% for *Kids Use CyberCafe/Portal/ ChatRoom* and 3% for *Kids Use TextMessaging*. Similarly, for the Meeting Scheduling System with single softgoal dependency *LowEffort* between the actors *MeetingInitiator* and *MeetingParticipants*, the softgoal *Happiness* is found to be increased by 9% for *LetSchedulerScheduleMeeting*. By considering single softgoal dependency, the proposed approach provides an improved score over the non-inter-actor dependencies. However, when there are a large number of dependencies, it is expected that the proposed approach will give a significantly better result comparatively and thus help in decision making.

Table 3.7: Scores Comparison for Kids Youth Counseling (with and without inter-actor dependencies)

Actor	Top Softgoals	Use CyberCafe/Portal/ ChatRoom		UseTextMessaging	
		With Inter-actor Dependencies	Without Inter-actor Dependencies	With Inter-actor Dependencies	Without Inter-actor Dependencies
Organization	Help Kids	77%	73%	29%	26%

Table 3.8: Scores Comparison for Meeting Scheduling System (with and without inter-actor dependencies)

Actor	Top Softgoals	ScheduleMeeting		LetSchedulerScheduleMeeting	
		With Inter-actor Dependencies	Without Inter-actor Dependencies	With Inter-actor Dependencies	Without Inter-actor Dependencies
MeetingInitiator	Happiness	100%	100%	76%	67%

3.5 Chapter Summary

This chapter proposed a fuzzy-based approach for goal analysis using inter-actor dependency in the i^* framework. The proposed approach was implemented and tested using two case studies from the existing literature: Youth Counsellor and Meeting Scheduling System. Analysis was based on improvements to the top softgoal scores of each actor in the goal model. The proposed approach of inter-actor dependencies gives an improved result over non-inter-actor dependencies.

In the next two chapters, we describe a method for including optimisation in goal models to select the weights for the LSG and thereby to select the alternative options for which the softgoals' levels of satisfaction are best.

Chapter 4

Optimal Reasoning and Sensitivity Analysis in the i^* Framework

In the previous chapter, we explained the idea of enhancing the i^* framework to support quantitative reasoning, including the inter-actor dependencies. We also presented the fuzzy-based quantitative approach to the i^* framework along with evaluation of the approach using two case studies from the existing literature. Although quantitative reasoning is being increasingly accepted, optimisation models have attracted significantly less consideration. In this chapter, we discuss the need for optimisation techniques in goal analysis of the i^* framework. We also present an optimisation model for the i^* framework; this optimisation model is validated by simulation-based analysis using case studies.

Another interesting point of optimisation is sensitivity analysis. Sensitivity analysis is employed to detect the system's behaviour when input data changes. Furthermore, this chapter examines the application of sensitivity analysis to the i^* framework in order to provide the analyst with extra information about the quantitative values selected.

The chapter is structured as follows: in Section 4.1, we discuss the need for optimisation in the i^* framework; Section 4.2 introduces Multi-Objective Optimisation; Section 4.3 presents the proposed optimal i^* framework; Section 4.4 explains application and evaluation of optimisation model to i^* case studies; Section 4.5 presents sensitivity analysis of the i^* framework.

Some of the material in this chapter has previously appeared in the following publications:

1. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2015). Optimal Reasoning of Goals in the i^* Framework. In Software Engineering Conference (APSEC), 2015 Asia-Pacific (pp. 346-353). IEEE.

2. Chitra M. Subramanian, Aneesh Krishna, Arshinder Kaur (2016) Sensitivity Analysis of the i^* Optimisation Model. JSW 11(1): 10-26.

4.1 Need for Optimisation in Quantitative Analysis of the i^* Framework

In practice, the input data required for goal evaluation are incomplete, or unobtainable, or imprecise. Moreover, when presented with real-life requirements engineering (RE) problems, one usually has to deal with multiple goals, each of which may be significantly important to address in relation to the RE problem presented. The priority of these goals may be different but, at the same time, it is important that they are considered. These goals can be either conflicting or congruent. There is a need to address such issues with a method which considers multi-objective optimisation. Furthermore, scalability is another issue associated with requirements evaluation. It is difficult to assign values to the goals of a goal model in a large and complex system. Hence, the decision making becomes a crucial task (Letier and Van Lamsweerde, 2004; Yu, 2011; Heaven and Letier, 2011; Liaskos et al., 2010).

In this research, optimisation has been used to find the values (weights) of leaf softgoals. These values (weights) are in turn used in the goal analysis procedure explained in the previous chapter to select the best alternative option. Optimisation, an operation research technique, is a method used to obtain the best possible solution under the given circumstances (Johnson and Montgomery, 1974; Sasieni et al., 1959). We explain the need for optimisation in goal analysis with a running example namely London Ambulance Service [LAS], which is adapted from (You, 2004). The partial Strategic Rationale (SR) model of LAS is shown in Figure 4.1. The LAS is a computer-aided system used to automate the dispatch of an ambulance to an emergency scene with an arrangement that the ambulance be dispatched in 3 seconds and arrive at the scene in 11 seconds. However, the system failed to address the requisite time specification and was also unreliable, and crashed. The LAS was used as a standard case study for goal modelling by the RE research community (You, 2004). The partial SR diagram in Figure 4.1 depicts four actors, namely the *Ambulance Crew*, the *Resource Allocator*, the *Resource Allocator Module* and the *Human Resource Allocator*, as well as some of their intentional relationships. The actor *Resource Allocator* has the goal *Becollected[IncInfo]* which represents the incident information that is to be collected. This goal can be accomplished in two ways using either paper-based information or network-based information. Hence, the goal *Becollected[IncInfo]* is decomposed into two tasks known as the *ByPaperbased Form* and the *By-*

Database or Network. The goal *Becollected[IncInfo]* represents a decision point. The goal *BeGenerated[MobileInst]* of actor *Resource Allocator* represents which vehicle is to be assigned to the incident scene. This information can be generated either by a computer-based algorithm or by a human. Therefore, the goal *BeGenerated[MobileInst]* is again *OR* decomposed into two tasks, namely the *ByMachineBasedAlgorithm* and the *ByHumanDecision* and, hence, the goal *BeGenerated[MobileInst]* becomes a decision point. The selection of a task for these goals *Becollected[IncInfo]* and *BeGenerated[MobileInst]* influences the satisfaction levels of non-functional goals or softgoals, namely *Timeliness[mobilization]* and *Optimal[mobInst]* of the actor *Resource Allocator*.

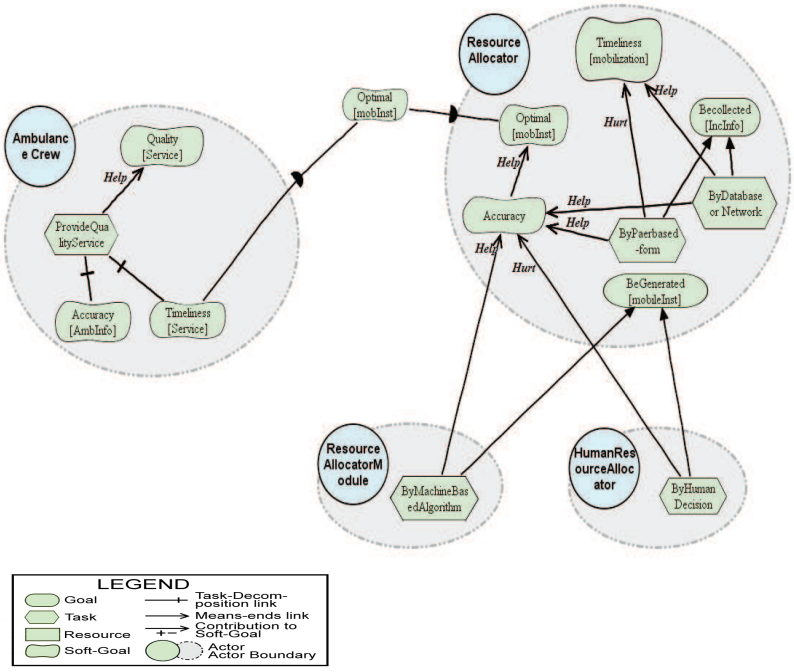

Figure 4.1: SR diagram for LAS (adapted from You (2004))

There are inter-actor dependencies among these four actors. These inter-actor dependencies demonstrate that an actor depends on another actor for its goal accomplishment. The actor *Ambulance Crew* depends on the actor *Resource Allocator* through the softgoal dependency namely *Optimal[mobInst]*. The softgoal *Timeliness[mobilisation]* of the actor *Ambulance Crew* depends upon the accuracy of the optimal information that has been collected. Furthermore, the goal *BeGenerated[MobileInst]* of the actor *Resource Allocator* depends upon the task *ByMachineBasedAlgorithm* of the actor *ResourceAllocatorModule*. And also

depends upon the task *ByHumanDecision* of the actor *HumanResourceAllocators*. These inter-actor dependencies also influence the decision making of each actor.

In Figure 4.1 the actor *Resource Allocator* has two decision points, namely the goals *Becollected[IncInfo]* and *BeGenerated[MobileInst]*. These two goals are *OR* decomposed into two tasks. The intent of a requirement analyst is to select an alternative task from these two tasks for each goal such that it delivers a maximum satisfaction percentage to the top softgoals. The selection of a task depends on its contribution to the non-functional requirements (represented by softgoals) of the goal model. To illustrate, in this case study, the LAS the analyst has to make a selection from the alternative tasks *Bypaperbasedform* or *ByDatabase or Network* in order to maximise the satisfaction level of the top softgoals known as *Timeliness[mobilization]*, *optimal[mobInst]* of the actor *Resource Allocator*. The goal *Timeliness[service]* of the actor *Ambulance Crew* depends upon the *Optimal[MobInst]* of the actor *Resource Allocator*. The selection of an alternative option for the actor *Resource Allocator* affects the softgoals of the actor, *Ambulance Crew*.

In explaining the inter-actor quantitative analysis (approach described in previous chapter), we refine the leaf softgoal score equation and softgoal score equation defined in the previous chapter and provide generalised equations. To begin the analysis, let us assume that an analyst assigns the weights 70%, 60%, 70% and 50% to the leaf softgoals (LSG) *Accuracy[AmbInfo]*, *Timeliness[service]*, *Accuracy* and *Timeliness[mobilization]* respectively. Let the weight of an i^{th} leaf softgoal be represented by ω_{L_i}. The goal *Becollected[IncInfo]* of the actor *Resource Allocator* has two tasks, namely *ByPaperbased-form* and *ByDatabase or Network*. The analyst selects the first option known as *ByPaperbased-form* and performs the goal analysis to find the impact of the selection of this option. The contribution (or impact) of this alternative to the leaf softgoals is in the form of triangular fuzzy numbers and is given as $\bar{C}_{A_j * L_i}$ where A_j is j^{th} alternative option that is selected and L_i is i^{th} leaf softgoal. The contribution of the alternative *ByPaperbased-form* on the leaf softgoals *Accuracy[AmbInfo]*, *Timeliness[service]*, *Accuracy* and *Timeliness[mobilization]* are (0.48, 0.64, 0.80), (0.48, 0.64, 0.80) , (0, 0.16, 0.32) and (0, 0.16, 0.32) accordingly. An LSG score is calculated using its weight and the contribution of the selected alternative option. Let us denote the score of i^{th} leaf softgoal by $\bar{S}_{L_i}$. An actor may depend on another actor for its goal performance. The inter-actor dependencies may influence the decision making regarding the alternative options. The dependency link is considered to be the *MAKE* contribution. If the dependency score and dependency impact are

denoted by $\bar{S}_d$ and $\bar{I}_d$ correspondingly, and if there are *'nd'* dependencies, then the equation for score calculation of an i^{th} leaf softgoal for j^{th} alternative of k^{th} actor at t^{th} level of hierarchy is given by equation 4.1 below:

$$\bar{S}_{Lijkt} = \bar{C}_{Aj*Li} * \omega_{Lik} + \sum_{b=1}^{nd} \left(\bar{S}_{db} * \bar{I}_{db}\right) \tag{4.1}$$

where t is the hierarchy level, and for leaf softgoals, t is zero.

Thus using equation 4.1, the scores calculated for the LSGs *Accuracy[AmbInfo]*, *Timeliness[service]*, *Accuracy* and *Timeliness[mobilization]* are (0.336, 0.448, 0.56), (0.288, 0.455, 0.65), (0, 0.11,0 .224) and (0, 0.08, 0.16) respectively. Next, the LSG scores are propagated backwards in the goal hierarchy until the top softgoals are reached so as to find the scores of the other softgoals in the model. Let the i^{th} softgoal (SG) score be represented by $\bar{S}_{SG_i}$. The score calculation of an i^{th} softgoal for j^{th} alternative of k^{th} actor at t^{th} level in the hierarchy is given by equation 4.2 below:

$$\bar{S}_{SGijkt} = \sum_{d=1}^{nc} \left(\left(\bar{C}_{SGi*(SGd|LSGd)}\right) * \left(\bar{S}_{Ldjk(t-1)} | \bar{S}_{SGdjk(t-1)}\right)\right) + \sum_{b=1}^{nd} \left(\bar{S}_{d'b} * \bar{I}_{d'b}\right) \tag{4.2}$$

where $\bar{C}_{SGi*(SGd|LSGd)}$ is the correlation link between the i^{th} softgoal and its d^{th} child which may be a softgoal or a leaf softgoal, $\bar{S}_{Ldjk(t-1)}$ is the score of its d^{th} leaf softgoal child, $\bar{S}_{SGdjk(t-1)}$ is the score of its d^{th} softgoal child,| represents *or*, $\bar{S}_{d'b}$ is the score of its b^{th} dependent, $\bar{I}_{d'b}$ is the b^{th} dependent impact, *'nc'* is the number of children of i^{th} softgoal and *'nd'* is the number of dependencies. By using equation 4.2, the scores calculated for the top softgoals *Quality[service]* and *Optimal[mobInst]* are (0.29, 0.515, 0.81) and (0, 0.07, 0.1792). These scores are defuzzified so as to obtain a quantifiable value. It shows the degree of satisfaction of the top softgoals for the selected alternative option. The defuzzified scores are 100% and 16% for *Quality[service]* and *Optimal[mobInst]* respectively. Similarly, the analyst has to perform the analysis to find the satisfaction values for the alternative options, known as *ByDatabase or Network*. By performing this analysis, the defuzzified scores of the top softgoals *Quality[service]* and *Optimal[mobInst]*, are 100% and 59% accordingly. By comparing the scores of the top softgoals of the two alternatives, the option *ByDatabase or Network* is found to have better satisfaction for the softgoals. Hence, the analyst decides to select the option *ByDatabase or Network*.

Since the weights are subjective in terms of the analyst, different scores are obtained for the same softgoals based on the weights assigned by the analyst. To illustrate this, let us assume that another analyst assigns the weights 50%, 50%, 60% and 40%, to the leaf softgoals *Accuracy[AmbInfo]*, *Timeliness[service]*, *Accuracy* and *Timeliness[mobilization]* respectively. The calculated scores for the top softgoals *Quality[service]* and *Optimal[mobInst]* are now found to be 84% and 13% for the first alternative option *ByPaperbased-form* and 84% and 50% for the second alternative option *ByDatabase or Network*. In the above analysis, we can see that different scores are calculated for the same alternative with different weights assigned to the leaf softgoals. Hence, the scores of the softgoals are subjective depending upon the subjective selection of weights made by the requirement analyst. In order to avoid these subjective scores, an optimisation model is proposed for finding the weights of the leaf softgoals. To determine the optimal weight, we choose multi-objective optimisation.

In the following section, we give a brief introduction to multi-objective optimisation.

4.2 Multi-Objective Optimisation

Nowadays in all professions, optimisation is used as a technique in decision making. The processes in engineering systems such as design, construction and maintenance include decision making both at managerial and technical levels. The aim of such decision making is to minimise certain parameters or maximise other parameters. Therefore, optimisation is defined as the process of maximising or minimising certain parameters within system considerations. Optimisation is a method of selecting the best or optimal alternatives from a list of possible choices (Harman, 2007). Linear programming, non-linear programming and quadratic programming are some of the optimisation research techniques used. Most of the real-world problems have distinct objectives to be satisfied. Therefore, the single objective optimisation technique is too inadequate to achieve a solution in such circumstances. Hence, techniques for solving problems with multiple-objective optimisation have been developed (Caramia and Dell'Olmo, 2008). In the LAS case study (Figure 4.1), the goal *Becollected[IncInfo]* has two different choices namely *ByPaperbased-form* and *ByDatabase or Network* for the actor *Resource Allocator*. Now the task of the requirement analyst is to select the best or optimal alternative from these two choices. Each choice is considered to be an objective

and hence this problem can be solved by using multi-objective optimisation.

A multi-objective optimisation problem is written mathematically as:

$$\begin{aligned} & Max|Min[f1(x), f2(x), ., fn(x)] \\ & \qquad\qquad\qquad\qquad x\varepsilon Y \end{aligned} \tag{4.3}$$

where *f1, f2,., fn* are scalar functions and Y is the set of constraints. A multi-objective optimisation generates a set of solutions which are called Pareto solutions or a Pareto frontier. The best solution is selected from a Pareto frontier. There are many techniques to solve multi-objective optimisation namely scalarization method, $\in$ - constraints method, goal programming and multi-level programming. To solve the multi-objective optimisation problem, we randomly chose the scalarization method or weighted sum method (Sasieni et al., 1959). In the scalarization method, the multi-objective problem is solved by combining its multiple objectives into one single objective scalar function. Using the scalarization method, the new optimisation problem with single objective function will be

$$\begin{aligned} Max|Min \sum_{i=1}^{n} \gamma_i F_i(\omega_L) \\ \sum_{i=1}^{n} \gamma_i = 1 \\ 0 \leq \omega_L \leq 100 \\ \gamma_i \geq 0, i = 1, 2, ..., n \end{aligned}$$

In the following section we present the optimisation of the i^* framework.

4.3 Optimal *i** Framework

As this research aims to completely automate the goal analysis process, there is a need to minimise the analyst's subjectivity when assigning the weights to leaf softgoals. Assigning weight in the case of a large goal model is difficult, and also preferences for the weights of the leaf softgoals may vary from analyst to analyst; hence, there is a need for a novel method. Therefore, in order to automate and handle the scalability issue, multi-objective optimisation will be applied to goal analysis.

The optimisation model mathematically expresses objective functions in terms of decision variables (Johnson and Montgomery, 1974; Sasieni et al., 1959) using linear programming. The constraints of the objective functions are the restrictions to be satisfied by the decision variables. Once the objective functions, variables and constraints are defined, they are solved by a tool solver to obtain the results. The objective of our optimisation model for an i^* framework is to compute an optimal weight that would achieve the maximum satisfaction of top softgoal for given alternative options, subject to the restriction imposed by the graph structure. The proposed optimisation framework is shown in Figure 4.2. In an SR model with more than one alternative, an objective function in terms of leaf softgoals is obtained for each alternative. These objective functions are solved using an optimiser to give the values of the decision variables in the objective functions. In our case, the decision variables are the weights of the leaf softgoals. The weights obtained by the optimiser are used in the goal analysis procedure to select the best alternative.

Hence, optimisation is performed in order to find the weights of the leaf softgoals and thereby identify an alternative option by which the softgoals satisfaction can be maximized. This minimizes the analyst's interaction and also, by automating the process, it can handle large complex systems.

To frame the optimisation model of an i^* framework, the SR diagram is viewed as the directed graph G (N, A) where N is the set of nodes and A is the set of arcs. The intentional elements of the SR diagram, namely softgoals, goals, and tasks are assumed to be the nodes of the directed graph G and the means-end, task-decomposition and operational contribution links are assumed to be the arcs of the graph G.

An objective function for the optimisation model is formed from the calculations of variables of the graph. For any node that is a leaf softgoal, there is a weight represented by ω_{Lik}, meaning that the weight of the i^{th} leaf softgoal of the k^{th} actor. Additionally, any arc from a goal/task to a leaf softgoal (representing the impact of goal/task) is denoted by a triangular fuzzy number $\bar{C}_{Aj*Li}$, indicating the impact of the j^{th} alternative option on i^{th} leaf softgoal. The i^{th} leaf softgoal score is calculated from the weight of the i^{th} leaf softgoal and its impact for j^{th} alternative and is represented by $\bar{S}_{Lij}$. The score of i^{th} leaf softgoal for j^{th} alternative of k^{th} actor is given by equation 4.1 as :

$$\bar{S}_{Lijk} = \bar{C}_{Aj*Li} * \omega_{Lik} + \sum_{b=1}^{nd} \left(\bar{S}_{db} * \bar{I}_{db}\right)$$

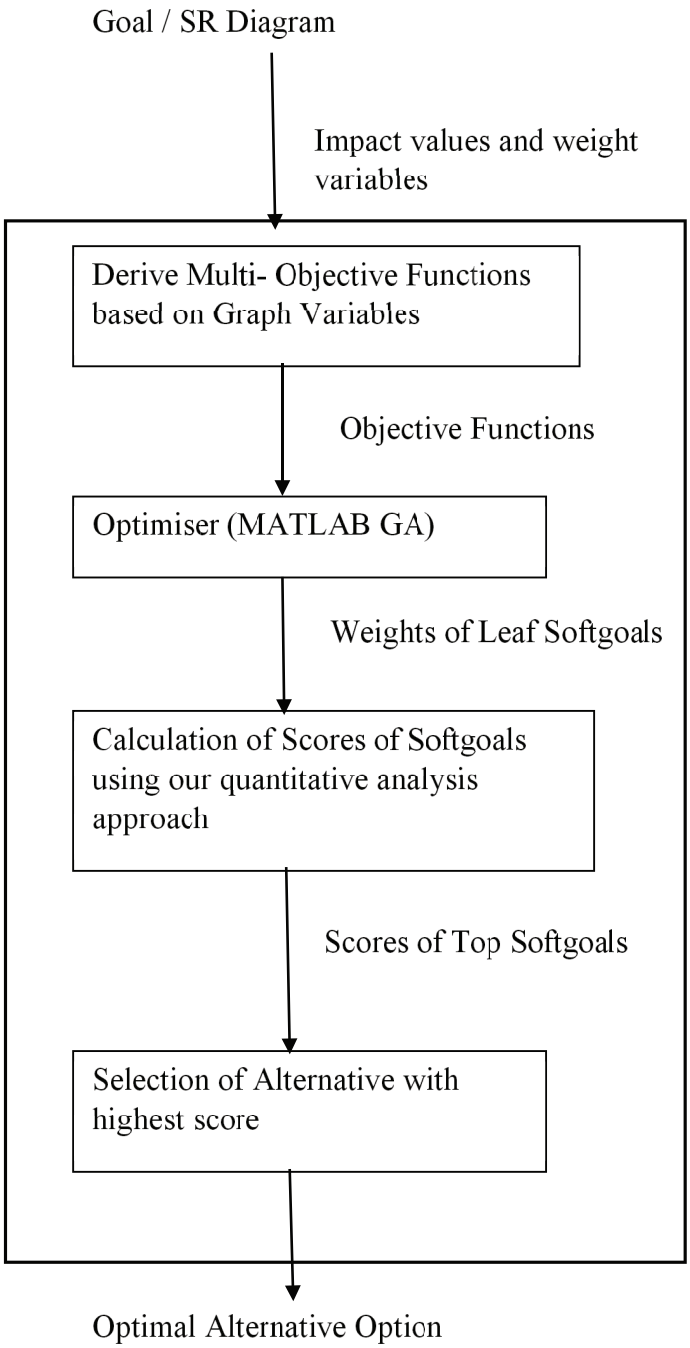

Figure 4.2: Optimisation Framework

Since, the hierarchy level *(t)* for leaf softgoals is zero, we avoid *'t'* in the leaf softgoal score representation. To get maximum satisfaction for any top softgoal, the sum of the leaf softgoal scores for an alternative has to be maximized. Hence the objective function for the first alternative of k^{th} actor with *'m'* leaf softgoals

is represented by:

$$Max(\bar{S}_{L11k} + \bar{S}_{L21k} + ... + \bar{S}_{Lm1k}) \tag{4.4}$$

Using equation 4.1 of leaf softgoal score calculation, equation 4.4 is expanded as below

$$Max\{\bar{C}_{(A1*L1)k} * \omega_{L1k} + \sum_{b=1}^{nd1}\left(\bar{S}_{db} * \bar{I}_{db}\right) + \bar{C}_{(A1*L2)k} * \omega_{L2k} + \sum_{b=1}^{nd2}\left(\bar{S}_{db} * \bar{I}_{db}\right) + ...+ \bar{C}_{(A1*Lm)k} * \omega_{Lmk} + \sum_{b=1}^{ndm}\left(\bar{S}_{db} * \bar{I}_{db}\right)\} \tag{4.5}$$

and equation 4.5 can be rewritten as given below.

$$Max\{\bar{C}_{(A1*L1)k} * \omega_{L1k} + \bar{C}_{(A1*L2)k} * \omega_{L2k} + ... + \bar{C}_{(A1*Lm)k} * \omega_{Lmk}+ \sum_{b=1}^{nd1}\left(\bar{S}_{db} * \bar{I}_{db}\right) + \sum_{b=1}^{nd2}\left(\bar{S}_{db} * \bar{I}_{db}\right) + ... + \sum_{b=1}^{ndm}\left(\bar{S}_{db} * \bar{I}_{db}\right)\}$$

$$Max\{\sum_{i=1}^{m}\bar{C}_{(A1*Li)k} * \omega_{Lik} + \sum_{i=1}^{m}\sum_{b=1}^{ndi}\left(\bar{S}_{db} * \bar{I}_{db}\right)\} \tag{4.6}$$

To avoid complexity in solving the objective function, we omit the inter-actor dependency score part (as its score is taken into consideration during goal analysis). Therefore, the objective function for an i^* framework is now given by equation 4.7 as below:

$$Max\{\sum_{i=1}^{m}\bar{C}_{(A1*Li)k} * \omega_{Lik}\} \tag{4.7}$$

Since, the weights are assumed to be from 0 to 100%, it forms one of the constraint of the objective function. Also the factor $\bar{C}_{(A1*Li)k} * \omega_{Lik}$ should be greater than zero, and it forms the second constraint of the objective equation. Hence, the objective function is written as

$$Max\{\sum_{i=1}^{m}\bar{C}_{(A1*Li)k} * \omega_{Lik}\} \tag{4.8}$$

Subject to

$$\omega_{L11}, \omega_{L21}, ..., \omega_{Lm1} \geq 0$$
$$\omega_{L11}, \omega_{L21}, ..., \omega_{Lm1} \leq 100$$
$$0 \leq \bar{C}_{(Aj*Li)k} * \omega_{Lik} \text{ for } i = 1 \text{ to } m$$

Supposing there are *'n'* alternatives for a given k^{th} actor, then there are *'n'* objective functions which are given by:

$$\begin{aligned} F1(\omega_L) &= Max \sum_{i=1}^{m} \bar{C}_{(A1*Li)k} * \omega_{Lik} \\ F2(\omega_L) &= Max \sum_{i=1}^{m} \bar{C}_{(A2*Li)k} * \omega_{Lik} \\ &..... \\ &..... \\ Fn(\omega_L) &= Max \sum_{i=1}^{m} \bar{C}_{(An*Li)k} * \omega_{Lik} \end{aligned} \tag{4.9}$$

Subject to

$$0 \leq \omega_{Lik} \leq 100 \text{ for } i = 1 \text{ to } m$$
$$0 \leq \bar{C}_{(Aj*Li)k} * \omega_{Lik} \text{ for } i = 1 \text{ to } m$$

Similarly, the objective functions are obtained for each actor in the SR model. In the case of a goal model in which the alternatives are the same for all actors, then a cumulative objective function involving all the actors can be used. Supposing a goal model has *'p'* number of actors, the objective function for a j^{th} alternative option in the goal model is given by:

$$Fj(\omega_L) = Max \left(\sum_{i=1}^{m} \bar{C}_{(Aj*Li)1} * \omega_{Li1} + \sum_{i=1}^{m} \bar{C}_{(Aj*Li)2} * \omega_{Li2} + ... + \sum_{i=1}^{m} \bar{C}_{(Aj*Li)p} * \omega_{Lip} \right) \tag{4.10}$$

In short the function is given by:

$$Fj(\omega_L = Max \sum_{k=1}^{p} \sum_{i=1}^{m} \bar{C}_{(Aj*Li)k} * \omega_{Lik} \tag{4.11}$$

Therefore, the objective functions for a goal model with *'n'* number of alternatives are given by:

$$
\begin{aligned}
F1(\omega_L) &= Max \sum_{k=1}^{p} \sum_{i=1}^{m} \bar{C}_{(A1*Li)k} * \omega_{Lik} \\
F2(\omega_L) &= Max \sum_{k=1}^{p} \sum_{i=1}^{m} \bar{C}_{(A2*Li)k} * \omega_{Lik} \\
&\ldots. \\
&\ldots. \\
Fn(\omega_L) &= Max \sum_{k=1}^{p} \sum_{i=1}^{m} \bar{C}_{(An*Li)k} * \omega_{Lik}
\end{aligned}
\tag{4.12}
$$

Subject to

$$
\begin{aligned}
0 \leq \omega_{Lik} \leq 100 \text{ for } i = 1 \text{ to } m \text{ and } k = 1 \text{ to } p \\
0 \leq \bar{C}_{(Aj*Li)k} * \omega_{Lik} \text{ for } i = 1 \text{ to } m, j = 1 \text{ to } n \text{ and } k = 1 \text{ to } p
\end{aligned}
$$

In general, the multi-objective functions are given by following equation:

$$
\begin{aligned}
Max[F1(\omega_L), F2(\omega_L), ., Fn(\omega_L)] \\
\text{with } \omega_L \varepsilon Y
\end{aligned}
\tag{4.13}
$$

where $n \geq 1$ and Y is the set of defined constraints.
These objective functions can be solved by using the scalarization or the weighted sum technique (Sasieni et al., 1959). The new optimisation problem with unique objective function in the scalarization method is given by:

$$
\begin{aligned}
Max \sum_{i=1}^{n} \gamma_i F_i(\omega_L) \\
\sum_{i=1}^{n} \gamma_i = 1 \\
0 \leq \omega_L \leq 100 \\
\gamma_i \geq 0, i = 1, 2, ..., n
\end{aligned}
\tag{4.14}
$$

where γ denotes the weight associated with each objective function.

In the following section, we illustrate how to solve the multi-objective optimisation problem.

4.3.1 Multi-Objective Optimisation Algorithms

Solving multi-objective optimisation problems is not as simple as a conventional single-objective optimisation problem as there are multiple Pareto optimal solutions. On exploring different ways to solve multi-objective optimisation problems, one such approach is to transform the multi-objective optimisation problem into a single objective optimisation problem. This is called scalarization or the weighted sum technique (Caramia and Dell'Olmo, 2008). A multi-objective optimisation generates a set of solutions called Pareto solutions or a Pareto frontier. The best solution is selected from the Pareto frontier. Evolutionary algorithms are predominant approaches for generating Pareto optimal solutions. Non-dominated Sorting Genetic Algorithm- II (NSGA- II) and Strength Pareto Evolutionary Algorithm 2 (SPEA 2) are standard evolutionary approaches. We have used the NSGA-II (Deb et al., 2002) evolutionary algorithm that has an implementation in MATLAB Global Optimization Toolbox. We briefly explain the use of this algorithm for finding the optimal weights of the leaf softgoals in goal models.

The evolutionary algorithm begins with a population of randomly generated individuals and finds a solution over a number of iterations. The population in each iteration is known as a 'generation'. In each generation, the fitness of the selected individual known as a chromosome (weight of leaf softgoal) in the population is evaluated. The fitness is the value of the objective function in the optimisation problem. The selected individuals from the current population are randomly mutated by a process called crossover to form a new generation, which is used in the next iteration. The algorithm can be terminated when either a maximum number of generations have been produced or a satisfactory fitness level has been reached for the population.

4.3.2 Encoding the Optimisation Problem for Weights of the Leaf Softgoals

Binary representations are used to show the chromosomes in genetic algorithms. Therefore, we define a mapping from weights of the leaf softgoals in the goal model to a binary representation. The aim of the objective function is to find the weights of the leaf softgoals. So, the number of bits in a chromosome depends on the number of leaf softgoals in the goal model. In the LAS goal model (Figure 4.1), for the actor *Resource Allocator*, the number of leaf softgoals is two; hence, the number of bits in the chromosome is 2. Let us represent the actors *Ambulance*

Crew, *Resource Allocator*, *Resource Allocator Module* and *Human Resource Allocator* with numbers 1, 2, 3, and 4 respectively. The actor *Ambulance Crew* has no alternatives and has no objective functions. Using equation 4.8, the objective functions for the LAS are obtained.

For the actor *Resource Allocator*, the objective functions for the alternatives *ByPaperbased Form* (F1) and *ByDatabase or Network* (F2) are given as

$$F1 = Max(0.16 * \omega_{L12} + 0.16 * \omega_{L22})$$
$$F2 = Max(0.64 * \omega_{L12} + 0.64 * \omega_{L22})$$

For the actor *Resource Allocator Module*, the objective functions for the alternative *ByMachineBasedAlgorithm* is

$$F3 = Max(0.64 * \omega_{L13} + 0.64 * \omega_{L23})$$

For the actor *Human Resource Allocator*, the objective function for the alternative *ByHumanDecision* is

$$F4 = Max(0.16 * \omega_{L14} + 0.16 * \omega_{L24})$$

The collective objective functions are given as

$$F1 = Max(0.16 * \omega_{L12} + 0.16 * \omega_{L22})$$
$$F2 = Max(0.64 * \omega_{L12} + 0.64 * \omega_{L22})$$
$$F3 = Max(0.64 * \omega_{L13} + 0.64 * \omega_{L23})$$
$$F4 = Max(0.16 * \omega_{L14} + 0.16 * \omega_{L24})$$

Subject to

$$0 \leq \omega_{Li} \leq 100 \text{ for } i = 1 \text{ to } 2$$

Using the scalarization method, the objective functions are combined into a single objective function and is given by

$$Max\{\gamma_1(0.16 * \omega_{L12} + 0.16 * \omega_{L22}) + \gamma_2(0.64 * \omega_{L12} + 0.64 * \omega_{L22}) + \gamma_3(0.64 * \omega_{L13} + 0.64 * \omega_{L23}) + \gamma_4(0.16 * \omega_{L14} + 0.16 * \omega_{L24})\}$$
$$\sum_{i=1}^{4} \gamma_i = 1$$
$$0 \leq \omega_{L1j}, \omega_{L2j} \leq 100, j = 2 \text{ to } 4$$
$$\gamma_i \geq 0, i = 1, 2, 3, 4$$

Figure 4.3 illustrates the process of crossover with $\gamma_1 = 1$ and $\gamma_2, \gamma_3, \gamma_4$ as zeros. In this figure, we have two bits in the chromosome representing the weights two leaf softgoals ω_{L12} and ω_{L22}. We randomly assign the values of these bits representing the initial population. For the first population, we randomly select 23 and 56 to ω_{L12} and ω_{L22} respectively. Similarly, for the second population we randomly assign 60 and 35 to ω_{L12} and ω_{L22} respectively. With this initial population, the value of the objective function for the first population and second population are 12.64 and 15.2 correspondingly. Crossover is performed by mutating the second bit of second population with the second bit of first population resulting with 60 and 56 for the current population. The objective function value for the crossover is 18.29. The process is repeated for a specified number of iterations or until an optimal solution is found.

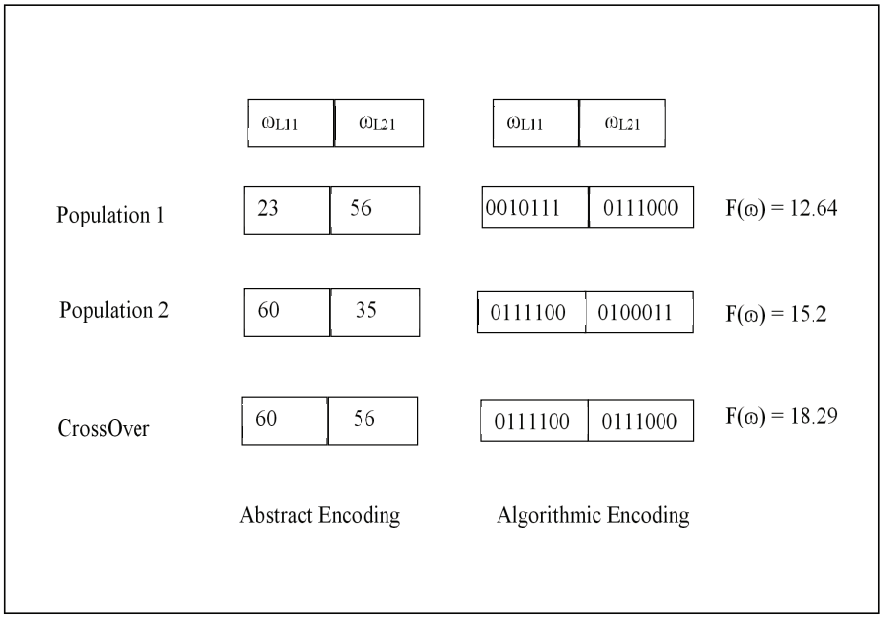

Figure 4.3: Crossover for Optimal weights

4.4 Application of Optimisation to a Case Study

In the LAS case study, the actor the *Resource Allocator* has goals *Becollected[IncInfo]* and *BeGenerated[MobileInst]* as two decision points. The goal *Becollected[IncInfo]* is *OR* decomposed into two tasks:

- ByPaperbased Form and

- ByDatabase or Network

and the goal *BeGenerated[MobileInst]* is *OR* decomposed into two tasks:

- ByMachineBasedAlgorithm
- ByHumanDecision.

The objective of the optimisation approach is to find the weights of the leaf softgoals *Accuracy* and *Timeliness[mobilization]* that, when used in goal analysis, maximises satisfactions for the softgoals *Optimal[mobInst]* and *Timelineness[mobilization]* of the actor *Resource Allocator* and *Quality[service]* of the actor *Ambulance Crew*. The softgoal *Timelineness[mobilization]* is leaf softgoal as well as top softgoal.

Let us represent the weight obtained from the optimiser as the optimal weight, which is denoted by $O\omega$. The leaf softgoal score equation 4.1 is updated as:

$$\bar{S}_{Lijkt} = \bar{C}_{Aj*Li} * \mathrm{O}\omega_{Lik} + \sum_{b=1}^{nd} \left(\bar{S}_{db} * \bar{I}_{db}\right) \tag{4.15}$$

The LAS goal model consists of four alternatives known as *ByPaperbased Form*, *ByDatabase or Network*, *ByMachineBasedAlgorithm* and *ByHumanDecision*. Based on the number of alternatives, this model has four objective functions, one for each alternative. The objective function for the alternative *ByPaperbased Form*, denoted by $F1$ is given by:

$$F1(\omega_L) = Max \sum_{i=1}^{2} \left(\bar{C}_{A1*Li2} * \omega_{Li2}\right)$$

$$F1 = Max(0.16 * \omega_{12} + 0.16 * \omega_{22})$$

where ω_{12}, ω_{22} represent the weights of the leaf softgoals *Accuracy* and *Timeliness* of actor *Resource Allocator* (actor number 2). For the convenience of solving the objective functions, the defuzzified values of correlation links are used in the functions.

Similarly, the objective functions for the other three options *ByDatabase or Network*, *ByMachineBasedAlgorithm* and *ByHumanDecision* are given by $F2, F3$ and $F4$ correspondingly:

$$F2 = Max(0.64 * \omega_{12} + 0.64 * \omega_{22})$$

$$F3 = Max(0.64 * \omega_{13} + 0.64 * \omega_{23})$$

$$F4 = Max(0.16 * \omega_{14} + 0.16 * \omega_{24})$$

The multi-objective functions for the LAS goal model are:

$$F1 = Max(0.16 * \omega_{12} + 0.16 * \omega_{22})$$
$$F2 = Max(0.64 * \omega_{12} + 0.64 * \omega_{22})$$
$$F3 = Max(0.64 * \omega_{13} + 0.64 * \omega_{23})$$
$$F4 = Max(0.16 * \omega_{14} + 0.16 * \omega_{24})$$

Subject to

$$0 \leq \omega_{ij} \leq 100 \text{ for } i = 1 \text{ to } 2 \text{ and } j = 2 \text{ to } 4$$
$$0.64 * \omega_{ij} \leq 100 \text{ for } i = 1 \text{ to } 2 \text{ and } j = 2 \text{ to } 4$$
$$0.16 * \omega_{ij} \leq 100 \text{ for } i = 1 \text{ to } 2 \text{ and } j = 2 \text{ to } 4$$

By solving, the above multi-objective functions using the MATLAB Genetic Algorithm, the weights of the leaf softgoals are identified and are presented in Table 4.1. These weights are now used in equation 4.15 to find the scores of leaf softgoals and thereby to find the optimal satisfaction of the softgoals and top softgoals. The calculated scores of the softgoals for the alternative option *ByDatabase or Network* are provided in Figure 4.4.

Table 4.1: Weights of the leaf softgoals of the actor Resource Allocator

Actor	Leaf Softgoals	Weight
Resource Allocator	Accuracy	0.9
	Timeliness[mobilization]	0.8

In Figure 4.4, it can be seen that the alternative option *ByDatabase or Network* was estimated to achieve 100%, 76% and 100% for the top softgoals of *Quality[service](Ambulance Crew)*, *Optimal[mobInst](Resource Allocator)* and *Timeliness[mobilization](Resource Allocator)* respectively. Similarly, goal analysis is performed for other alternative options. The scores of the softgoals for all the alternatives are given in Table 4.2. From the table, it can be seen that the alternative options *ByDatabase or Network* and *ByMachineBasedAlgorithm* were found to be the optimal alternative options for the goals *BeCollected[IncInfo]* and *BeGenerated[mobileInst]* respectively.

4.4.1 Evaluation of the Approach using a Case Study

For the evaluation of the approach, we illustrated the optimisation of the i^* framework using another case study - the Telemedicine goal (adapted from (Yu,

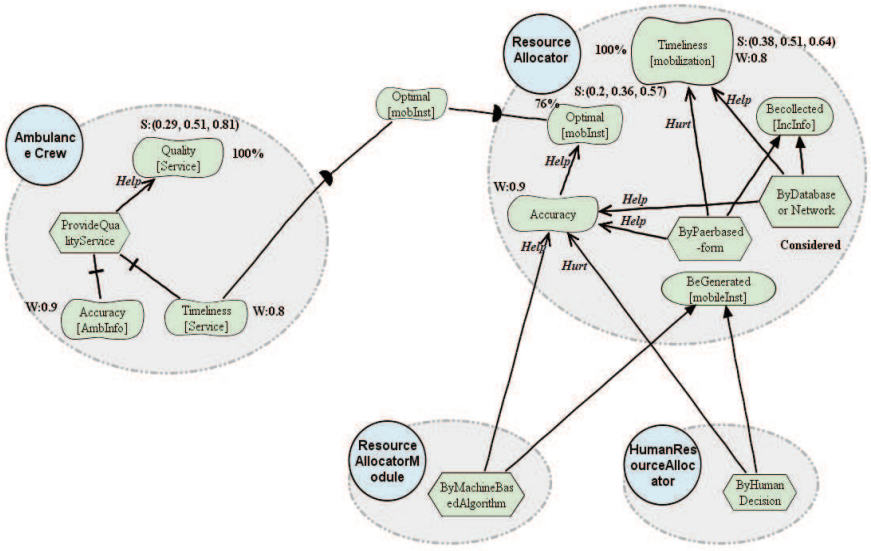

Figure 4.4: Optimal softgoal scores for the alternative option *ByDatabase or Network*

Table 4.2: Top softgoal scores for all the alternative options in LAS case study

Actor	Top softgoal	BeCollected[IncInfo]		BeGenerated[mobileInst]	
		ByDatabase or Network	ByPaperbased Form	ByMachineBasedAlgorithm	ByHumanDecision
Ambulance Crew	Quality[service]	100%	100%	100%	100%
Resource Allocator	Optimal[mobInst]	76%	20%	76%	20%
	Timeliness[mobilization]	100%	100%	100%	100%

2001)) model. In the Telemedicine system (Figure 4.5), a patient depends upon a healthcare provider for the treatment of an illness. A patient may prefer to have a flexible treatment plan which incorporates other activities. The healthcare provider may choose to honour the patient's request by monitoring the patient remotely with the use of patient monitoring equipment. In addition to this, the provider might enhance his/her performance by using a host system that manages a number of patients at the same time.

In the Telemedicine case study, we have two actors, namely *Patient* and *Healthcare Provider*.

For the two actors, the two different alternative tasks are:

- Patient Centred Care
- Provider Centred Care.

The objective of the optimisation is to find the weights of leaf softgoals that when used in goal analysis select an alternative that achieves maximum satisfacti-

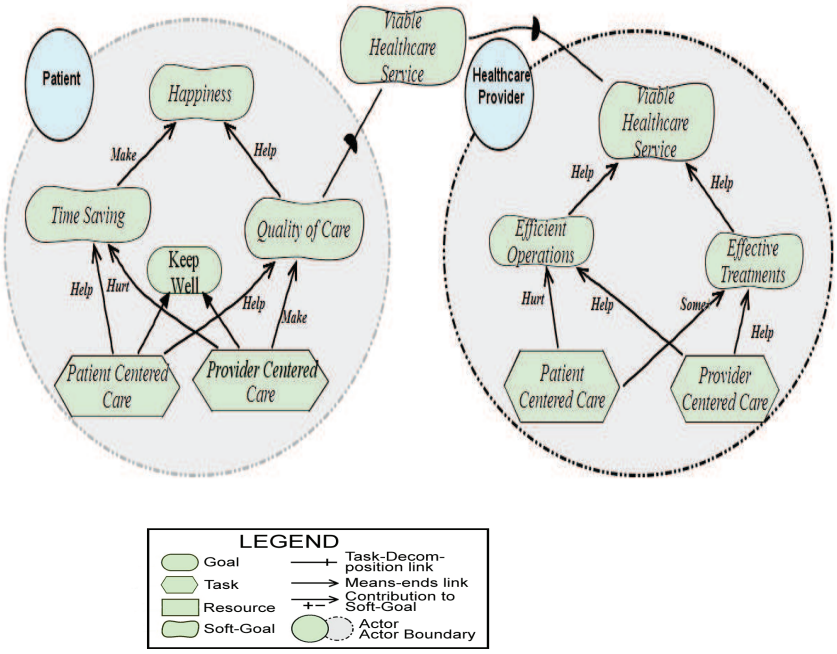

Figure 4.5: An SR Model of Telemedicine System (adapted from (Yu, 2001))

ons for the softgoals *Happiness* of actor *Patient* and *viable Healthcare Service* of actor *Healthcare Provider* .

The Telemedicine goal model consists of two actors: *Patient* and *Healthcare Provider* with the same type alternatives namely *Patient Centred Care* and *Provider Centred Care*. Based on the number of alternatives, this model has two objective functions one for each alternative. The objective function for the alternative option *Patient Centred Care* of both the actors using equation 4.11 is given by $F1$

$$F1 = Max \sum_{k=1}^{2} \sum_{i=1}^{2} \bar{C}_{A*Lik} * \omega_{Lik}$$

$$F1 = Max(0.64 * \omega_1 + 0.64 * \omega_2 + 0.16 * \omega_3 + 0.16 * \omega_4)$$

where $\omega_1, \omega_2, \omega_3, \omega_4$ represents the weights of the leaf soft goals *Normal Life Style*, *Quality of Care, Efficient Operations* and *Effective Treatments* in two actors. For the ease of use, the variables $\omega_1, \omega_2, \omega_3, \omega_4$ are used instead of $\omega_{11}, \omega_{21}, \omega_{12}$, ω_{22} respectively. For the convenience of solving these objective functions, the defuzzified values of correlation links are used in the functions. Similarly, the

objective function for the second alternative option, *Provider Centred Care* is given by equation 4.11:

$$F2 = Max(0.16 * \omega_1 + 0.81 * \omega_2 + 0.64 * \omega_3 + 0.64 * \omega_4)$$

The multi-objective functions for the Telemedicine goal model are:

$$F1 = Max(0.64 * \omega_1 + 0.64 * \omega_2 + 0.16 * \omega_3 + 0.16 * \omega_4)$$
$$F2 = Max(0.16 * \omega_1 + 0.81 * \omega_2 + 0.64 * \omega_3 + 0.64 * \omega_4)$$

Subject to

$$0 \leq \omega_{ij} \leq 100 \text{ for } i = 1 \text{ to } 4$$
$$0.64 * \omega_i \leq 100 \text{ for } i = 1, 2, 7, 8$$
$$0.16 * \omega_i \leq 100 \text{ for } i = 3, 4, 5, 6$$

By solving the above multi-objective functions using the MATLAB Genetic Algorithm, the weights of the leaf softgoals are found and are shown in Table 4.3. These weights are now used in our approach as described in Chapter 3 to find the optimal satisfaction of the top softgoals. The calculated scores of the softgoals for the alternative option *Provider Centred Care* are given in Figure 4.6.

Table 4.3: Optimised weights of leaf softgoals of the Telemedicine goal model

Actor	Leaf Softgoals	Weight
Patient	Normal Life Style	0.85
	quality of Care	0.99
Healthcare Provider	Efficient operations	0.9
	effective Treatments	0.75

In Figure 4.6, it can be noted that the alternative option *Provider Centred Care* was estimated to achieve 60%, and 69% for the top soft goals *Happiness (Patient)* and *Viable Health Service (Healthcare Provider)* correspondingly. To analyse the estimated values, these values are compared with the satisfaction values from the other alternative option *Patient Centred Care*. The top softgoals' values and the satisfaction comparison of these two alternatives are displayed in Table 4.4. The table shows that the alternative *Provider Centred Care* outperforms the *Patient Centred Care* in the actor *Healthcare Provider* whereas alternative *Patient Centred Care* outperforms the alternative *Provider Centred Care* in case of the actor *Patient*.

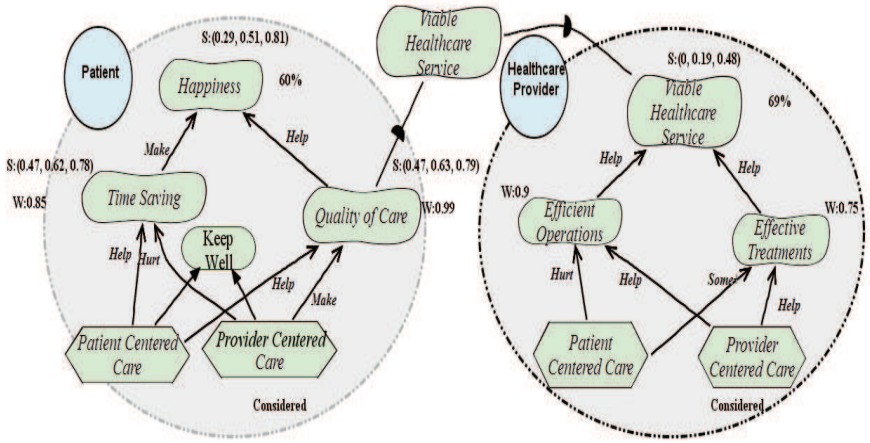

Figure 4.6: A Goal Analysis with optimal weights for alternative option *Provider Centred Care*

Table 4.4: Top SoftGoals Satisfaction Scores for Telemedicine Goal model (* indicates goal selection).

Actor	Top Softgoals	Alternative option scores		Defuzzified scores	
		Patient Centered Care	Provider Centered Care	Patient Centered Care	Provider Centered Care
Patient	Happiness	(0.41,0.73,1)	(0.3,0.597,0.9)	71% *	60%
Healthcare Provider	Viable Healthcare Service	(0,0.168,0.42)	(0.38,0.67,1)	21%	69%*

To analyse the effectiveness of the proposed approach, the scores obtained from optimised goal analysis are compared with scores obtained without optimisation. In goal analysis without using optimisation, weights have to be assigned to the leaf softgoals by a requirement analyst. Since the assigned weights are subjective to the analyst, different scores are obtained for the same softgoals based on the analyst's weights. To illustrate this, let us assume that an analyst assigns the weights 30%, 50%, 40% and 40% to the leaf soft goals *Normal Life Style, Quality of Care, Efficient Operations* and *Effective Treatments* respectively. The satisfaction scores of the softgoals *Happiness* and *Viable Healthcare Service* are found to be 35% and 67% respectively for the alternative option *Patient Centred Care*. Similarly, for the alternative option *Provider Centred Care*, the scores of the softgoals *Happiness* and *Viable Healthcare Service* are found to be 31% and 54% respectively. However, if another analyst assigns weights of 70%, 80%, 60% and 60% to the leaf softgoals *Normal Life Style, Quality of Care, Efficient Operations* and *Effective Treatments* respectively, then satisfaction scores of the softgoals *Happiness* and *Viable Healthcare Service* are found to be 11% and 15% respectively for the alternative option *Patient Centred Care*. For the alternative

option, *Provider Centred Care*, the scores of the softgoals *Happiness* and *Viable Healthcare Service* are 35% and 52% respectively. The scores of top softgoals for both the actors are given in Table 4.5 and Table 4.6. Hence the scores of the softgoals are subjective depending on the subjective selection of weights by the requirement analyst. In order to avoid these subjective scores, the proposed optimisation approach can be used. And also, by previewing the above calculated scores, it can be seen that the proposed optimised approach gives a better optimal softgoal scores than those without optimisation. The score comparisons are shown in Table 4.7 and the graphical representation of score comparison is shown in Figure 4.7. **Algorithm 2** outlines the steps in the optimal inter-actor goal analysis of the i^* framework.

Table 4.5: Top softgoal scores of Patient (for different weights)

Weights of LSG		Happiness Score	
Normal Life Style	Quality of Care	Patient Centered Care	Provider Centered Care
30	50	35%	31%
70	80	67%	54%

Table 4.6: Top softgoal scores of Healthcare Provider (for different weights)

Weights of LSG		Happiness Score	
Efficient Operations	Effective Treatments	Patient Centered Care	Provider Centered Care
40	40	11%	35%
60	60	15%	52%

Table 4.7: Scores Comparison for Telemedicine System (with and without Optimisation)

Actor	Top Softgoals	Patient Centered Care		Provider Centered Care	
		With Optimisation	Without Optimisation	With Optimisation	Without Optimisation
Patient	Happiness	71%	35%, 67%	60%	31%, 54%
Healthcare Provider	Viable Healthcare Service	21%	11%, 15%	69%	35%, 52%

4.5 Sensitivity Analysis

Another interesting point of optimisation is sensitivity analysis. Sensitivity analysis is employed to detect the system's behaviour when input data change. The

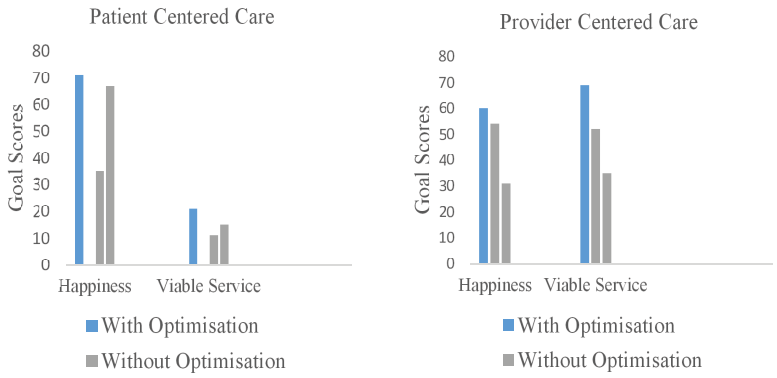

Figure 4.7: Graphical representation of scores comparisons with and without optimisation

main advantage of this technique is that a thorough investigation of the estimation of the input variables is conducted before a final decision is made. It also helps to identify the errors in the model and comprehend the effect of input parameters. The concept of sensitivity analysis is new to software engineering domains and this work is the earliest attempt at applying sensitivity analysis in case of the i^* framework. The only other research work on sensitivity analysis of goal models was conducted by Affleck et al. (2015). However, this research was conducted on a different framework known as a Non-Functional Requirements framework.

In regards to modelling, sensitivity analysis can help an analyst in numerous ways. Sensitivity analysis is one of the most appealing and interesting optimisation field (Hadigheh and Terlaky, 2006). Efforts are made to explore the problem's behaviour for changes in the input data. The following questions are used when conducting sensitivity analysis. What is the range of the input parameter? How positive or optimal are the results? How much will the result change if the data is slightly varied? Will these changes have a minor or a major impact on the results? Formally, the question is: Is optimal solution sensitive to a small change in one of the problem coefficients? Usually, variation occurs in the right hand side of the constraints and /or the objective function's coefficient. If the solution of the Linear Program (LP) changes, when the original coefficient is changed, then it is referred to as LP sensitive. Imagine a model in linear form,

Algorithm 2 Inter-actor optimal reasoning of Goals/tasks in the i^* Framework

INPUT :
Graph Collection G, Set of task (T), Leaf SoftGoals(LSG), SoftGoals(SG)

OUTPUT :
Optimal alternative task that is selected for each actor.

for all graph g in G **do**
  for all task t in T **do**
    for all leaf softgoal lsg in LSG **do**
      Obtain objective functions
      W[ ]= gamultiobj()
    end for
  end for
end for

for all graph g in G **do**
  for all task t in T **do**
    for all leaf softgoal lsg in LSG **do**
      $lsg.score = lsg.impact * W[lsg]$
    end for
  end for
end for
for all graph g in G **do**
  for all softgoal sg in SG **do**
    for all child C in SG **do**
      $sg.score+ = sg.impact * C_{lsg/sg}.score$
    end for
    for all dependent sg in SG **do**
      $sg.score+ = dsg.impact * dClsg/sg.score$
    end for
  end for
end for
$optimal\_alternative = maxscore(sg.score[])$

$$Y = \sum_{i=1}^{n} C_i X_i$$

$$lowerlimit \leq C_i \leq upperlimit$$

where the input parameters are C_i, lowerlimit and upperlimit. Now by changing the values of the input parameters the sensitivity analysis is performed. A diagrammatic view of sensitive analysis for an optimisation model is given in Figure 4.8.

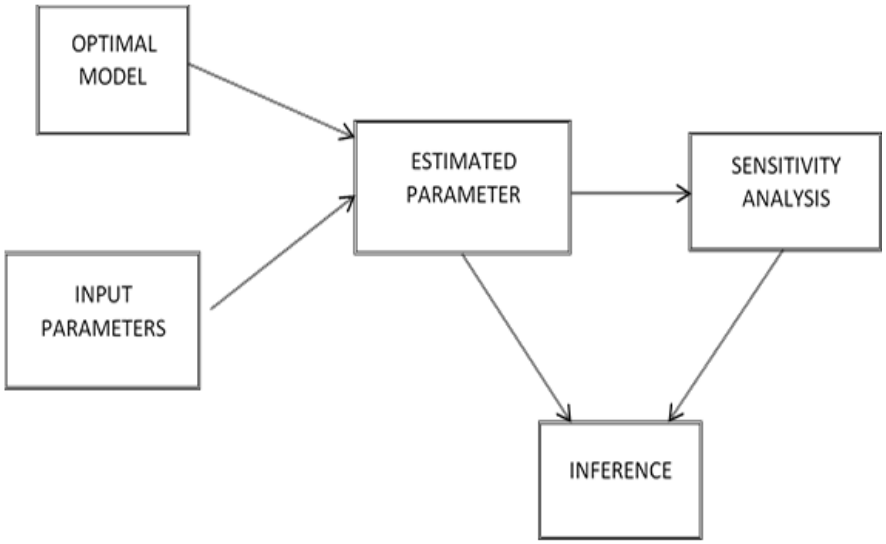

Figure 4.8: A model for the sensitivity analysis of optimisation

Sensitivity analysis capability is provided by most optimisation tools; however, the data obtained is dependent on continuous variables. This control on maintaining integer variables increases the complexity of the problem, and therby decreasing the computational efficiency LP (date).

4.5.1 Sensitivity Analysis of the i^* Framework

To overcome the above mentioned issue, a simulation was created to check the system behaviour for change after each input parameter. The values of the input parameter are altered until a change in the solution takes place. The sensitive data provide the range for an input data where there is no change in the optimal output value. The analyst is alerted if the value exceeds the range obtained from

the sensitivity analysis. An analyst can take action by re-considering the input data. For our optimal i^* framework, the objective function in simple form is:

$$F = Max \sum_{i=1}^{m} \bar{C} * \omega$$

The input parameter on the right side of the objective function is the impact of the alternative on the leaf softgoals and it is given by triangular fuzzy number $(a1, a2, a3)$. Usually, the choice of fuzzy numbers varies from expert to expert. Hence, it is interesting to observe the dependency of the solutions obtained from the parameters of the fuzzy numbers. A special case is now considered in which these numbers are perturbed by $\delta 1$ and $\delta 2$ as in Figure 4.9.

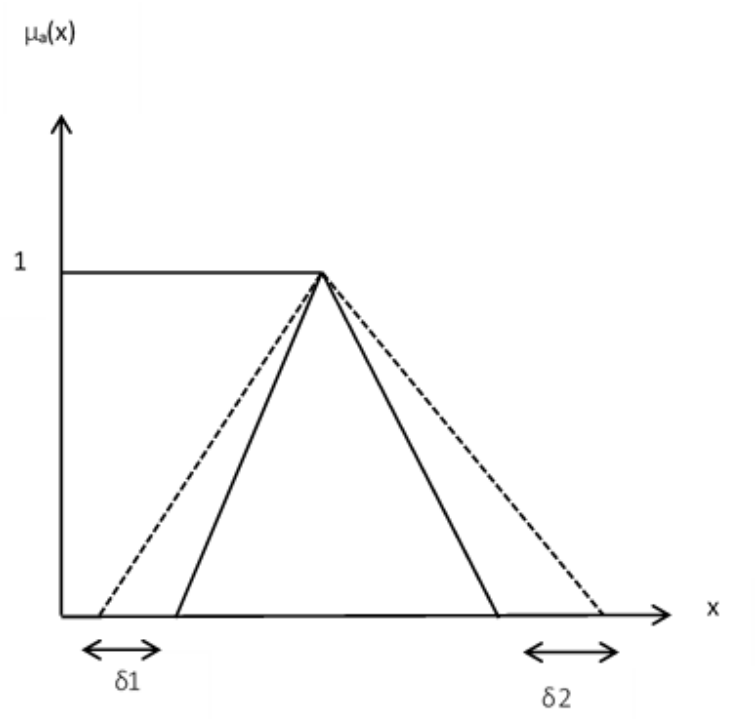

Figure 4.9: Perturbation of fuzzy number

In this case, the task is to find the range in which $\delta 1$ and $\delta 2$ may vary without violating the optimal solution. The impacts of the goals(or task) on softgoals *make, help, some+, some-, hurt, and break* are represented by triangular fuzzy numbers (0.64, 0.80, 1), (0.48, 0.64, 0.80), (0.32, 0.48, 0.64), (0.16, 0.32, 0.48),

(0, 0.16, 0.32) and (0, 0, 0.16) respectively. For each type of impact, the lower and upper bounds are varied, except for *make and break*, because the impacts are represented as fuzzy numbers from 0 to 1. To improve the analysis, initially the lower (upper) bound is tested and moved on, only if the bound is not a limit for the analysis. The middle point between the current impact's lower (upper) limit, and adjacent impact's lower (upper) limit, forms the lower (upper) bound

$$m_l = (l_1 + l_2)/2 \quad m_u = (u_1 + u_2)/2$$

Where m_l is the middle point for lower limit, m_u is the middle point for the upper limit, l_1 is the current impact's lower limit, l_2 is the adjacent impact's lower limit, u_1 is the current impact's upper limit and u_2 is the adjacent impact's upper limit. The values are decreased (increased) from this middle point until a change in the optimal solution occurs.

4.5.2 Analysis with LAS Case Study

In the LAS goal model (Figure 4.1), *help* and *hurt* are the two impacts associated with the alternative options. The fuzzy values of *help* and *hurt* are (0.48, 0.64, 0.8) and (0, 0.16, 0.32) respectively. The lower bound of *help* and the upper bound of *hurt* are varied to check for any change in the optimal satisfaction levels of the top softgoals. The upper bound of *help* is not varied, because the upper limit of the membership function is 1. Similarly, the lower bound of *hurt* is not changed, because the lower limit of the membership function is 0. For example, for the actor *Resource Allocator*, the impact of the alternative option B*yDatabase or Network* on the leaf softgoal *Timeliness[mobilization]* has an impact value of (0.48, 0.64, 0.8). According to sensitivity analysis, the lower bound is varied to see if there is any change in the optimal satisfaction value. The impact has the new bounds (0.48, 0.64, 0.8), (0.2, 0.64, 0.8); this implies that the impact can take any value in this range without a change occurring in the optimal satisfaction levels of the softgoals. Similarly, the impact of the alternative option, *ByPaperbased Form* on the leaf softgoal *Timeliness[mobilization]* has an impact value of (0, 0.16, 0.32). According to sensitivity analysis, the impact was found to have the bounds (0, 0.16, 0.32), (0, 0.16, 0.6). Figures 4.10 and 4.11 demonstrate the graphical representation of the sensitivity analysis for the top softgoals, *Timeliness[mobilization]* and *Optimal[mobInst]* of the actor *Resource Allocator*, with both impacts being taken into consideration. For instance, the graph demonstrates the score of the softgoals for both of the impacts *help* and *hurt*. It

is found that beyond the sensitivity analysis bounds of the impacts, the optimal satisfaction scores of the softgoals decrease and within the specified bound the optimal solution, remains unchanged.

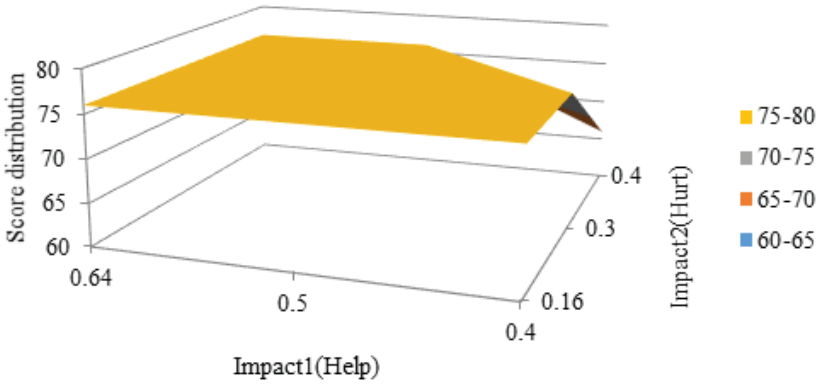

Figure 4.10: Sensitivity Analysis of the softgoal *optimal[mobInst]*

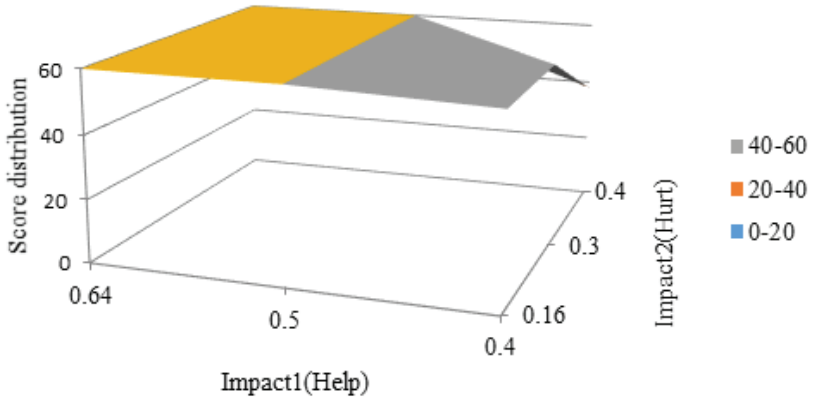

Figure 4.11: Sensitivity Analysis of softgoal Timeliness[mobilisation]

The benefit of sensitivity analysis is that it helps the analyst to decide whether the inputs are within the accepted range. Also, analysts can examine the solutions obtained from different inputs and decide on the best solution. Furthermore, the analyst need not perform the sensitivity analysis every time optimisation is conducted; it is done only when access to the data provided is required. This

means that when an impact comes under review the analyst can make a demand for a calculation of bounds.

4.6 Chapter Summary

In this chapter, we presented a method of representing the i^* framework as a directed graph. This graph was then used to create a multi-objective function optimisation model for the i^* framework based on maximising the top softgoals of a given system. The optimisation model was used to obtain the weights of the leaf softgoals that are used for the goal analysis. The optimisation model was evaluated using the case studies from the existing literature: the London Ambulance System, and the Telemedicine System. Furthermore, a sensitivity analysis approach was developed and implemented to examine the solution used for the London Ambulance System. The optimisation assists the analyst to identify the weights of leaf softgoals, thereby avoiding subjective preferences. The sensitivity analysis helps the analyst to determine the bounds of the inputs, for which there is no change in the optimal solution. In the next chapter, we discuss a complete optimisation model for the i^* framework using the goal programming approach.

Chapter 5

Optimal Goal Programming of Softgoals in the i^* Framework

In Chapter 4, we presented a method for finding the weights of the leaf softgoals using the optimisation approach. These optimal weights are used in goal analysis to select an option from a given set of alternatives. Although the analysis used in the previous chapter was less intensive and can be scaled to a very large number of design alternatives, it performs only a partial optimisation of the goal model. This optimisation model is based upon the leaf softgoals in an SR model of the i^* framework and does not take into consideration the other softgoals within the hierarchy. As the alternative choices selected are based upon the propagation of values throughout the entire hierarchy of softgoals, an optimal model still needs to be developed that takes into consideration all of the softgoals within the hierarchy. Hence, in this chapter, we address this particular limitation, by developing an optimisation model that is based on all of the softgoals and leaf softgoals in an SR model of the i^* framework, presenting a complete optimisation model for the i^* framework. In Section 5.1, we explain multi-objective optimisation in the i^* framework, building a complete optimisation model based upon the softgoals of a given i^* goal model. In Section 5.2, we give an introduction to multi-objective goal programming and in Section 5.3 the formulation of the i^* framework as a multi-objective goal model is presented. A simulation-based evaluation of the approach is presented in Section 5.4.

Some of the material in this chapter has previously appeared in the following publication:

1. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2016). Optimal Goal Programming of Softgoals in Goal- Oriented Requirements Engineering. The 20th Pacific

Asia Conference on Information Systems (PACIS), Taiwan June 27- July 1.

5.1 Optimisation Model based on SoftGoals in the *i** Goal Model

To obtain a complete, generalised optimisation model of an i^* framework in terms of softgoals, consider the Strategic Rationale (SR) diagram as a directed graph G (N, A), with N as a set of nodes and A as set of arcs (Figure 5.1). The intentional elements comprising softgoals, goals, and tasks in the SR diagram form the nodes of the directed graph G and the means-end, task-decomposition and operational contribution links form the arcs of graph G.

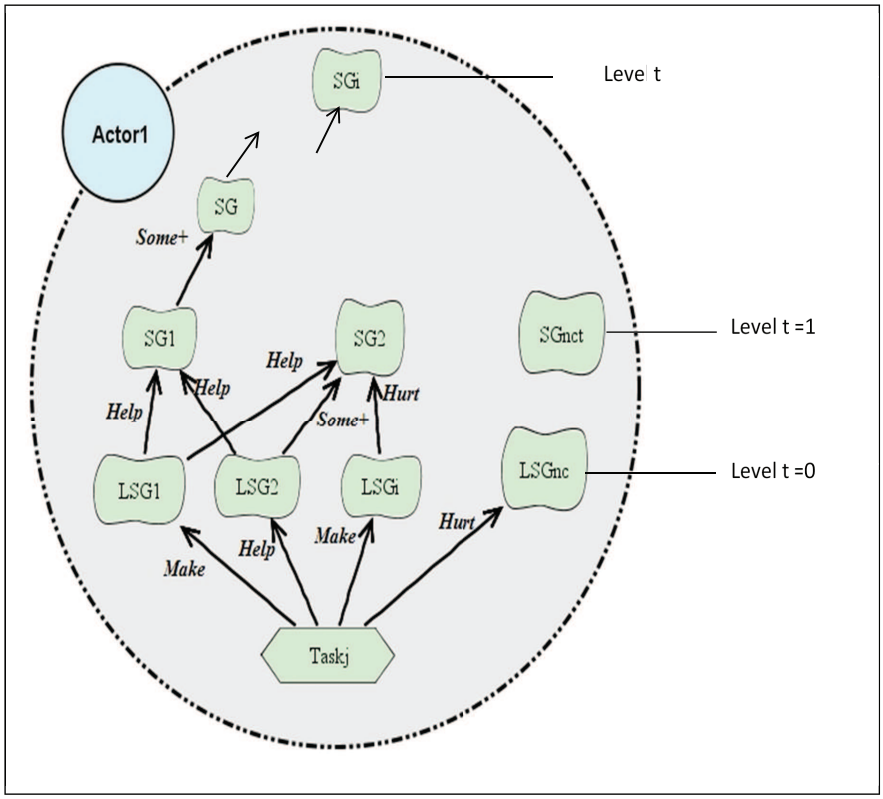

Figure 5.1: Directed Graph representation of a SR diagram

An objective function for the optimisation model is created in terms of the elements of the graph. For any node that is a leaf softgoal (LSG), there is a weight represented by ω_{LSGik}, representing the weight of the i^{th} leaf softgoal of the k^{th} actor. Additionally, any arc from a goal/task to a leaf softgoal that is an impact of goal/task is denoted by a triangular fuzzy number $\bar{C}_{Aj*LSGi}$, denoting the impact of the j^{th} alternative (A) option on i^{th} leaf softgoal. The i^{th} leaf softgoal score is calculated from the weight of the i^{th} leaf softgoal and its impact

for j^{th} alternative and is represented by $\bar{S}_{LSGij}$.

The score of i^{th} leaf softgoal for j^{th} alternative of k^{th} actor is given by

$$\bar{S}_{LSGijkt} = \bar{C}_{(Aj*LSGij)k} * \omega_{LSGijk} \tag{5.1}$$

for leaf softgoals t is zero. The dependency score part is avoided since we are focusing on the optimisation of SR only.

The score calculation of an i^{th} softgoal for j^{th} alternative of k^{th} actor at t^{th} level in the hierarchy is given by $\bar{S}_{SGijkt}$

$$\bar{S}_{SGijkt} = \sum_{d=1}^{nc} \left(\bar{C}_{SGij*(SGdj|LSGdj)} * \bar{S}_{LSGdjk(t-1)|SGdjk(t-1)}\right) \tag{5.2}$$

Where *'nc'* is the number of leaf softgoals.

To find an optimal model based on softgoals, we need to write the softgoal score equation as a leaf softgoal score equation. To perform this, let us start with softgoals at level 1, which depends on the scores of leaf softgoals. The score of an i^{th} softgoal with *'nc'* children for j^{th} alternative of k^{th} actor at level $t = 1$ is given by expanding equation 5.2

$$\begin{aligned}\bar{S}_{SGijk} = \left(\bar{C}_{(SGij*LSG1j)} * \bar{S}_{LSG1jk}\right) + \left(\bar{C}_{(SGij*LSG2j)} * \bar{S}_{LSG2jk}\right) \\ +.... + \left(\bar{C}_{(SGij*LSGncj)} * \bar{S}_{LSGncjk}\right)\end{aligned}$$

Replacing the scores of leaf softgoal with equation 5.1,

$$\begin{aligned}\bar{S}_{SGijk} = \{(\bar{C}_{(SGij*LSG1j)} * \bar{C}_{(Aj*LSG1j)k} * \omega_{LSG1jk} \\ +(\bar{C}_{(SGij*LSG2j)} * \bar{C}_{(Aj*LSG2j)k} * \omega_{LSG2jk} \\ +.....+ \\ (\bar{C}_{(SGij*LSGncj)} * \bar{C}_{(Aj*LSGncj)k} * \omega_{LSGncjk}\}\end{aligned}$$

$$\bar{S}_{SGijk} = \sum_{d=1}^{nc} (\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk}) \tag{5.3}$$

If there are 'm' softgoals at level 1 and to obtain maximum softgoal score, the sum of softgoal scores have to be maximized. Therefore, the objective function is given by

$$Max\{\bar{S}_{SG1jk} + \bar{S}_{SG2jk} + ... + \bar{S}_{SGmjk}\}$$

By replacing the softgoal score with equation 5.3

$$Max\{\left[\sum_{d=1}^{nc}(\bar{C}_{(SG1j*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right] + \left[\sum_{d=1}^{nc}(\bar{C}_{(SG2j*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right] +....+ \left[\sum_{d=1}^{nc}(\bar{C}_{(SGmj*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right]\}$$

Rewriting the above equation

$$Max\{\left[\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right]\}$$

The score of any softgoal at hierarchy level $t > 1$ is obtained by multiplying its impact by the child score. In this way, it propagates upwards. Therefore, for any softgoal at level *'t'* , the score can be generalised as

$$Max\Pi_{l=1}^{t}\bar{C}_{SGijl}\sum_{i=1}^{m}\{\left[\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right]\}$$

Hence, the objective function to obtain optimal weight of the leaf softgoals that maximise the top softgoal scores for the j^{th} alternative is

$$Max\Pi_{l=1}^{t}\bar{C}_{SGijl}\sum_{i=1}^{m}\{\left[\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk})\right]\} \tag{5.4}$$

Such that

$$0 \leq \omega_{LSGd} \leq 100 \text{ for } d = 1 \text{ to } nc$$

Hence, the objective function for the i^* framework is now given by equation 5.5 as below:

$$Max\Pi_{l=1}^{t}\bar{C}_{SGijl}\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk}) \tag{5.5}$$

Such that

$$0 \leq \omega_{LSGd} \leq 100 \text{ for } d = 1 \text{ to } nc$$

If there are *'n'* alternatives for an actor, then there are *'n'* objective functions as follows:

$$\begin{aligned}
f_1(\omega_L) &= Max\Pi_{l=1}^{t}\bar{C}_{SGi1l}\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGi1*LSGd1)} * \bar{C}_{(A1*LSGd1)k} * \omega_{LSGd1k}) \\
f_2(\omega_L) &= Max\Pi_{l=1}^{t}\bar{C}_{SGi2l}\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGi2*LSGd2)} * \bar{C}_{(A2*LSGd2)k} * \omega_{LSGd2k}) \\
&\ldots\ldots \\
&\ldots\ldots \\
f_n(\omega_L) &= Max\Pi_{l=1}^{t}\bar{C}_{SGinl}\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGin*LSGdn)} * \bar{C}_{(An*LSGdn)k} * \omega_{LSGdnk})
\end{aligned} \tag{5.6}$$

Such that

$$0 \leq \omega_{LSGd} \leq 100 \text{ for } d = 1 \text{ to } nc$$

Likewise, for each actor in the SR model, objective functions are generated. Cumulative objective functions can be generated if all the actors have the same type of alternatives. In a goal model with *'p'* number of actors, the objective function for a j^{th} alternative option is given by:

$$\begin{aligned}
f_j(\omega_L) = Max\Pi_{l=1}^{t}\bar{C}_{SGijl}&\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)1} * \omega_{LSGdj1}) \\
+\Pi_{l=1}^{t}\bar{C}_{SGijl}&\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)2} * \omega_{LSGdj2}) \\
&+....+ \\
\Pi_{l=1}^{t}\bar{C}_{SGijl}&\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)p} * \omega_{LSGdjp})
\end{aligned}$$

In short, the function is given by:

$$f_j(\omega_L) = Max\sum_{k=1}^{p}\Pi_{l=1}^{t}\bar{C}_{SGijl}\sum_{i=1}^{m}\sum_{d=1}^{nc}(\bar{C}_{(SGij*LSGdj)} * \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk}) \tag{5.7}$$

Therefore, the objective functions for a goal model with *'n'* number of alternatives are given by:

$$f_1(\omega_L) = Max \sum_{k=1}^{p} \Pi_{l=1}^{t} \bar{C}_{SGi1l} \sum_{i=1}^{m} \sum_{d=1}^{nc} (\bar{C}_{(SGi1*LSGd1)} * \bar{C}_{(A1*LSGd1)k} * \omega_{LSGd1k})$$

$$f_2(\omega_L) = Max \sum_{k=1}^{p} \Pi_{l=1}^{t} \bar{C}_{SGi2l} \sum_{i=1}^{m} \sum_{d=1}^{nc} (\bar{C}_{(SGi2*LSGd2)} * \bar{C}_{(A2*LSGd2)k} * \omega_{LSGd2k})$$

.....

.....

$$f_n(\omega_L) = Max \sum_{k=1}^{p} \Pi_{l=1}^{t} \bar{C}_{SGinl} \sum_{i=1}^{m} \sum_{d=1}^{nc} (\bar{C}_{(SGin*LSGdn)} * \bar{C}_{(An*LSGdn)k} * \omega_{LSGdnk}) \tag{5.8}$$

Subject to

$$0 \leq \omega_{LSGdk} \leq 100 \text{ for } d = 1 \text{ to } nc \text{ and } k = 1 \text{ to } p$$

$$0 \leq \bar{C}_{(Aj*LSGdj)k} * \omega_{LSGdjk}) \leq 100 \text{ for } d = 1 \text{ to } nc, j = 1 \text{ to } nc \text{ and } k = 1 \text{ to } p$$

In general, the multiple objective functions are given by following equation:

$$Max f(\omega_L) = [f_1\omega_L, f_2\omega_L, ..., f_n\omega_L]$$
$$\text{with } \omega_L \epsilon Y \tag{5.9}$$

where $n > 1$ and Y is the set of constraints defined

A great strategy for solving the above type of multi-objective linear programming (MOLP) is goal programming (Oliveira and Saramago, 2010; Caramia and Dell'Olmo, 2008). The following section gives an introduction to multi-objective goal programming.

5.2 Multi-Objective Goal Programming (MOGP)

Optimisation problems usually include circumstances involving minimizing and/or maximizing several conflicting functions simultaneously. Such cases are specified as multi-objective optimisation problems, also known as multicriteria, multiperformance, or vector optimizations. Among different approaches used to solve multi-objective functions, goal programming proposed by Charnes and Cooper (Charnes and Cooper, 1961), is a highly effective strategy which can be used to solve multi-objective problems by assigning multiple goals (Caramia and Dell'Olmo, 2008).

Goal programming requires that the user designates targets/goals for each objective that needs to be met. The main concept in goal programming is to compute solutions that achieve a predefined goal for one or more objective functions. Thus, goal programming involves expressing a set of goals $g = [g_1, g_2, ..., g_n]$ with a set of objectives, $f(x) = [f_1(x), f_2(x), ..., f_2(x)]$. The optimization problem can be formulated as

$$f_i(x) = g_i, i = 1, ..., n; x\epsilon\Omega \tag{5.10}$$

where Ω is the feasible search region.

In the next section, we describe the formulation of the i^* framework as a multi-objective goal model.

5.3 Formulation of the i^* Framework as a Multi-Objective Goal Model

The goal programming method requires that the decision maker specifies a goal or a target for each objective (a set of goals for an MOLP) that he/she wishes to achieve. The objective of goal programming is to obtain a predefined target for one or more objective functions. If no solution reaches predefined targets for all of the objective functions, then preferred solutions that minimize the deviations from those goals are to be identitfied. Presented below is a formal description of the optimisation problem.

The MOLP of the i^* framework as given by equation 5.9 is

$$Maxf(\omega_L) = [f_1\omega_L, f_2\omega_L, ..., f_n\omega_L]$$
$$\text{with } \omega_L\epsilon Y$$

where $n > 1$ and Y is the set of constraints defined

In goal programming, the user chooses the goal value g, for every objective function and the task is then to focus on making each objective $f_i\omega_L$ as close to its goal g_i as possible, subject to the condition that the resulting solution is feasible ($\omega_L\epsilon\Omega$). The optimisation problem can be formulated as follows:

$$\text{goal } f_i\omega_L = g_i, i = 1, ..., n; \omega_L\epsilon\Omega, \tag{5.11}$$

where Ω is the feasible region.

Two positive deviation variables, v_i^-, v_i^+ are introduced representing the under and over achievement of the i^{th} goal g_i for the i^{th} objective $f_i\omega_L (i = 1, 2, ..., n)$ respectively. Now the objective is to minimize the sum of the deviations (v_i^-, v_i^+), so that the optimal solution is minimally distant from the goal, in either direction. The optimisation problem is now remodelled as follows

$$\begin{aligned} \min_{\omega_L, v_1^-, v_1^+, ..., v_n^-, v_n^+} \quad & v_1^-, v_1^+, ..., v_n^-, v_n^+ \\ & \text{s.t} \\ & f_1\omega_L + v_1^- - v_1^+ = g_1, \\ & f_2\omega_L + v_2^- - v_2^+ = g_2, \\ & .... \\ & .... \\ & f_n\omega_L + v_n^- - v_n^+ = g_n, \\ & v_1^-, v_1^+, ..., v_n^-, v_n^+ \geq 0 \\ & \omega_L \epsilon \Omega, \end{aligned} \tag{5.12}$$

To perform direct comparisons of the objectives, a requirements analyst can use weighted or non-pre-emptive goal programming. To indicate the relative importance of the objectives, all of the deviations between the objectives and goals are multiplied by weights and are expressed as a standard optimisation problem using the following formulation.

$$\begin{aligned} \text{Min v } = & \sum_{i=1}^{n} \alpha_i v_i^- + \beta_i v_i^+ \\ & \text{Subject to} \\ f_i\omega_L + v_i^- - v_i^+ = g_i, , i = 1, 2, ..., n & \\ v_i^-, v_i^+ \geq 0, \omega_L \epsilon \Omega, & \end{aligned} \tag{5.13}$$

MOGP demands that goals are assigned for each objective and a favoured solution is designated to be one that minimises the deviations of the goals.

Let us assume that, the goals $g = (g_1, g_2, ..., g_n)$ for the objective functions $f\omega_L = (f_1\omega_L, ..., f_n\omega_L)$ are given by a requirements analyst, A decision variable $\omega_L^* \epsilon W_L$ in the MOLP problem is sought, such that the objective functions, $f^*(\omega_L) = (f_1^*\omega_L, ..., f_n^*\omega_L)$, are as close as possible to the goals,

$g = (g_1, g_2, ..., g_n)$. The deviation between $f^*(\omega_L)$ and g is defined as a deviation function $D(f(\omega_L)), g)$. The MOGP is now defined as below:

$$\begin{aligned} &\min_{\omega_L^* \epsilon W_L} D(f(\omega_L), g) \\ &\text{s.t} \\ &\omega_L^* \epsilon W_L = \left\{ \omega_L^* \epsilon R^L | Y \right\} \end{aligned} \tag{5.14}$$

The deviation function $D(f(\omega_L), g)$ is a maximum of deviations of individual goals

$$D(f(\omega_L), g) = \text{Max} \left\{ D_1(f_1(\omega_L), g_1), ..., D_n(f_n(\omega_L), g_n) \right\} \tag{5.15}$$

From equation 5.12 and 5.13, the min-max approach is applied to the GP problem.

$$\min_{\omega_L^* \epsilon W_L} \max \left\{ D_1(f_1(\omega_L), g_1), ..., D_n(f_n(\omega_L), g_n) \right\} \tag{5.16}$$

By introducing an auxiliary variable γ, equation 5.16, is now written as a linear program problem as

$$\begin{aligned} &\min_{\omega_L} \gamma \\ &\text{s.t} \\ &D_1(f_1(\omega_L), g_1) \leq \gamma \\ &D_2(f_2(\omega_L), g_2) \leq \gamma \\ &.... \\ &.... \\ &D_n(f_n(\omega_L), g_n) \leq \gamma \\ &\omega_L \epsilon Y \end{aligned} \tag{5.17}$$

Equation 5.17 is used to create the objective functions of an SR model. The objective functions are solved to obtain the weights of leaf softgoals that are in turn used in the goal analysis procedure.

5.4 Evaluation of Alternatives with Softgoal Optimisation

A code for optimal model-based goal programming was developed and implemented in Java Eclipse integrated with the IBM CPLEX optimisation tool (Figure 5.2). The reasons for this combination are: availability and prior experience made Java the preferred selection. Also, the IBM ILOG CPLEX optimizer is used to solve business mathematical models using powerful algorithms to obtain precise and logical decisions. Additionally, IBM ILOG CPLEX has a modeling layer called Concert that enables interfacing with Java, C++ and C# languages.

The input to the CPLEX optimizer was a set of objective functions for a given i^* framework and the outputs were the weights of the leaf softgoals of the given model. These weights are in turn used in the goal analysis to find the alternatives that maximise the top softgoals for the given i^* model. The MOGP model for an i^* framework was evaluated using the goal models from the existing RE literature: Youth Counseling (Horkoff and Yu, 2009), Meeting Scheduler System (Letier and Van Lamsweerde, 2004), London Ambulance System (You, 2004) and Telemedicine (Yu, 2001). To demonstrate this approach, the adapted Youth Counseling System and Telemedicine goal models are used in this chapter.

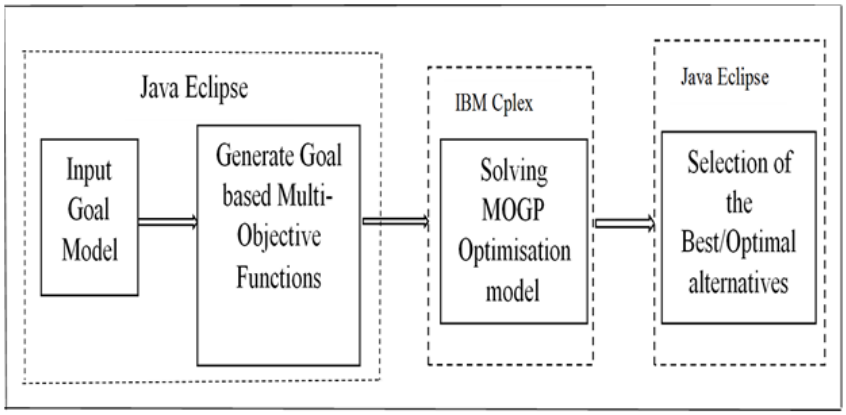

Figure 5.2: Scheme of the Multi-Objective Goal Programming Softgoal Optimisation and alternative selection

5.4.1 Deriving Objective Functions for the Actors

The following illustrates the derivation of multi-objective functions in terms of goal programming for the Youth Counseling case study (Figure 5.3). The alternatives in all three actors are

- Kids Use CyberCafe/Portal/Chat Room
- Kids Use TextMessaging

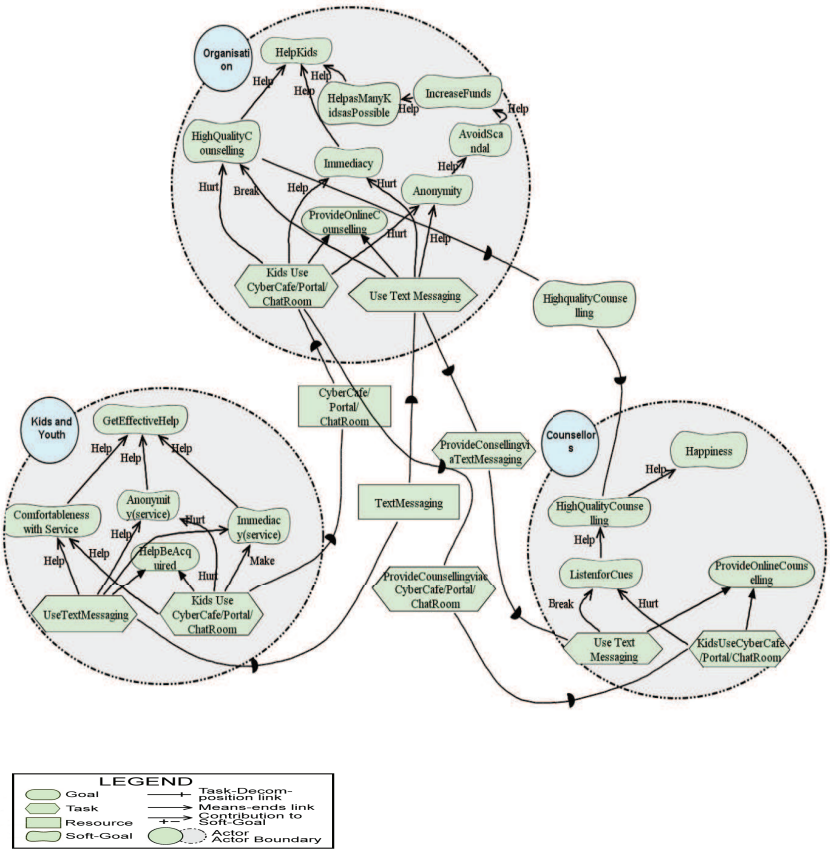

Figure 5.3: SR Model for Youth Counseling (adapted from Horkoff and Yu (2009))

In this case, the problem is to select an alternative that achieves maximum satisfactions for the top softgoals, *GetEffectiveHelp*, *Happiness* and *HelpKids* of actors, *Kids and Youth*, *Counsellor* and *Organisation* respectively. In Chapter 3, we have proposed a quantitative analysis approach that can be applied to such problems. In this analysis, the weights for the leaf softgoals which are assigned by the analyst, are subjective to the analyst. To avoid this subjective preference,

the weights are obtained by the MOGP optimisation model. These weights are in turn used in the analysis to select an alternative that yields maximum satisfaction of the top softgoals. Now by considering the first actor, *Kids and Youth*:
Score of the top softgoal, *GetEffectiveHelp* for alternative, *UseTextMessaging* is

$$\begin{aligned} S_{GetEffectiveHelp} &= Help * S_{Comfortableness} + Help * S_{Anonymity} + Help * S_{Immediacy} \\ &= 0.64 * S_{Comfortableness} + 0.64 * S_{Anonymity} + 0.64 * S_{Immediacy} \\ &= 0.64 * [0.64 * \omega_1] + 0.64 * [0.64 * \omega_2] + 0.64 * [0.16 * \omega_3] \\ &= 0.4096 * \omega_1 + 0.4096 * \omega_2 + 0.1024 * \omega_3 \end{aligned}$$

Note: For simplicity of calculation the defuzzified values of impact are used in the objective function.

The objective function for the alternative, *UseTextMessaging* in terms of the top softgoal, *GetEffectiveHelp* is

$$F_{Text}(\omega) = \max \{0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{31}\}$$

Similarly, the Score of the top softgoal, *GetEffectiveHelp* for alternative, *KidsUseCyberCafe* is

$$\begin{aligned} S_{GetEffectiveHelp} &= Help * S_{Comfortableness} + Help * S_{Anonymity} + Help * S_{Immediacy} \\ &= 0.64 * S_{Comfortableness} + 0.64 * S_{Anonymity} + 0.64 * S_{Immediacy} \\ &= 0.64 * [0.64 * \omega_1] + 0.64 * [0.16 * \omega_2] + 0.64 * [0.8 * \omega_3] \\ &= 0.4096 * \omega_1 + 0.1024 * \omega_2 + 0.512 * \omega_3 \end{aligned}$$

The objective function for the alternative, *KidsUseCyberCafe* in terms of the top softgoal, *GetEffectiveHelp* is given below:

$$F_{Cybercafe}(\omega) = \max \{0.4096 * \omega_{11} + 0.1024 * \omega_{21} + 0.512 * \omega_{31}\}$$

Similarly, the objective functions for the other two actors are obtained. For the actor, *Organisation*, the objective functions are

$$\begin{aligned} F_{Text}(\omega) &= \max \{0 * \omega_{12} + 0.1024 * \omega_{22} + 0.1074 * \omega_{32}\} \\ F_{Cybercafe}(\omega) &= \max \{0.4096 * \omega_{12} + 0.1024 * \omega_{22} + 0.2685 * \omega_{32}\} \end{aligned}$$

For the actor, *Counsellor*, the objective functions are

$$F_{Text}(\omega) = \max\{0 * \omega_{13}\}$$
$$F_{Cybercafe}(\omega) = \max\{0.0655 * \omega_{13}\}$$

Since, the alternatives are the same for all three actors, we have cumulative objective functions as follows:

$$F_{Text}(\omega) = \max\{0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{31} + 0 * \omega_{12} + 0.1024 * \omega_{22} + 0.1074 * \omega_{32} + 0 * \omega_{13}\}$$
$$F_{Cybercafe}(\omega) = \max\{0.4096 * \omega_{11} + 0.1024 * \omega_{21} + 0.512 * \omega_{31} + 0.4096 * \omega_{12} + 0.1024 * \omega_{22} + 0.2685 * \omega_{32} + 0.0655 * \omega_{13}\}$$
$$\text{Subject to}$$
$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 3 \text{ and } k = 1 \text{ to } 3 \tag{5.18}$$

Now, by considering the Telemedicine (Figure 5.4) goal model, the objective functions are obtained in a similar way. The alternatives are the same for the two actors. The two alternatives of the two actors are *PatientCenteredCare* and *ProviderCenteredCare*. The objective functions of the two alternatives are given as below:

$$F_{PatientCenteredCare}(\omega) = \max\{0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{12} + 0.1024 * \omega_{22}\}$$
$$F_{ProviderCenteredCare}(\omega) = \max\{0.1024 * \omega_{11} + 0.512 * \omega_{21} + 0.4096 * \omega_{12} + 0.4096 * \omega_{22}\}$$
$$\text{Subject to}$$
$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 2 \text{ and } k = 1 \text{ to } 2 \tag{5.19}$$

5.4.2 Obtaining Weights by MOGP

For the Youth counseling case study, let us consider the goals for each objective function to be equal to

$$F_{Text}(\omega) = 80 \text{ and } F_{Cybercafe}(\omega) = 90$$

The optimisation problem according to equation 5.17 with the auxiliary variable γ is :

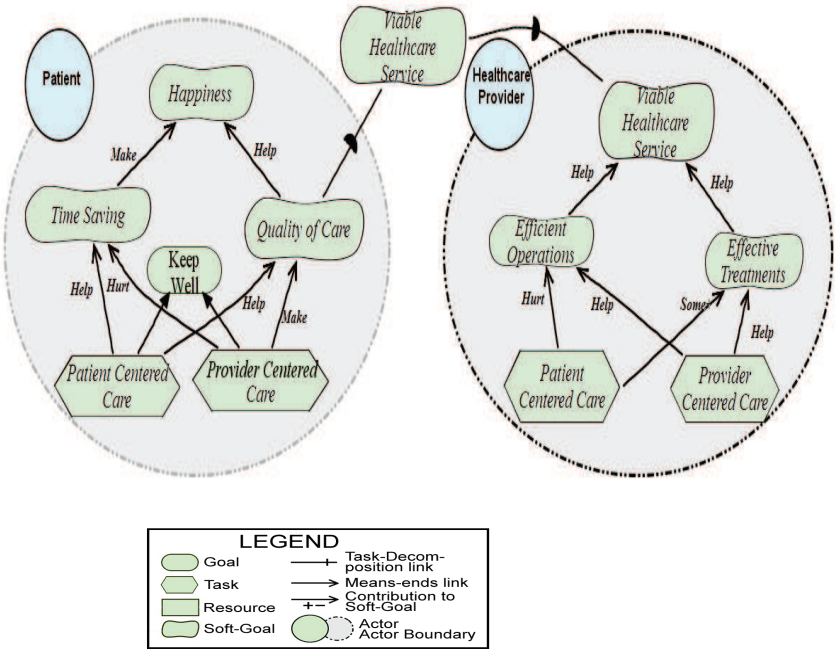

Figure 5.4: An Strategic Rationale Model: Telemedicine system (Adapted from Yu (2001))

$$\min_{\omega_L} \gamma$$

s.t

$$0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{31} + 0 * \omega_{12} + 0.1024 *$$
$$\omega_{22} + 0.1074 * \omega_{32} + 0 * \omega_{13} - 80 \leq \gamma$$
$$0.4096 * \omega_{11} + 0.1024 * \omega_{21} + 0.512 * \omega_{31} + 0.4096 * \omega_{12}$$
$$+0.1024 * \omega_{22} + 0.2685 * \omega_{32} + 0.0655 * \omega_{13} - 90 \leq \gamma$$
$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 3 \text{ and } k = 1 \text{ to } 3$$

For the Telemedicine goal model, by setting the goals to 70 and 80, the optimisation problem with the auxiliary variable γ is :

$$\min_{\omega_L} \gamma$$

$$\text{s.t}$$

$$0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{12} + 0.1024 * \omega_{22} - 70 \leq \gamma$$

$$0.1024 * \omega_{11} + 0.512 * \omega_{21} + 0.4096 * \omega_{12} + 0.4096 * \omega_{22} - 80 \leq \gamma$$

$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 2 \text{ and } k = 1 \text{ to } 2$$

The above multi-objective functions are solved by a programming code in the IBM CPLEX tool to find the weights of the leaf softgoals and the weights obtained are shown in Tables 5.1 and 5.2 for Youth Counseling and Telemedicine case studies respectively.

Table 5.1: Optimal weights for the Kids Youth Counseling case study

Actor	Leaf Softgoals	Weight
Kids and Youth	Comfortableness with service	1
	Anonymity	0.65
	Immediacy	0.10
Organisation	HighQualityCounseling	0.57
	Immediacy	1
	Anonymity	0.1
Counsellor	ListenforCues	0.1

Table 5.2: Optimal weights for the Telemedicine case study

Actor	Leaf Softgoals	Weight
Patient	Normal Life Style	1
	Quality of Care	0.67
HealthCare Provider	Efficient Operations	0.99
	Effective Treatments	0.1

5.4.3 Evaluation of the Approach

The weights computed by the optimisation goal model are now used in the quantitative analysis described in Chapter 3 to find the alternative option that provides maximum satisfaction of the top softgoals. For the inter-actor goal analysis, a code is implemented in the Java Eclipse environment. The reason for switching

from VC++ (as in Chapter 3) to Java Eclipse is a step towards tool development. This Java implementation allows the user to input graphical representation of the i^* goal model. The weights obtained by the CPLEX optimiser are given as weights of the leaf softgoals in the input for Java implementation. The top softgoals scores and the satisfaction comparison for the two goal models, Youth Counseling and Telemedicine, are given in Tables 5.3 and 5.4. From Table 5.3, it can be observed for Youth Counseling that the alternative Use *CyberCafe/Portal/ChatRoom* contributes 100% , 100% and 18% to the top softgoals *GetEffectiveHelp*, *Happiness* and *HelpKids* respectively. The second alternative *Use Text Messaging* of Youth Counseling contributes 99% ,40% and 4% to the top softgoals *GetEffectiveHelp*, *Happiness* and *HelpKids* respectively. By comparing the score values, the alternative *CyberCafe/Portal/ChatRoom* is selected for Youth Counseling case study.

From Table 5.4, it can be seen that the alternative *Patient Centred Care* contributes 100% and 37% to *Happiness* and *Viable Healthcare Service* respectively. The alternative *Provider Centred Care* contributes 98% and 92% to *Happiness* and *Viable Healthcare Service* respectively. On comparing the scores, the alternative *Patient Centred Care* is selected for actor *Patient* and the alternative *Provider Centred Care* is selected for actor *HealthCare Provider*.

Table 5.3: Scores of Top softgoals of the Kids Youth Counseling case study (* indicates selection)

Actor	Top Softgoals	Alternatives Score	
		Use Text Messaging	Use CyberCafe
Kids and Youth	GetEffectiveHelp	99%	100%*
Organisation	Happiness	40%	100%*
Counsellor	HelpKids	4%	18%*

Table 5.4: Scores of Top softgoals of the Telemedicine case study (* indicates selection)

Actor	Top Softgoals	Alternatives Score	
		PatientCenteredCare	ProviderCenteredCare
Patient	Happiness	100% *	98%
HealthCare Provider	Viable Healthcare service	37%	92%*

To evaluate the effectiveness of the proposed optimal model, the scores computed from the softgoals optimisation, are compared with the leaf softgoals op-

timisation model. By using the leaf-based optimal model, presented in Chapter 4, the objective functions are obtained for both case studies: Youth Counseling and Telemedicine. For these objective functions, the goal programming approach implemented using the CPLEX optimiser tool was applied to solve the weights of the leaf softgoals. The leaf softgoal based objective function for the case study Youth Counseling is given as

$$\begin{aligned}
&\min_{\omega_L} \gamma \\
&\text{s.t} \\
&0.64 * \omega_{11} + 0.64 * \omega_{21} + 0.16 * \omega_{31} + 0.0 * \omega_{12} + 0.16 * \omega_{22} + \\
&0.64 * \omega_{32} + 0.0 * \omega_{13} - 80 \leq \gamma \\
&0.64 * \omega_{11} + 0.16 * \omega_{21} + 0.8 * \omega_{31} + 0.16 * \omega_{12} + 0.64 * \omega_{22} + \\
&0.16 * \omega_{32} + 0.16 * \omega_{13} - 90 \leq \gamma \\
&0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 3 \text{ and } k = 1 \text{ to } 3
\end{aligned}$$

The leaf softgoal-based objective function for the Telemedicine case study is given as

$$\begin{aligned}
&\min_{\omega_L} \gamma \\
&\text{s.t} \\
&0.64 * \omega_{11} + 0.64 * \omega_{21} + 0.16 * \omega_{12} + 0.16 * \omega_{22} - 70 \leq \gamma \\
&0.16 * \omega_{11} + 0.81 * \omega_{21} + 0.64 * \omega_{12} + 0.64 * \omega_{22} - 80 \leq \gamma \\
&0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 2 \text{ and } k = 1 \text{ to } 2
\end{aligned}$$

The weights of the leaf softgoals were found using the CPLEX tool and are used in the analysis of the alternative selection. The score comparisons are shown in Tables 5.5 and 5.6 for Youth Counseling and Telemedicine respectively. The graphical representation of comparisons is shown in Figure 5.5 and Figure 5.6 . It can be inferred from these tables that the proposed softgoals based optimisation model gives a better scores compared to the leaf softgoals optimisation model for most of the top softgoals except for the alternative *Use TextMessage* of the softgoal *HelpKids*. Hence, the softgoal optimisation model outperforms the leaf softgoal optimisation model. The scores obtained from the proposed approach were not compared with the approach without optimisation, as the aim of the optimisation model is to avoid the subjective selection of weights for the leaf softgoals.

Table 5.5: Scores Comparison for the Youth Counseling case study

Actor	Top Softgoals	Use TextMessage		Use CyberCafe	
		With Softgoals Optimisation	With Leaf softgoals Optimisation	With Softgoals Optimisation	With Leaf softgoals Optimisation
Kids and Youth	GetEffectiveHelp	99%	98%	100%	100%
Organisation	Happiness	40%	25%	100%	25%
Counsellor	HelpKids	4%	7%	18%	13%

Table 5.6: Scores Comparison for the Telemedicine case study

Actor	Top Softgoals	Patient Centered Care		Provider Centered Care	
		With Softgoals Optimisation	With Leaf softgoals Optimisation	With Softgoals Optimisation	With Leaf softgoals Optimisation
Patient	Happiness	100%	83%	98%	41%
HealthCare Provider	Viable Healthcare Service	37%	36%	92%	89%

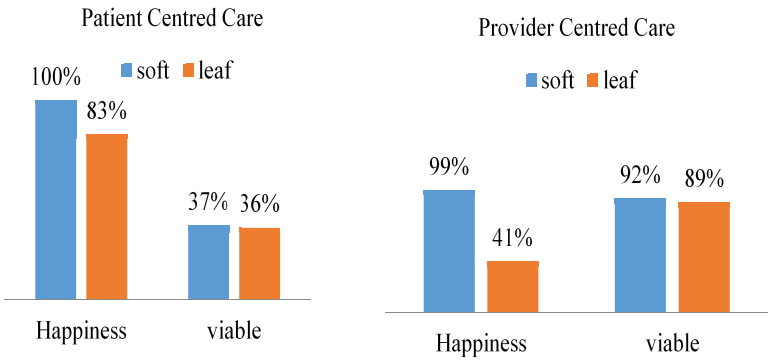

Figure 5.5: Telemedicine Comparison of Softgoals optimisation scores with Leaf softgoals Optimisation scores

5.5 Chapter Summary

Optimisation techniques have a significant part to play in the goal analysis of the goal models. This chapter has demonstrated how multi-objective optimisation can help decision making regarding alternative design choices. In particular, this chapter presented a technique for representing a given i^* framework as multi-objective optimisation models. These models are then solved by means of the goal programming approach, used to compute the weights of the leaf softgoals of the given i^* framework. These weights are in turn used in the goal analysis to select the alternative that maximises the satisfaction of the top softgoals. A

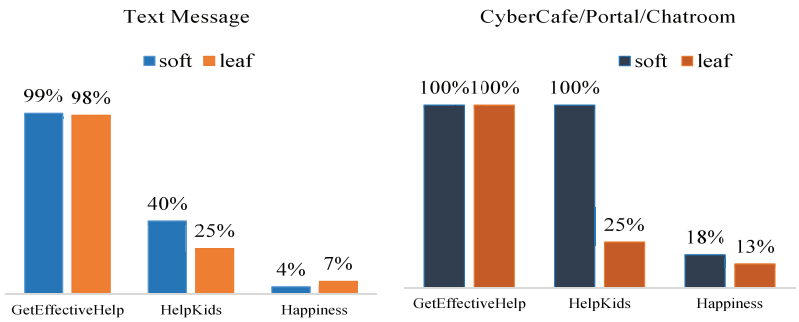

Figure 5.6: Youth Counseling Comparison of Softgoals optimisation scores with Leaf softgoals Optimisation scores

key feature of these models, as opposed to other optimisation models, is that objective functions are derived by considering all of the non-functional (softgoals) elements of the given system. The optimisation model is evaluated and the evaluation results are demonstrated using the case studies the Youth Counseling System and the Telemedicine System, taken from existing RE literature. The evaluation results showed that the softgoal-based goal programming optimisation is an improvement on the existing optimisation model.

In the next chapter, we discuss a method for handling the requirements of opposing objective functions.

Chapter 6

Softgoals with Opposing Nature

In Chapter 5, we presented a method based on Multi-Objective Goal Programming (MOGP) for obtaining the optimal model of the i^* framework. The objective functions generated from the optimal model are of maximising type (all objective functions of same type). Incredibly, most of the real-world business problems encounter simultaneous optimisation of many competitive objective functions. To illustrate, consider a system with two non-functional requirements such as profit and defect. Now the task of an analyst is to select an alternative design option that maximises profit and minimises defect. So, to deal with such problems, in this chapter we propose a game-theory-based goal analysis approach for the i^* framework. After a motivating example in Section 6.1, we present the challenges and motivation in Section 6.2. In Section 6.3, we explain the application of game theory concepts to solve multi-objective optimisation in the i^* framework. In Section 6.4, we present the illustration of game-theory-based goal analysis using i^* frameworks.

Some of the material in this chapter has been submitted to the Computer Journal (Accepted and to be published):

1. Chitra M Subramanian, Aneesh Krishna, and Arshinder Kaur (2017). Game Theory based Requirements Analysis in the i^* framework .

6.1 Motivating Example: Supply Chain Management System

In the current era, the use of software agents and their networks have greatly enhanced e-commerce planning and execution. One such example is the e-supply chain (Nair, 2013). Supply chain management (SCM) "*is the management of material and information flows both in and between facilities, such as vendors, manufacturing and assembly plants and distribution centres*" (Thomas and Griffin, 1996). Figure 6.1 depicts a simplified supply chain network. SCM is an area that is currently attracting tremendous attention from the business community.

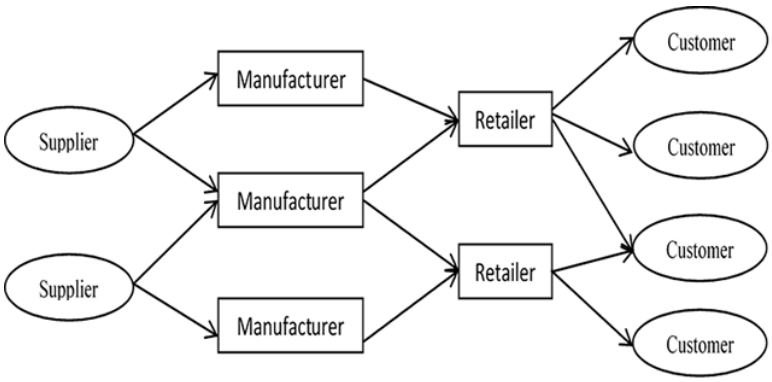

Figure 6.1: Supply Chain Network

The e-supply chain consists of a group or collection of intelligent software agents, each responsible for one or more activities. They also interact with each other in the planning and execution of their responsibilities. The e-supply chain uses software agents to perform various tasks with their local intelligence and problem-solving paradigm. Although the different entities in the supply chain have different sets of constraints and objectives, they operate interdependently to improve the performance of the system in terms of quality, cost minimisation and on-time delivery. Furthermore, the systems and their enclosing circumstances are dynamic. Hence, there is a need for systematic frameworks –models, methods, and tools that help with goal/task analysis and message communication.

Multi-agent system programming is appropriate for implementing the coordination involved in multiple autonomous or semi-autonomous agents where communications are conducted through messages (Bond and Gasser, 2014). Since supply chain management involves the integrity of decision makers, a multi-agent modeling framework is the best option to use in the design of a communica-

tion system which can be used between agents (such as manufactures, suppliers, retailers, customers).

The goal- and agent-oriented modeling framework, namely i^*, is an efficient way to model and analyse the relationships among the strategic entities in a social network (Yu, 2011), which includes human organisations and other forms of social structures. The i^* framework has also been used in business process modeling and redesign (Yu and Mylopoulos, 1995) and for software process modelling (Yu and Mylopoulos, 1994). Hence, in this research, the i^* framework is preferred as a modeling tool for the business process engineering of supply chain management.

Figure 6.2 shows an example of a Strategic Dependency (SD) model for global computer manufacturer and supplier supply chain. In this figure, the entities of the supply chain namely *Supplier, Manufacturer, Retailer* and *Customer* are denoted as actors (represented by circles) of the SD model. The dependency/communication between the entities of a supply chain can be represented by the dependency link of the SD model. The dependency can be one of any of the following types: goal dependency (represented by an oval), resource dependency (represented by a rectangle), task dependency (represented by a hexagon) or softgoal dependency (represented by a cloud). For instance, in Figure 6.2, the actor, *Supplier* has a softgoal dependency on the *Manufacturer*, namely *On time Delivery*. A simplified SD representation of the supply chain is given in Figure 6.3.

The top-down approach is used for identifying the goals of the actors. The overall objective/goal is to implement the supply chain services that can help various supply chain processes in the entire life cycle. The overall goal is decomposed into a set of sub-goals by using " how" and " what " questions. Each sub-goal is further broken down by asking HOW to achieve it and this is repeated until all the leaf goals are atomic. A goal may have different ways of implementing it. One of the Goal-Oriented Requirements Engineering (GORE) approaches is to select an alternative goal option that maximises the satisfaction of the softgoals (non-functional requirements) of an actor. The softgoals of an actor are identified and they are decomposed into sub-softgoals until the softgoals are atomic. The following illustrates the decomposition of the softgoals and the goals of an actor in the SCM case study.

The SCM model contains four actors: *Supplier, Manufacturer, Retailer* and *Customer*. For convenience and explanation purposes, only two actors namely *Manufacturer* and *Retailer*, are considered in this research work. Figure 6.4 contains a simplified example (with only 2 actors) of the i^* goal model created for

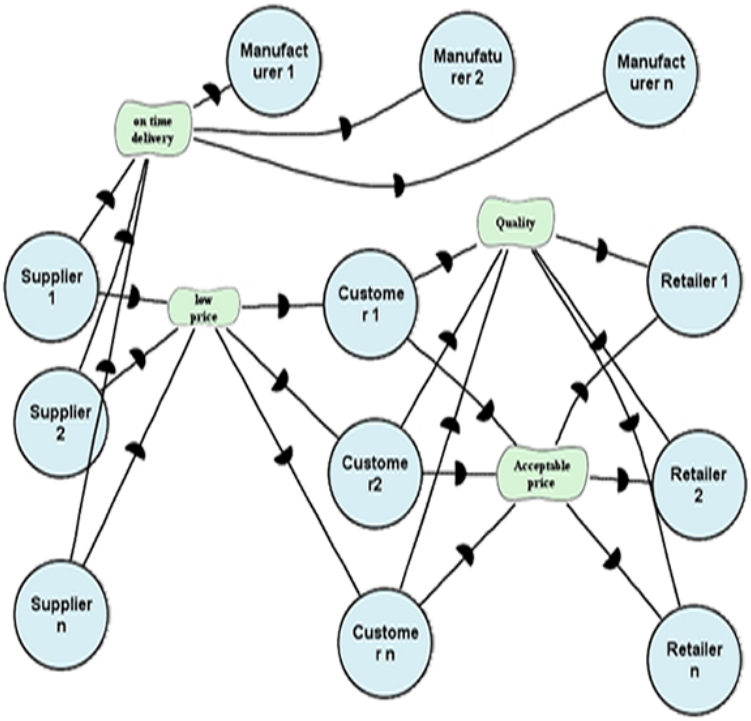

Figure 6.2: SD model for Supply Chain Network

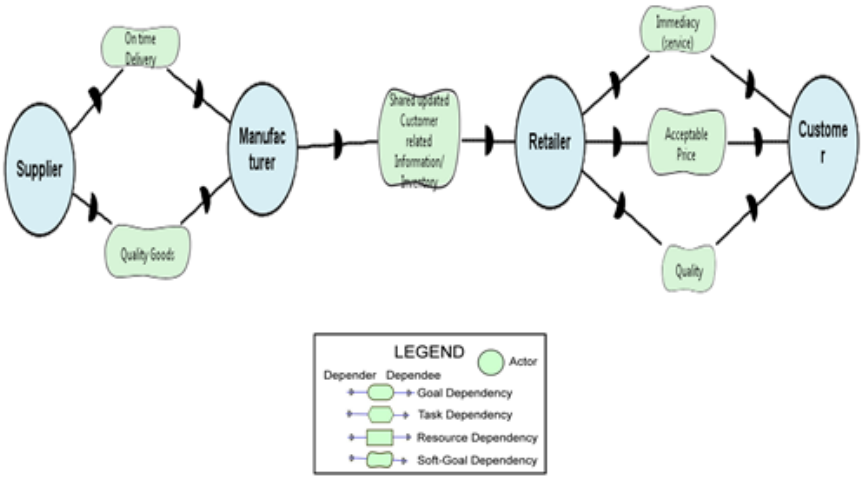

Figure 6.3: Simplified SD model for Supply Chain Network

this domain. For any organisation, there are many organisational goals such as

excellent service, quality of products, performance, transportation cost and so on. These are qualitative and harder to measure. These organisational goals are achieved by the operational goals. The organisational goals are called softgoals and the operational goals are called hard goals in the goal framework (Yu, 2011). The dependency of organisational goals on the operational goals is depicted by SR diagram of the i^* framework. For simplicity and illustration purpose, let us assume that for the actor *Manufacturer*, the two main softgoals (non-functional requirements) are *Performance* and *LogisticsCost*. These two softgoals in turn depend on the *OntimeDeliveryProcess* and *ProductionProcess*. The *Manufacturer* has a hard goal (functional requirement) of *Inventory* and explores two alternative ways of implementing this goal: *Traditional Inventory* and *Vendor Managed Inventory (VMI)*. These two alternatives contribute to the *Manufacturer* goal in different ways, which in turn contribute to each other. To illustrate, the contribution of the *VMI* to *ontimeDelivery* is *help* whereas *VMI* to *ProductionProcess* is *make*. The softgoal *OntimeDelivery* in turn contributes *some+* to the softgoal *Performance* and the softgoal *ProductionProcess* which in turn contributes *help* to the softgoal *Performance*. The actor, *Manufacturer* depends upon the actor *Retailer* for its inventory management. The actor, *Retailer* has its own goal to achieve. When selecting the best way to implement a goal, there arises a problem for the requirement analysts. These issues and challenges are explained in Section 6.2.

6.2 Challenges and Motivation

An analysis of this goal model (Figure 6.4) leads to an important question: The *manufacturer* has two high level (top) softgoals of opposing objective functions, namely *Performance* and *LogisticsCost*. The satisfaction percentage for the softgoal *Performance* should be maximised, whereas the softgoal *LogisticsCost* should be minimised in relation to the other goals and softgoals of the actor *manufacturer*. This situation leads to the question of which alternative the *Traditional* or the *VMI* is most effective for both the *Performance* & the *LogisticsCost* softgoals. The existing goal analysis approaches described in Chapters 3, 4 and 5 find an alternative based on maximising top softgoals. These approaches cannot be used to find a solution for an actor with one or more softgoals having opposing objective functions. There is a need for a new analysis procedure which helps to select an alternative for the actor with one or more top softgoals which have op-

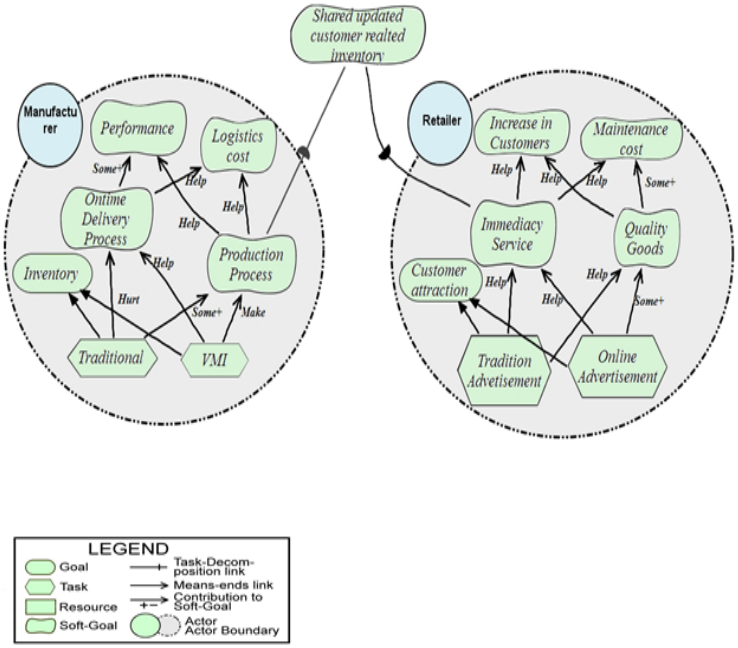

Figure 6.4: Simplified SR model for Supply Chain Network

posing objective functions. In this chapter, a novel approach is proposed to solve this type of problem. This approach is based on game theory, since game theory is used to find optimal solutions in competitive environments (Kelly, 2003). The proposed use of the new game-theory-based analysis procedure is illustrated using global computer manufacturer supplier supply chain case study for greater understanding in Section 6.4.

For elicitation and analysis of requirements in the early RE, many models focusing on stakeholder's goals have been proposed. These models indicate the fundamental motivation for the system, acquire non-functional criteria, and present the impact of high-level design alternatives on the goal achievement for diverse stakeholders through the dependencies. Several questions are raised on examination of the model: Which is the most effective alternative, and for whom? How to select an alternative, in case of softgoals with opposing nature? What information is missing from the model? Is the model adequately correct?

Even though answers to some of these types of questions can be found from the model, fast and consistent tracing effects become too complex for humans in the case of larger models. The model in Figure 6.4 is a simplified version of

a larger model; tracing the effects of alternative functionality becomes difficult when the model is large. Hence, there arises a need for a systematic analysis procedure which advises the analyst to evaluate alternative requirements, and explore the model. In such circumstances, to find an alternative option in the case of softgoals with opposing objective functions, the proposed game theory based approach can be applied.

6.3 Game Theory Model for Goal Modeling

In this section, game theory concepts and the application of game theory to the i^* framework for goal analysis is introduced.

6.3.1 Introduction to Game Theory

"*Game Theory (GT) is a scientific field dealing with the study and analysis of strategic, rational decision processes of individuals and their interactions in a (social) environment*" (Démuth, 2013). Game theory facilitates the anticipation and elucidation of systems whose behaviour and decisions use rational algorithms for problem solving. It is possible to determine the probability, weight and accuracy of certain aspects of the decision process. It is also applicable in fields where the outcome of an action is not dependent on pre-determined action, but depends upon a certain amount of choice. There is no boundary to the application of game theory. The application of game theory ranges from simple issues such as the determination of a favourable option that involves logic, and mathematics to solve complex issues such as social behaviour, economic behaviour, ethics, biological theory of choice and many others.

Game theory symbolizes an abstract model of decision making with a complete set of rules describing a game. An instance of the game is called a play. The intention of game theory is to find optimal solutions under the circumstances of conflict and cooperation with the assumption that players are rational and behave in their own interest (Kelly, 2003). The main elements of any game are its participating, autonomous decision makers, known as Players. A game must have two or more players. Even though a game can have a huge number of players, the number must be finite and known. The choices for each player must be more than one, since a player with only one choice has no strategy and hence cannot control the outcome of the game. There are three types of games: games of skill; games of chance and games of strategy. The games of strategy

are those involving two or more players and each player has partial control over the outcomes. Games of strategy can be sub-divided into two-player games and multi-player games. Within each of these two sub-divisions, there are three sub-categories depending on the way in which the pay-off functions are related to one another:

Cooperative games of strategy: Games of strategy, whether two-player or multi-player, in which the players' interests coincide. The following example taken from (Kelly, 2003) illustrates cooperative games of strategy. "*Two cyclist are going in opposite directions in a narrow path. They are due to collide and their intention is to avoid collision. Each has three strategies: move to right; move to left; or maintain direction. Obviously, the outcome depends on the decision of both cyclist and their interests coincide exactly. This is fully cooperative and the players need to signal their intentions to one other*".

Zero-sum games of strategy: Games in which the players' interests are conflicting (strictly competitive games). Chess and Poker are examples of Zero-sum games.

Mixed-motive games of strategy: Games in which the interests of players are neither fully conflicting nor fully coincident. The following example taken from (Kelly, 2003) illustrates mixed-motive games of strategy. "*The teachers' union at a school is threatening not to participate in parents' evenings unless management rescinds the redundancy notice of a long-serving colleague. Management refuses. The union now complicates the game by additionally threatening not to cooperate preparations for government inspection, if their demands are not met. Management has a choice between conceding and refusing, and which ever option it selects, the union has four choices: to resume both normal work practices; to participate in parents' evenings only; to participate in preparations for the inspection only; or not resume participation in either. Only one of the possible strategic combinations leads to a satisfactory outcome from the managements' point of view - management refusing to meet the union's demands not withstanding the resumption of normal work - although clearly some outcomes are worse than others. Both players (management and union) prefer some outcomes to others. For example, both would rather see resumption of participation in parents' evenings - since staff live in the community and enrollment depends on it - than not resume participation in either. So players' interest are simultaneously opposed and coincident*".

Of these types of games, the Zero-sum game suits our requirement of finding alternatives in situations where there are conflicting softgoals. Having given this

brief introduction to game theory, in the following section we explain the application of game theory, to aid in the decision making of selecting alternative options in the i^* goal model.

6.3.2 Game Theory for Multi-Objective Optimisation of i^* Goal Model

In game theory, if *G1* symbolizes one game with 't' players then feasible strategies for each player are expressed as $S_1, S_2, ...S_t$ with S_i is player i's strategy set. Let S_{ij} represent player i's j^{th} strategy and let pi be the player i's payoff. The problem of decision making in multi-objective optimisation (MOP) can be compared to the problem of decision making in a game. The goals of MOP can be considered to be players of the game. The design variable of MOP can be treated as the game strategies and constraints of MOP as constraints of game theory. The objective function value can be viewed as the gains of the game. Therefore, the game of 't' players can be expressed as

$$G1 = \{F_1, F_2, ., F_t, S_1, S_2, ...S_t\}$$

with $F_1, F_2, ., F_t$ as design goals and $S_i = \{A_j, ..., A_k\}$ for $i\epsilon 1$ to t as the strategy set for 't' players. Optimising the objective functions individually will identify the optimal value, which then forms the payoff matrix of the game.

6.3.2.1 Formalization

Let us consider an actor B of an i^* goal model having two top softgoals *TS1* and *TS2* with opposing objective functions. Top softgoals are softgoals that are high in the goal hierarchy. Let us assume that the *TS1* softgoal's objective function is to be maximised and the *TS2* softgoal's objective function is to be minimised. The actor B may have more than one alternative, among which the best alternative has to be selected. This is considered to be analogous to the Zero-sum game between two players. The top softgoals are treated as the players of the game; the alternatives are to be strategies of the players. Let the top softgoal *TS1* (player 1), have alternatives (strategies) $A_1 = \{A_j, ..., A_k\}$ and the top softgoal *TS2* (player 2) have alternatives (strategies) $A_2 = \{A_j, ..., A_k\}$ (with the assumption that both softgoals have the same set of alternatives). The payoff of a softgoal (player) is defined as a quantifiable value that the softgoal receives from arriving at a particular outcome (strategy). A matrix is used to represent the

possible outcome (strategy) for a two-person Zero-sum game with both players' possible strategies. This matrix is called a payoff matrix and the payoff matrix with both softgoals' (players') payoff represents the game. The optimal strategy is obtained by analysing the payoff matrix. We define a game for an actor of the i^* goal model as follows:

Definition 1 ***(i* game model and payoff matrix):*** A two-softgoal (person) Zero-sum game for an actor in an i^* goal model is given by a tuple $\mathbf{G} =< \mathbf{A_1}, \mathbf{A_2}, \mathbf{P} >$, where

(1) $\mathbf{A_1}$ is a nonempty set, the set of alternatives (strategies) of top softgoal 1**(TS1)**

(2) $\mathbf{A_2}$ is a nonempty set, the set of alternatives (strategies) of top softgoal 2 **(TS2)**

(3) $\mathbf{P}$ is a payoff matrix defined on $\mathbf{A_1} * \mathbf{A_2}$ with $\mathbf{p(a_1, a_2)}$ as a real number for every $a_1 \epsilon A_1$ and $a_2 \epsilon A_2$ and

(4) $\mathbf{p_1(a_1, a_2) + p_2(a_1, a_2) = 0}$

Softgoal 1 (Player 1) tries to maximise $p_1(a_1, a_2)$ and Softgoal 2 (player 2) tries to maximise $p_2(a_1, a_2) = -p_1(a_1, a_2)$. Subsequently, Softgoal 2 (player 2) tries to minimise $p_1(a_1, a_2)$. This procedure is called the ***maxmin strategy or optimal strategy*** and is defined as

Definition 2 ***(maxmin):*** For every finite two softgoal (person) Zero-sum game, the expected payoff to softgoal **TS1** when it uses strategy $\mathbf{A_p}$ and softgoal **TS2** using strategy $\mathbf{A_q}$ be denoted by $\mathbf{P_{pq}}$. If softgoal **TS1** plays strategy $\mathbf{A_p}$ ie., $\mathbf{min_q P_{pq}}$ then it has to maximise these minimum payoff. This is known as the ***maximin*** criterion.

By using the maxmin criterion, softgoal **TS1**'s minimum payoff is $\mathbf{t_l}$, the lower value of the game, where $\mathbf{t_l = max_p min_q P_{pq}}$. The value of the game is called the **saddle point** and is defined as

Definition 3 ***(Saddle Point):*** In a finite two softgoal Zero-sum game payoff matrix, if

(1) Softgoal **TS1**'s minimum payoff is $\mathbf{t_l}$, the lower value of the game, where $\mathbf{t_l = max_p min_q P_{pq}}$ and

(2) Softgoal **TS2**'s payoff is no more than $\mathbf{t_u}$, the upper value of the game,

where $\mathbf{t_u} = \mathbf{min_q max_p P_{pq}}$

(3) With $\mathbf{t_l} = \mathbf{t_u}$, then this point is called the ***saddle point*** and is the value of the game.

Definition 4 ***(Nash equilibrium)*****:** If $\mathbf{p}^*$ and $\mathbf{q}^*$ satisfies the ***maxmin*** criterion with $\mathbf{t_l} = \mathbf{t_u} = \mathbf{t}$, then t along with the optimal strategies $\mathbf{p}^*$ and $\mathbf{q}^*$ forms the solution of the game and is called ***Nash equilibrium***. If there is no saddle point in the game, then the game can be solved by an ***equalizing strategy (primal linear programming)***.

6.3.2.2 Strategy Space and Payoff Matrix Computation

The multi-objective optimisation function for an actor with 'n' number of alternatives for the first top softgoal (maximise) can be written as (for details of objective function generation, the readers are directed to Chapter 5).

$$
\begin{aligned}
f_{11}(\omega_L) = \bar{S}_{SGi1k} = \max \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc} \\
(\bar{C}_{(SGi1*LSGd1)} * \bar{C}_{(A1*LSGd1)k} * \omega_{LSGd1k}) \\
f_{12}(\omega_L) = \bar{S}_{SGi2k} = \max \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc} \\
(\bar{C}_{(SGi2*LSGd2)} * \bar{C}_{(A2*LSGd2)k} * \omega_{LSGd2k}) \\
... \\
... \\
f_{1n}(\omega_L) = \bar{S}_{SGink} = \max \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc} \\
(\bar{C}_{(SGin*LSGdn)} * \bar{C}_{(An*LSGdn)k} * \omega_{LSGdnk}) \\
\text{such that } 0 \leq \omega_{LSGd} \leq 100, \text{ for } d = 1 \text{ to } nc
\end{aligned}
\tag{6.1}
$$

Similarly, the multi-objective functions for the second softgoal with a minimising nature can be written as

$$
\begin{aligned}
f_{21}(\omega_L) = \bar{S}_{SGi1k} = \min \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc} \\
(\bar{C}_{(SGi1*LSGd1)} * \bar{C}_{(A1*LSGd1)k} * \omega_{LSGd1k}) \\
f_{22}(\omega_L) = \bar{S}_{SGi2k} = \min \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc} \\
(\bar{C}_{(SGi2*LSGd2)} * \bar{C}_{(A2*LSGd2)k} * \omega_{LSGd2k}) \\
... \\
... \\
f_{2n}(\omega_L) = \bar{S}_{SGink} = \min \Pi_{l=1}^{t} \bar{C}_{SGijl} \Sigma_{i=1}^{m} \Sigma_{d=1}^{nc}
\end{aligned}
$$

$$(\bar{C}_{(SGin*LSGdn)} * \bar{C}_{(An*LSGdn)k} * \omega_{LSGdnk})$$
$$\text{such that } 0 \leq \omega_{LSGd} \leq 100, \text{ for } d = 1 \text{ to } nc \tag{6.2}$$

In general, the multiple-objective functions for the first softgoal in equation (6.1) can be written as:

$$\max[f_{11}(\omega_L), f_{12}(\omega_L), ..., f_{1n}(\omega_L)]$$
$$\text{with } \omega_L \epsilon Y \tag{6.3}$$

where $n > 1$ and Y is the set of defined constraints.

The multiple objective functions for the second softgoal in equation (6.2) can be written as:

$$\min[f_{21}(\omega_L), f_{22}(\omega_L), ..., f_{2n}(\omega_L)]$$
$$\text{with } \omega_L \epsilon Y \tag{6.4}$$

where $n > 1$ and Y is the set of defined constraints.

Optimise the objective functions in equations (6.3) and (6.4) individually to obtain the ideal solutions. The ideal solutions are symbolically shown $(x_{11}, x_{12}, ..., x_{1n}, x_{21}, x_{22}, ..., x_{2n})$ with $(x_{11}, x_{12}, ..., x_{1n})$ representing the solution to equation (6.3) and $(x_{21}, x_{22}, ..., x_{2n})$ representing the solution to equation (6.4). Now the payoff matrix (P) can be represented as

$$\mathbf{P} = \begin{array}{c} \\ A1 \\ \\ An \end{array} \begin{array}{c} \begin{array}{cccc} A1 & A2 & & An \end{array} \\ \begin{pmatrix} y_{11}(x_{11}, x_{21}) & y_{12}(x_{11}, x_{22}) & .... & y_{1n}(x_{11}, x_{2n}) \\ .... & .... & .... & .... \\ .... & .... & .... & .... \\ y_{n1}(x_{1n}, x_{21}) & y_{n2}(x_{1n}, x_{22}) & .... & y_{nn}(x_{1n}, x_{2n}) \end{pmatrix} \end{array}$$

where

$$y_{ij}(x_{ab}, x_{cd}) = f_{ij}(x_{ab}) + f_{ij}(x_{cd}) \text{ for } i, j = 1 \text{ to } n \tag{6.5}$$
$$a, b, c, d = 1 \text{ to } n$$

The element y_{ij} is called a *saddle point* of the matrix P if

$$y_{ij} \leq y_{il} \ \forall l = 1, .., n \text{ and}$$
$$y_{ij} \geq y_{kj} \ \forall k = 1, .., n$$

That is, the element y_{ij} is simultaneously a minimum in its row and a maximum in its column. If *(i, j)* is a saddle point of a given game matrix, then the payoff that the row player gets in the saddle point is called the value of the game and corresponds to the *pure strategy Nash equilibrium* of the game.

6.3.2.3 Solving by Primal Linear Programming

If there is no saddle point, then the value of the game is obtained by equalizing strategy. The softgoal 1 follows the Maxmin strategy and the softgoal 2 follows Minmax strategy. If the softgoal 1 uses *A1*, $y_{11}(x_{11}, x_{21})$ proportion of times, *A2*, $y_{21}(x_{12}, x_{21})$ proportion of times, ... and *An*, $y_{n1}(x_{1n}, x_{21})$ proportion of times with softgoal 2 consistently having *A1*, then player 1's gain will be

$$y_{11}(x_{11}, x_{21})A1 + y_{21}(x_{12}, x_{21})A2 + .. + y_{n1}(x_{1n}, x_{21})An \tag{6.6}$$

If softgoal 2 consistently uses *A2*, then the softgoal 1's gain will be

$$y_{12}(x_{11}, x_{22})A1 + y_{22}(x_{12}, x_{22})A2 + .. + y_{n2}(x_{1n}, x_{22})An \tag{6.7}$$

and so on and finally with softgoal 2 consistently using *An*, then the softgoal 1's gain will be

$$y_{1n}(x_{11}, x_{2n})A1 + y_{2n}(x_{12}, x_{2n})A2 + .. + y_{nn}(x_{1n}, x_{2n})An \tag{6.8}$$

Softgoal 1 knows that it has to maximise the minimum that it will obtain. Since it is linear, the minimum that softgoal 1 will obtain lie in equation (6.6) or (6.7) or (6.8) or it is the point of intersection of these equations. Hence, the softgoal 1's formulation is given as:

$$\begin{aligned}
&\max v \\
&\text{Subject to} \\
&\qquad v \le y_{11}(x_{11}, x_{21})A1 + y_{21}(x_{12}, x_{21})A2 + ...+ \\
&\qquad\qquad y_{n1}(x_{1n}, x_{21})An \\
&\qquad v \le y_{12}(x_{11}, x_{22})A1 + y_{22}(x_{12}, x_{22})A2 + ...+ \\
&\qquad\qquad y_{n2}(x_{1n}, x_{22})An \\
&\qquad\qquad .... \\
&\qquad\qquad .... \\
&\qquad v \le y_{1n}(x_{11}, x_{2n})A1 + y_{2n}(x_{12}, x_{2n})A2 + ...+ \\
&\qquad\qquad y_{nn}(x_{1n}, x_{2n})An \\
&\qquad\qquad \sum_{i=1}^{n} Ai = 1; \\
&\qquad\qquad Ai \ge 0 \text{ for } i = 1 \text{ to } n
\end{aligned} \tag{6.9}$$

Where Ai represent strategy or proportion of times using a particular strategy.

Similarly, for softgoal 2 following the minmax strategy, the linear formulation is given by:

$$\min u$$

Subject to

$$\begin{aligned} u \geq y_{11}(x_{11}, x_{21})A1 + y_{12}(x_{11}, x_{21})A2 + ...+ \\ y_{1n}(x_{11}, x_{21})An \\ u \geq y_{21}(x_{12}, x_{21})A1 + y_{22}(x_{12}, x_{22})A2 + ...+ \\ y_{2n}(x_{12}, x_{2n})An \\ .... \\ .... \\ u \geq y_{n1}(x_{1n}, x_{21})A1 + y_{n2}(x_{1n}, x_{22})A2 + ...+ \\ y_{nn}(x_{1n}, x_{2n})An \end{aligned}$$

$$\begin{aligned} \sum_{i=1}^{n} Ai = 1; \\ Ai \geq 0 \text{ for } i = 1 \text{ to } n \end{aligned} \tag{6.10}$$

Since equation (6.10) is a dual of equation (6.9), the solution to the game is found by either solving equation (6.9) or equation (6.10). In the following section, the duality of equations (6.9) and (6.10) will be demonstrated.

6.3.2.4 Proof of Duality

Consider softgoal 1's formulation given by equation (6.9) (which is known as primal) and we will try to find its dual. In order to find the dual of softgoal 1's primal, rewrite the softgoal 1's formulation given in equation (6.9) as follows:

$$\max v$$

Subject to

$$\begin{aligned} v - y_{11}(x_{11}, x_{21})A1 - y_{21}(x_{12}, x_{21})A2 - ...- \\ y_{n1}(x_{1n}, x_{21})An \leq 0 \\ v - y_{12}(x_{11}, x_{22})A1 - y_{22}(x_{12}, x_{22})A2 - ...- \\ y_{n2}(x_{1n}, x_{22})An \leq 0 \\ .... \\ .... \\ v - y_{1n}(x_{11}, x_{2n})A1 - y_{2n}(x_{12}, x_{2n})A2 - ...- \\ y_{nn}(x_{1n}, x_{2n})An \leq 0 \end{aligned}$$

$$\begin{aligned} A1 + A2 + ... + An = 1; \\ A1, A2, ..., An \geq 0 \end{aligned} \tag{6.11}$$

The primal has $n + 1$ constraints; hence, the dual will have $n + 1$ variables. Let us define dual variables as $Z1, Z2, Zn, w$ for the constraints as follows:

$$\max v$$

Subject to

$$\begin{aligned} Z1 : v - y_{11}(x_{11}, x_{21})A1 - y_{21}(x_{12}, x_{21})A2 - ...- \\ y_{n1}(x_{1n}, x_{21})An \leq 0 \\ Z2 : v - y_{12}(x_{11}, x_{22})A1 - y_{22}(x_{12}, x_{22})A2 - ...- \\ y_{n2}(x_{1n}, x_{22})An \leq 0 \\ .... \\ .... \\ Zn : v - y_{1n}(x_{11}, x_{2n})A1 - y_{2n}(x_{12}, x_{2n})A2 - ...- \\ y_{nn}(x_{1n}, x_{2n})An \leq 0 \end{aligned}$$

$$\begin{aligned} w : A1 + A2 + ... + An = 1; \\ A1, A2, ..., An \geq 0 \end{aligned} \tag{6.12}$$

The dual will be given as:

$$\min w$$

Subject to

$$\begin{aligned} Z1 + Z2 + ... + Zn &= 1 \\ -y_{11}(x_{11}, x_{21})Z1 - y_{12}(x_{11}, x_{22})Z2 - ...- & \\ y_{1n}(x_{11}, x_{2n})Zn &\geq w \\ -y_{21}(x_{12}, x_{21})Z1 - y_{22}(x_{12}, x_{22})Z2 - ...- & \\ y_{2n}(x_{12}, x_{2n})Zn &\geq w \\ .... & \\ .... & \\ -y_{n1}(x_{1n}, x_{21})Z1 - y_{n2}(x_{1n}, x_{22})Z2 - ...- & \\ y_{nn}(x_{1n}, x_{2n})Zn &\geq w \\ Z1 + Z2 + ... + Zn &\geq 0 \end{aligned} \tag{6.13}$$

Now rewriting equation (6.13) as

$$\min w$$

Subject to

$$\begin{aligned} Z1 + Z2 + ... + Zn &= 1 \\ w \geq y_{11}(x_{11}, x_{21})Z1 + y_{12}(x_{11}, x_{22})Z2 + ...+ & \\ y_{1n}(x_{11}, x_{2n})Zn & \\ w \geq y_{21}(x_{12}, x_{21})Z1 + y_{22}(x_{12}, x_{22})Z2 + ...+ & \\ y_{2n}(x_{12}, x_{2n})Zn & \\ .... & \\ .... & \\ w \geq y_{n1}(x_{1n}, x_{21})Z1 + y_{n2}(x_{1n}, x_{22})Z2 + ...+ & \\ y_{nn}(x_{1n}, x_{2n})Zn & \\ Z1 + Z2 + ... + Zn &\geq 0 \end{aligned} \tag{6.14}$$

Comparing the equation (6.14) with softgoal 2's formulation equation (6.10), we can say that the dual of softgoal 1's formulation is the softgoal 2's formulation. There is a primal dual relationship between softgoal 1 and softgoal 2. If the solution for one of them is found, then the duality principle is applied to solve the other. Hence, by solving either softgoal 1's linear formulation or softgoal 2's linear formulation, the proportions for the strategies (in our case the alternatives) can be found. The strategy (alternative) with high proportion (probability) is selected.

6.4 Illustration of Game-Theory-Based Goal Analysis using *i** Framework

In order to evaluate the optimisation model using game theory, a tool was created using Java Eclipse environment integrated with the IBM ILOG CPLEX optimisation tool. The reasons for this combination are: availability and prior experience made Java the preferred selection. Also, the IBM ILOG CPLEX optimizer is used to solve business mathematical models using powerful algorithms to obtain precise and logical decisions. Additionally, IBM ILOG CPLEX has a modeling layer called Concert that facilitates interfacing with Java, C++ and C# languages. The tool is analogous to that used in Chapter 5; however, the current tool includes solving both the maximisation and minimisation objective functions by using the Zero-sum game-theory-based approach. The simulation takes an input graph in which the top softgoals are associated with a gauge variable that shows the type of optimisation to be performed. Based on the gauge variable of each top softgoal, the objective functions are generated. Once the objective functions have been generated, they are solved individually to obtain the objective function values. A payoff matrix is created using these function values with the alternatives as strategies of the game. Then, using the game-theory-based approach, the payoff matrix is solved so as to identify the optimal alternative. To demonstrate this approach, we return to the Supply Chain Management case study.

Figure 6.4 presents a supply chain model with two actors that are considerably simplified, but supports some types of reasoning namely identification and the exploration of alternatives. The reasoning behind one possible arrangement is illustrated namely, *Manufacturer*. The main softgoals of the actor *Manufacturer* are *Performance* and *Logistics cost*, which depend upon the softgoals *Ontime Delivery Process* and *Production Process*. The actor, *Manufacturer*, has the *Inventory* goal, representing the *Manufacturer* aim of maintaining the inventory management system. The goal, *Inventory*, has two ways of being implemented and thus is ***OR*** decomposed into two tasks known as *Traditional* and *Vendor Managed Inventory (VMI)*. *VMI* needs strong information technology support to connect vendor and manufacturer in order to implement *VMI* successfully. The goal, *Inventory*, represents a decision point. The selection of a task for this goal influences the satisfaction levels of the non-functional goals (softgoals *Performance* and *Logistics cost*) for the actor, *Manufacturer*.

6.4.1 Generation of the Objective Functions using Gauge Variables

The multi-objective functions generated for the supply chain management case study (Figure 6.4) are illustrated as follows. The alternative tasks of the actor, *Manufacturer* are

- Traditional
- Vendor Managed Inventory(VMI)

In this case, the problem is to select an alternative that maximises the satisfaction of top softgoal *Performance* and minimises the satisfaction of top softgoal *Logistics cost*. The top softgoals are associated with the gauge variable ($\hat{g}$) whose value is 1 (one) if it has to be maximised, and zero if it has to be minimised. For the actor, *Manufacturer*, let us assume that $\hat{g} = 1$ for top softgoal *Performance* and $\hat{g} = 0$ for top softgoal *logistics cost*.

The score of the top softgoal *Performance* for the alternative *Traditional* is given as follows:

$$\begin{aligned} S_{Performance} &= (Some+) * (S_{OntimeDeleiveryProcess}) + \\ &\quad (Help) * (S_{ProductionProcess}) \\ &= 0.48 * S_{OntimeDeleiveryProcess} + 0.64 * S_{ProductionProcess} \\ &= 0.64 * [0.16 * \omega 1] + 0.64 * [0.48 * \omega 2] \\ &= 0.0768 * \omega 1 + 0.3072 * \omega 2 \end{aligned}$$

Since $\hat{g} = 1$ for top softgoal *Performance*, the objective function for the *Traditional* alternative in terms of the top softgoal *Performance* is given below

$$F_{Performance(\omega)} = \max\{0.0768 * \omega 1 + 0.3072 * \omega 2\}$$

Similarly, the objective function for the *VMI* alternative in terms of the top softgoal *Performance* is given as

$$F_{Performance(\omega)} = \max\{0.3072 * \omega 1 + 0.512 * \omega 2\}$$

Correspondingly, the objective functions for the top softgoal *Logistics Cost* with $\hat{g} = 0$ are obtained as

$$F_{Logisticscosts(\omega)} = \min\{0.1024 * \omega 1 + 0.3072 * \omega 2\}$$
$$F_{Logisticscosts(\omega)} = \min\{0.4096 * \omega 1 + 0.512 * \omega 2\}$$

The objective functions for the actor, *Manufacturer*, including both top soft-goals *Performance* and *Logistics Cost* are given as follows:

$$F11 = F_{Performance(\omega)} = \max\{0.0768 * \omega 1 + 0.3072 * \omega 2\}$$
$$F12 = F_{Performance(\omega)} = \max\{0.3072 * \omega 1 + 0.512 * \omega 2\}$$
$$F21 = F_{Logisticscosts(\omega)} = \min\{0.1024 * \omega 1 + 0.3072 * \omega 2\}$$
$$F22 = F_{Logisticscosts(\omega)} = \min\{0.4096 * \omega 1 + 0.512 * \omega 2\}$$

Similarly, the objective functions obtained for the actor, *Retailer*, are

$$F11 = F_{IncreaseinCustomers(\omega)} = \max\{0.4096 * \omega 1 + 0.4096 * \omega 2\}$$
$$F12 = F_{IncreaseinCustomers(\omega)} = \max\{0.4096 * \omega 1 + 0.3072 * \omega 2\}$$
$$F21 = F_{MaintenanceCost(\omega)} = \min\{0.4096 * \omega 1 + 0.3072 * \omega 2\}$$
$$F22 = F_{MaintenanceCost(\omega)} = \min\{0.4096 * \omega 1 + 0.2304 * \omega 2\}$$

The solutions to these objective functions are obtained by invoking the IBM ILOG CPLEX; the obtained function values are given in Table 6.1 as a ready reference.

Table 6.1: Objective functions values for SCM goal model

Objective Function	Manufacturer	Retailer
F11	38.4	81.9
F12	81.92	71.68
F21	4.096	7.16
F22	9.216	6.4

6.4.2 Pay-off Matrix and Alternative Selection

The pay-off matrix for the actor, *Manufacturer*, is created using equation 6.5, as follows:

$$\mathbf{P} = \begin{array}{c} \\ Traditional \\ \\ VMI \end{array} \begin{array}{c} \begin{array}{cc} Traditional & VMI \end{array} \\ \begin{pmatrix} 42.496 & 47.616 \\ 86.016 & 91.136 \end{pmatrix} \end{array}$$

$42.496T + 86.016V = 47.616T + 91.136V$[T represents *Traditional* and V represents *VMI*]

Primal linear programming has been used to solve the above equation and the solutions obtained are $T = -5$ and $V = 6$.

Similarly, the pay-off matrix for the actor, *Retailer* is

$$\mathbf{P} = \begin{array}{c} \\ Traditional \\ \\ Online \end{array} \begin{array}{c} \begin{array}{cc} Traditional & Online \end{array} \\ \begin{pmatrix} 89 & 88 \\ 78.8 & 78.08 \end{pmatrix} \end{array}$$

$89T+78.8L = 88T+78.08L$[T represents *Traditional* and L represents *Online*]. The solutions for the actor *Retailer*, are $TraditionalAdvertisement = 1$ and $OnlineAdvertisement = 0$.

The tool outputs for the actor *Manufacturer* are provided in Figures 6.5 & 6.6. The graphical representation of the outputs are given in Figure 6.7. From the Figure 6.7, it can be seen that the alternative *VMI* has a higher probability (proportion) than the alternative *Traditional* and hence is selected for the actor *Manufacturer* by considering both the top softgoals *Performance* and *Logistics Cost*. Also for the actor *Retailer*, it can be seen that the alternative, *Traditional* is only given, as the value of alternative, *Online* is zero. Thus, the alternative *Traditional Advertisement* is selected for the actor *Retailer*. The game-theory-based goal analysis pseudo-code is given in **Algorithm** 3.

The validation of the proposed approach has been carried out by comparing results of the existing approach proposed in Chapter 3. In Chapter 3, we proposed an approach to find the satisfaction percentage of alternatives based on top softgoals. According to this proposal, the leaf softgoals are assigned percentage

Algorithm 3 Psuedo code for the Tool Implementation of Game theory based Goal Analysis of i^* Framework

Require: Parameter G, such that $G = \{G_1, G_2, ..., G_n\}$ is a collection of directed graphs. Each graph G_i is a quadruple $\{T, LSG, SG, TSG\}$ comprises of a set of tasks T, a set of Leaf Softgoals LSG, a set of intermediate softgoals SG and a set of top softgoals TSG with each top softgoal associated with a gauge variable $\hat{g}$.

MAIN MODULE : optimal goal analysis

for all $G_i \epsilon G$ **do**
  for all task $t \epsilon T$ **do**
    for all top softgoals $tsg \epsilon TSG$ **do**
      if $\hat{g}$ is 0 **then**
        Generate minimisation objective function
      else
        Generate maximisation objective function
      end if
    end for
  end for
end for
Let $F1 \leftarrow \min\{f_1, f_2, ..., f_n\}$
Let $F2 \leftarrow \max\{g_1, g_2, ..., g_n\}$
for all f_i in F1 **do**
  Let $x_i \leftarrow optim(f_i, \hat{g})$
end for
for all g_i in F2 **do**
  Let $y_i \leftarrow optim(g_i, \hat{g})$
end for
generate payoff matrix p using x_i and y_i as
for all task $t \epsilon T$ **do**
  $p_{ij} \leftarrow x_i + y_j$ //generate Primal equation
end for
solve the primal equation to obtain optimal solution
SUBMODULE : optim (F , G)
ASSERTION : solves the objective function to find the ideal value of the function

ALGORITHM

declare the variables.
Define the expression, the objective and the constraints based on G (if G=0 define minimisation function, else maximisation function)
if $\hat{g}$ is 0 **then**
    define minimisation function
else
    define maximisation function
end if
$W \leftarrow cplex.solve()$ //invoking CPLEX function
return W

weights (from 0 to 100) based on their relative importance given by the requirements analyst. The impacts of the goals or tasks on the softgoals namely *Make*, *Help*, *Some+*, *Some-*, *Hurt*, and *Break* are assigned fuzzy numbers (0.64, 0.80, 1), (0.48, 0.64, 0.80),(0.32,0.48,0.64), (0.16, 0.32, 0.48), (0, 0.16, 0.32) and (0, 0, 0.16) respectively. After this assignment, for each alternative option the scores of leaf softgoals are computed. These scores are then propagated backwards to compute the scores of other softgoals in the hierarchy. Once the top softgoals (goals that are at the top in the hierarchy) scores are computed, they are compared in order to select an alternative option that best satisfies the top softgoals. The drawback of the approach proposed in Chapter 3 is the subjective preference of weights assigned to the leaf softgoals by the analyst. The validition of the proposed game-theory-based approach has been realised by comparing results obtained in Chapter 3 for different set of weights.

First, let us assume that the weights (in percentage) of the leaf softgoals *OntimeDeliveryProcess*, *ProductionProcess*, *ImmediacyService* and *QualityGoods* are 53%, 10%, 73% and 10% respectively. Using these weights for the actor *Manufacturer*, the satisfaction percentage of the top softgoals *Performance* and *LogisticsCost* for the alternative *Traditional* are found to be 21% and 25% respectively. The scores of the top softgoals for the second alternative *VMI* are 45% and 55% respectively. From these values, we can infer that the alternative *VMI* has the best satisfaction percentage and, hence, it is selected for the actor *Manufacturer*. Similarly, for the actor *Retailer*, the satisfaction percentage of the top softgoals *IncreaseinCustomers* and *MaintenanceCost* for the alternative *Traditional* are found to be 70% and 68% respectively. Also, the satisfaction percentage for the second alternative *Online* are 68% and 67% respectively. From

these values, we can infer that the alternative *Traditional* has the best satisfaction percentage and, hence, it is selected for the actor, *Retailer*. These results correspond with the results obtained by the proposed game-theory-based approach. Results obtained for the actors, *Manufacturer* and *Retailer*, with different sets of weights are provided in Tables 6.2 and 6.3 as a ready reference for readers.

Table 6.2: Satisfaction Percentage of Top Softgoals for the actor Manufacturer(SCM goal model)(* indicates options selected)

Weights of LSG		Satisfaction Percentage of TSG			
Ontime Delivery Process	Production Process	Performance		Logistics Cost	
		Traditional	VMI	Traditional	VMI
50	10	21%	45%*	25%	55% *
60	40	41%	81%*	46%	93%*
70	50	51%	98%*	56%	100%*
40	50	43%	79%*	46%	87%*
30	60	46%	83%*	49%	89%*

Table 6.3: Satisfaction Percentage of Top Softgoals for the actor Retailer(SCM goal model)(* indicates options selected)

Weights of LSG		Satisfaction Percentage of TSG			
Immediacy Service	Quality Goods	Increase in Customers		Maintenance Cost	
		Traditional	Online	Traditional	Online
73	10	70%*	68%	68%*	67%
70	60	100%*	97%	98%*	88%
70	50	100%*	91%	91%*	83%
30	55	72%*	61%	61%*	53%
40	80	100%*	84%	85%*	73%

The inference from Tables 6.2 and 6.3 is that for any set of weights, for the actor, *Manufacturer*, the alternative *VMI* option has the best satisfaction values than the alternative *Traditional* option and is therefore selected for the actor, *Manufacturer*. On the other hand, for the actor, *Retailer* the alternative option *Traditional* dominates over the alternative option *Online* and hence it is selected for the actor *Retailer*. These results correspond with the result obtained from the game-theory-based approach and thus validate the game theory approach results.

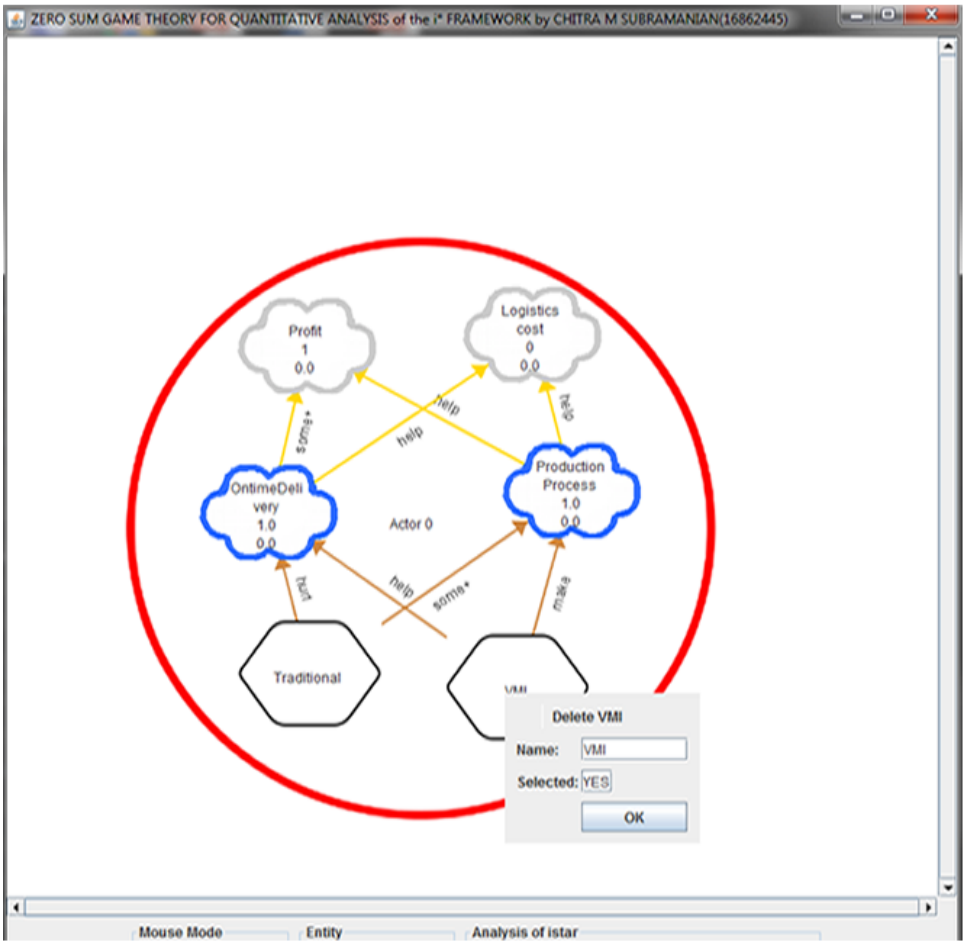

Figure 6.5: Tool Result for actor Manufacturer for alternative VMI

6.4.3 Evaluation of the Approach

To check the effectiveness and feasibility of the proposed game-theory-based approach, experiments were carried out using different case studies from the literature, namely Meeting Scheduler System (Van Lamsweerde, 2009), Kids Youth Counselling (Horkoff and Yu, 2016) and Telemedicine (Yu, 2001). The results of the Telemedicine case study are presented in this chapter.

Telemedicine system is the use of information technology and telecommunication to provide remote diagnosis and treatment for patients. The adapted Telemedicine (Figure 6.8) shows two actors: *Patient* and *Healthcare Provider*. For illustration and simplicity, let us assume that the two main non-functional requirements of the actor *Patient* are *Expense* of the treatment and *Happiness (satisfaction)* obtained from the remote treatment. These two softgoals in turn depend upon *Time Saving* and *Quality of Care*. There are two alternative ways of

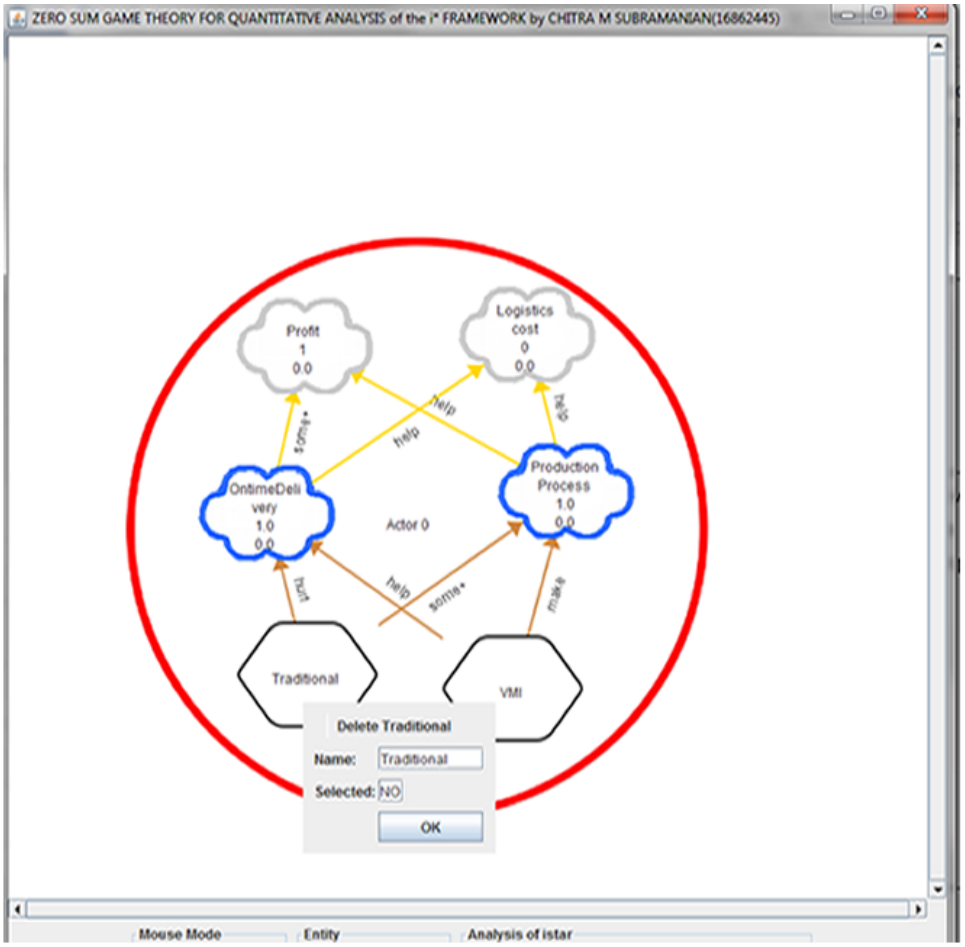

Figure 6.6: Tool Result for actor Manufacturer for alternative Traditional

obtaining treatment for the patient: *Patient Centered Care* or by *Provider Centered Care*. The *Patient* has to choose an alternative option such that his/her *Expense* is less and *Happiness* is more. Similarly, for the *Healthcare Provider*, the two main non-function requirements are *Viable Healthcare Service* and *Maintenance Cost*. These two softgoals are further decomposed into *Efficient Operations* and *Effective Treatments*. The two alternative ways of providing services are *Patient Centered Care* and *Provider Centered Care*. Here, the task is to select an alternative option that increases the *Healthcare service* and decreases *Maintenance Cost*.

The objective functions for the actor, *Patient*, for both the top softgoals *Expense* and *Happiness* are given as:

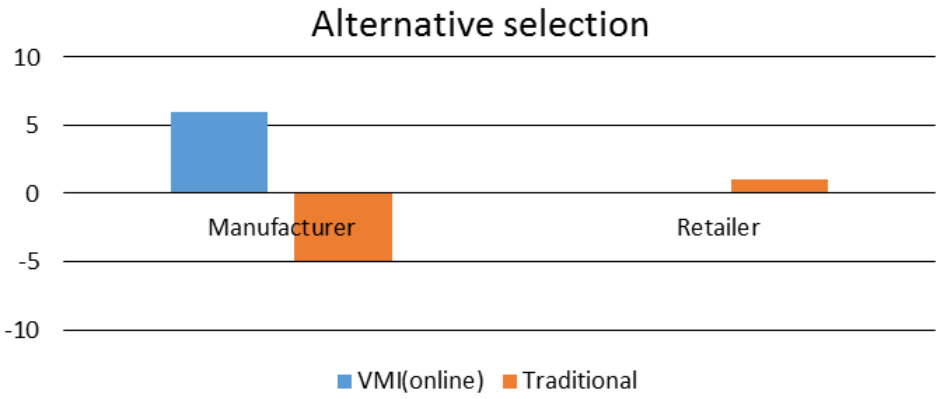

Figure 6.7: Graphical representation of proportional values of alternatives for SCM

$$F11 = F_{Expense(\omega)} = \min\{0.4096 * \omega1 + 0.4096 * \omega2\}$$
$$F12 = F_{Expense(\omega)} = \min\{0.1024 * \omega1 + 0.512 * \omega2\}$$
$$F21 = F_{Happiness(\omega)} = \max\{0.512 * \omega1 + 0.4096 * \omega2\}$$
$$F22 = F_{Happiness(\omega)} = \max\{0.128 * \omega1 + 0.512 * \omega2\}$$

Similarly, the objective functions for the actor, *Healthcare Provider*, for both the top softgoals *Viable Healthcare Service* and *Maintenance Cost* are given as:

$$F11 = F_{ViableHealthcareService(\omega)} = \max\{0.1024 * \omega1 + 0.3072 * \omega2\}$$
$$F12 = F_{ViableHealthcareService(\omega)} = \max\{0.4096 * \omega1 + 0.4096 * \omega2\}$$
$$F21 = F_{MaintenanceCost(\omega)} = \min\{0.1024 * \omega1 + 0.384 * \omega2\}$$
$$F22 = F_{MaintenanceCost(\omega)} = \min\{0.4096 * \omega1 + 0.512 * \omega2\}$$

The values of the objective functions are obtained by using the tool and are provided in Table 6.4 as a ready reference.

The pay-off matrix for the actor, *Patient*, is as follows:

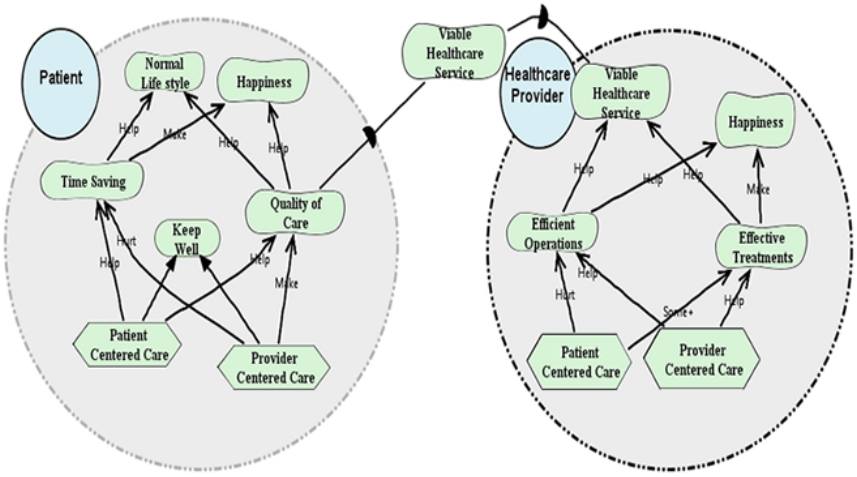

Figure 6.8: Simplified SR model for Telemedicine (adapted from Yu (2001))

Table 6.4: Objective functions values for Telemedicine goal model

Objective Function	Patient	Healthcare Provider
F11	8.19	40.96
F12	6.14	81.92
F21	92.16	4.86
F22	64	9.22

$$\mathbf{P} = \begin{array}{c} \\ Patient \\ Provider \end{array} \begin{array}{c} \begin{array}{cc} Patient & Provider \end{array} \\ \begin{pmatrix} 100 & 72 \\ 98 & 70 \end{pmatrix} \end{array}$$

$100Pa + 98Pr = 72Pa + 70Pr$ [Pa represents *Patient Centered Care* and Pr represents *Provider Centered Care*]

Solving by primal linear programming (by using equation 6.9), the solutions

are Pa=1 and Pr=0. The alternative *Patient Centered Care* has high probability and hence it is selected for the actor, *Patient*.

Similarly, the pay-off matrix for the actor, *Healthcare Provider*, is

$$\mathbf{P} = \begin{array}{c} \\ Patient \\ Provider \end{array} \begin{array}{c} \begin{array}{cc} Patient & Provider \end{array} \\ \begin{pmatrix} 45.8 & 50 \\ 86 & 91 \end{pmatrix} \end{array}$$

$45.8Pa + 86Pr = 50Pa + 91Pr$ [Pa represents *Patient Centered Care* and Pr represents *Provider Centered Care*]

Again solving by primal linear programming, the solutions are Pa=0 and Pr=1. The alternative *Provider Centered Care* has high probability and, hence, it is selected for the actor, *Healthcare Provider*.

To ensure the validity of the results, the procedure as described in the SCM case study is followed. The satisfaction percentage of the top softgoals of the actors, *Patient* and *Healthcare Provider*, are calculated and are provided in Tables 6.5 and 6.6 respectively.

Table 6.5: Satisfaction Percentage of Top Softgoals for the actor Patient (Telemedicine goal model) (* indicates options selected)

Weights of LSG		Satisfaction Percentage of TSG			
Time Saving	Quality of Care	Expense		Happiness	
		Patient Centered Care	Provider Centered Care	Patient Centered Care	Provider Centered Care
100	20	100%*	55%	100%*	64%
50	70	100%*	92%	100%*	96%
80	40	100%*	70%	100%*	77%
65	60	100%*	86%	100%*	92%
40	75	97%*	94%	100%*	97%

Form Tables 6.5 and 6.6, it can be seen that the alternatives selected is same as that of the alternatives selected using game-theory-based approach.

To evaluate the game-theory-based approach with optimal goal analysis (proposed in Chapter 5), the alternatives selected from the game-theory-based goal analysis are compared with those that are selected by using the optimal goal analysis method. The evaluation was performed using the case studies: Meeting Scheduler System (Van Lamsweerde, 2009), Kids Youth Counselling (Horkoff and Yu, 2016) and Telemedicine (Yu, 2001) from the existing literature. Only the

Table 6.6: Satisfaction Percentage of Top Softgoals for the actor Healthcare Provider(Telemedicine goal model)(* indicates options selected)

Weights of LSG		Satisfaction Percentage of TSG			
Efficient Operations	Effective Treatments	Viable Healthcare Service		Maintenance Cost	
		Patient Centered Care	Provider Centered Care	Patient Centered Care	Provider Centered Care
20	84	61%	88%*	75%	100%*
70	60	62%	100%*	72%	100%*
50	70	62%	100%*	74%	100%*
80	40	53%	100%*	60%	100%*
60	30	40%	76%*	45%	83%*

results of the Telemedicine and SCM case studies are presented in this chapter. Since the optimal goal analysis in Chapter 5 performs only maximisation of top softgoals, the top softgoals of the SCM goal diagram are *Performance* for actor *Manufacturer* and *Increase in Customers* for actor *Retailer*. Similarly, for the Telemedicine case study goal diagram, the top softgoals are *Happiness* for actor *Patient* and *Viable Healthcare service* for actor *Healthcare Provider*. For these goal models, the objective functions are obtained based on the optimal goal analysis method (for details on objective functions generation, the reader is directed to Chapter 5). The objective functions for the SCM goal model are represented as follows:

$$F_{Traditional}(\omega) = \max\{0.0768 * \omega_{11} + 0.3072 * \omega_{21}\}$$
$$F_{VMI}(\omega) = \max\{0.3072 * \omega_{11} + 0.512 * \omega_{21}\}$$
$$F_{Traditional}(\omega) = \max\{0.4096 * \omega_{12} + 0.4096 * \omega_{22}\}$$
$$F_{Online}(\omega) = \max\{0.4096 * \omega_{12} + 0.3072 * \omega_{22}\}$$

Subject to

$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 2 \text{ and } k = 1 \text{ to } 2$$

The objective functions for the Telemedicine goal model are represented as follows:

$$F_{PatientCenteredCare}(\omega) = \max\{0.4096 * \omega_{11} + 0.4096 * \omega_{21} + 0.1024 * \omega_{12} + 0.1024 * \omega_{22}\}$$
$$F_{ProviderCenteredCare}(\omega) = \max\{0.1024 * \omega_{11} + 0.512 * \omega_{21} + 0.4096 * \omega_{12} + 0.4096 * \omega_{22}\}$$

Subject to

$$0 \leq \omega_{dk} \leq 100 \text{ for } d = 1 \text{ to } 2 \text{ and } k = 1 \text{ to } 2$$

Solving the above objective functions by optimal goal analysis tool (Chapter 5), the weights of the leaf softgoals (Table 6.7) and the scores of the top softgoals (Tables 6.8 and 6.9) are obtained.

Table 6.7: Weights of SCM and Telemedicine goal models

SCM			Telemedicine		
Actor	Leaf Softgoals	Weight	Actor	Leaf Softgoals	Weight
Manufacturer	OntimeDeliveryProcess	53	Patient	Time Saving	100
	Production Process	10		Quality of Care	18
Retailer	Immediacy Service	73	Healthcare Provider	Efficient operations	10
	Quality Goods	10		Effective Treatments	84

Table 6.8: Top Softgoals scores of SCM goal model (* indicates options selected)

Actor	Top Softgoals	Alternative Scores	
		Traditional	VMI/Online
Manufacturer	Performance	21	45*
Retailer	Increase in Customers	70*	68

Table 6.9: Top Softgoals scores of Telemedicine goal model(* indicates options selected)

Actor	Top Softgoals	Alternative Scores	
		Patient Centered care	Provider Centered Care
Patient	Happiness	100*	62
Healthcare Provider	Viable Healthcare Service	71	97*

It can be seen from Table 6.8 that for the SCM goal model, the alternative *VMI* is selected for the actor *Manufacturer* and the alternative *Traditional* is selected for the actor *Retailer*. Table 6.9 shows that for the Telemedicine goal model, the alternative *Patient Centered care* is selected for the actor, *Patient*, and the alternative *Provider Centered Care* is selected for the actor, *Healthcare Provider*. These selected alternatives correspond to the alternatives selected using the proposed game-theory-based approach. Hence, it can be concluded that the proposed approach is an improvement on the optimal goal programming approach, in terms of handling the goal analysis of the softgoals having opposing objective functions.

6.4.4 Actor with $n(> 2)$ Top Softgoals

In general, an actor will have any number of top softgoals ($n > 2$) based upon the system requirements. For an actor with n top softgoals, there will be n factorial ($n!$) relationships between the top softgoals. If $n!$ is very large, the analysis process becomes complex. To make the process simpler, the concept of a representative agent is used. It is also assumed that every agent uses the same strategy and obtains the same payoff. The top softgoals have to be either maximised or minimised. Therefore, there will be two representative agents, one representing the maximising top softgoals and another representing the minimising top softgoals. Hence, for the i^* framework with $n > 2$ top softgoals there will be two representative agents. Now, the problem becomes a two-player game which can be solved using the above Zero-sum game-theory-based goal analysis.

6.5 Chapter Summary

In this chapter, a game-theory-based approach for goal evaluation in the i^* framework has been proposed. This approach was implemented using the IBM CPLEX tool integrated with the Java eclipse environment and tested using case studies from the existing literature: Supply chain management, Youth Counsellor and the Telemedicine System. Evaluation of the proposed approach was based on the alternative selection of each actor in the selected goal model. The proposed game-theory-based goal analysis gives the same result as the optimal goal analysis approach. The optimal goal analysis method performs goal analysis of the i^* model by only maximising top softgoals. The advantage of the game theory based goal analysis is that it can perform analysis of the i^* model with top softgoals having maximising and minimising objective functions simultaneously occurring in real-life situations. On the other hand, it is not possible to perform this task using the optimal goal analysis method. This chapter also concludes the details of our goal analysis framework. In the next chapter, we present an evaluation of our tool and compare it with the existing Open OME tool.

Chapter 7

Tool Validation

Having implemented our approach for quantitative and optimal goal analysis in the i^* framework, we now perform an empirical evaluation of the approach and tool in comparison with an existing tool for the textiti* framework namely OpenOME. The evaluation is performed in terms of the following criteria: time, result interpretation, automation, functionality, performance, satisfaction and ease of use. We first describe the qualitative approach using OpenOME tool and its limitations. We give a brief introduction to our approach to overcoming the limitations of OpenOME in Section 7. In Section 7, we describe the evaluation method and then analyse the results of the evaluation. And finally in Section 7, we present the discussion of the results and threats to validity.

7.1 State of the Art

For i^* modeling, many tools (wik, date) have been developed. These include OpenOME, OME, REDEPEND-REACT-BCN, T-Tool, J-PRiM and DesCARTES to name a few. Although these tools support modeling of the i^* framework, only a few support the analysis or evaluation of the i^* model. The tools that support analysis or evaluation of the i^* framework are OpenOME (/OME), REDEPEND-REACT, J-PRiM and DesCARTES.

7.1.1 REDEPEND-REACT

REDEPEND (Pavan et al., 2003; Grau et al., 2005), is a graphical modelling tool that enables the modeling of complex agent-based systems and their dependencies using the i^* formalism. This tool helps to generate and evaluate systems' different architectures. The software architectures are modelled as an i^* SD model in which the actors represent software domains and stakeholders namely human, hardware, software and organisation. The interaction is represented by the dependencies among the actors of SD. The generated architectures are evaluated based on certain properties of the system. The analysis of SD is done to select an architecture based on several interesting properties (security, accuracy or efficiency) of the system being modelled.

7.1.2 J-Prim

J-PRiM (Grau et al., 2006) is another tool for modeling of the i^* framework based on the PRiM methodology. In PRiM, the modeling is addressed in the context of process re-engineering. In process re-engineering, the new system's specification starts from the examination of the current system and ends with the attainment of the specifications of the system-to-be. In J-PRiM, the elements are added individually when modeling the i^* SD and SR diagram. It differs from other tools in the way the elements are introduced and visualised. J-PRiM does not support visualising models in graphical representation of the i^* elements but shows these in a tree-form hierarchy. It is used for the evaluation of the system model.

7.1.3 DesCARTES

The DesCARTES (Design CASE Tool for Agent-Oriented Repositories, Techniques, Environments and Systems) (des, date) tool is designed to support various model editions: i^* models (Strategic Dependency and Strategic Rationale models), NFR (Non-Functional Requirements) models, UML (Unified Modeling Language) models, AUML (Agent UML) models in the context of Tropos and I-Tropos developments. It helps in the development of the methodology, analysis and design of models as well as forward engineering capabilities and an integrated software project management module. It mainly supports Tropos developments and UML/RUP developments. In addition, it supports the following developments:

- i^* Strategic Dependency and Strategic Rationale Models;
- NFR Goal Analysis Models;
- UML Use-Cases and Business Use-Cases Models;
- Enterprise Models;

The goal analysis of the i^* framework is not supported by this tool.

7.1.4 OpenOME

The Organisation Modelling Environment (OME) (ome, date) is a general, goal-oriented and agent-oriented modeling and analysis tool. The Eclipse based OpenOME tool, is an improved version of OME and the word *'open'* indicates its open source nature. It provides a graphical interface to develop i^* models and also computer-aided analysis. The computer-aided analysis is both backwards (top-down) and forwards (down-top), qualitative, interactive analysis using SAT solving techniques. This tool helps in goal analysis.

7.1.5 Optimal Goal Analysis of the i^* framework

To support quantitative goal analysis and to automate the goal analysis process of the i^* framework, we developed an Optimal tool based on the methods proposed in Chapters 4 and 5. The analysis is forward (bottom-top), quantitative, and uses optimisation to avoid any subjective preferences in goal analysis. The tool was created using Java Eclipse integrated with JUNG (sf, date) and the IBM ILOG

CPLEX optimisation tool (ilo, date). Despite of using Java Swing library for the interface, displaying the framework and other forms of modifications like addition and deletion of intentional elements are controlled through the JUNG open-source software. The JUNG (Java Universal Network/Graph Framework) software is used to support the graphical representation of SR diagram and visualization of the model. The Java-based JUNG is an open source software library that supports a common and extendible language for modeling, reasoning, and visualization of information that can be depicted as a graph or network. The visualization framework provided by JUNG makes it easy to construct tools that involve the interactive analysis of data. The IBM ILOG CPLEX tool is used to support the multi-objective optimisation of this approach. The IBM ILOG CPLEX optimizer is used to solve business mathematical models using powerful algorithms to obtain precise and logical decisions. Additionally, the modeling layer, Concert, in the IBM ILOG CPLEX optimizer facilitates the interface with Java, C++ and C# languages.

An empirical evaluation of the Optimal tool was conducted to test its feasibility. The evaluation involved student volunteers from bachelor and master of software engineering courses. We performed a comparison of Optimal goal analysis tool using the OpenOME tool. The reason for selecting only OpenOME was that the analysis in REDEPEND-REACT was based solely on the SD diagram and also, the analysis requires that specific system properties be known. The reason for not considering J-Prim for empirical evaluation was that it does not support node visualization in graphical representation. Also, the DesCARTES was not used in the empirical evaluation as it did not support goal analysis of the i* framework. In the following section, a brief explanation of the qualitative analysis of the i* framework using the OpenOME tool and the quantitative analysis using the Optimal tool is given before the empirical evaluation is presented.

7.2 OpenOME: Qualitative Analysis of the *i** Framework

For alternative design/goal selection, Jennifer et al. (Horkoff and Yu, 2009) proposed an interactive, qualitative analysis procedure that was implemented using the OpenOME tool. In this method, a set of qualitative labels were used to represent the degree of satisfaction or denial of the intentional elements. A brief description of the procedure is as follows: Once the model has been constructed,

the analysis starts with a general question *"How effective is an alternative with respect to model goals?"*. The intentional elements related to this analysis question are assigned qualitative labels to convey their degree of satisfaction. Using predefined rules, these labels are propagated through the model links to find the degree of satisfaction of the other intentional elements in the model. Human interaction is involved in place of multiple conflicting or partial values to resolve the satisfaction or denial of an intentional element. For each actor, the final satisfaction and denial values for the intentional elements are determined in view of the analysis question. Further analysis and model refinement is done based on judgement of whether or not the design choice is satisfied. We illustrate the steps involved in the qualitative analysis with the Telemedicine case study (Yu, 2002). Telemedicine is the use of information technology and telecommunication to provide remote diagnosis and treatment for patients. Let us consider a simplified Telemedicine case study as shown in Figure 7.1.

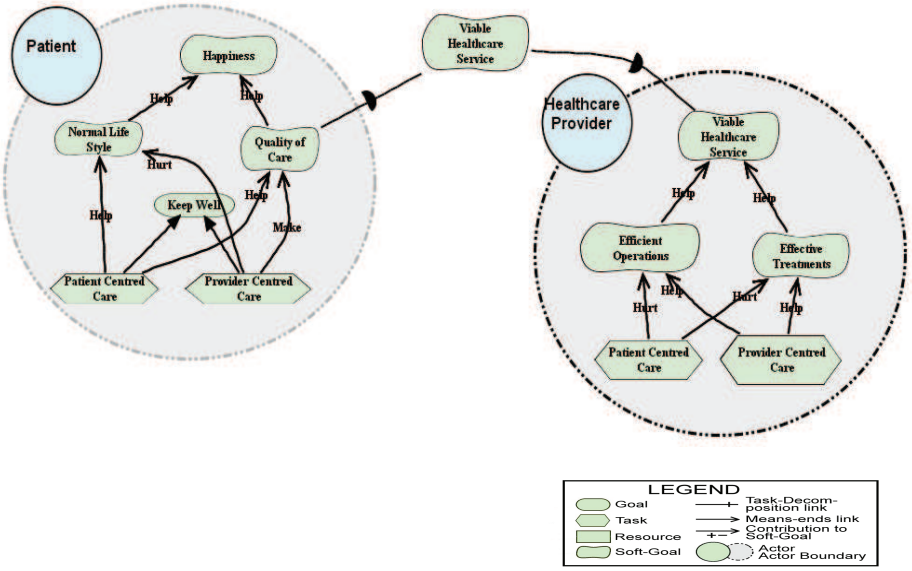

Figure 7.1: SR Model: Telemedicine Example (adapted from (Yu, 2002)).

We illustrate the evaluation of the option in actor *Patient* of the Telemedicine case study. The analysis starts with the question "What is the effect of using the alternative *Provider Centered Care*?". The label assigned for the alternative *Patient Centered Care* is *denied* and for the alternative *Provider Centered Care* is *satisfied*. With this initial assignment and using predefined rules, the softgoal *Normal Life Style* receives *partially denied* from *Patient Centered Care*

and *partially denied* from *Provider Centered Care*. Since the softgoal *Normal Life Style* receives two labels for the same polarity, it received the label *partially denied*. The softgoal *Quality of Care* receives *partially denied* from *Patient Centered Care* and *satisfied* from *Provider Centered Care*. Since the softgoal *Quality of Care* has two labels of different polarity, human judgement is needed to resolve the label assignment. Using the predefined priority rules, the softgoal *Quality of Care* receives the label *satisfied*. The labels are propagated forward to obtain the label of the top softgoal, *Happiness*. The softgoal, *Happiness*, receives *partially denied* from *Normal Life Style* and *partially satisfied* from *Quality of Care*. Again, human judgement is used to decide the label assignment and, using the predefined priority rules, the softgoal *Happiness* receives *partially satisfied*. This overall assessment leads to *partial satisfaction* of the top softgoal, *Happiness*. This evaluation is shown in Figure 7.2.

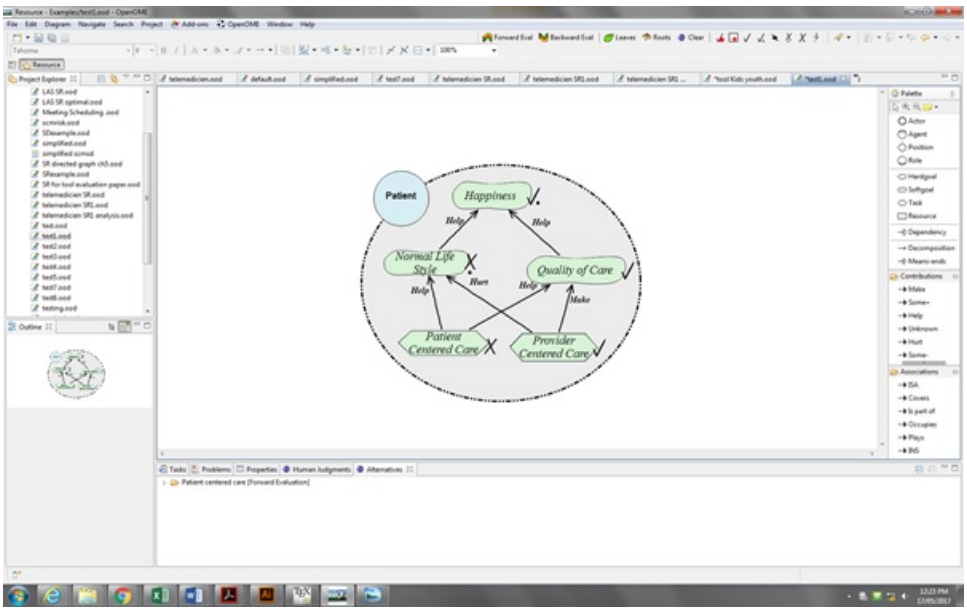

Figure 7.2: OpenOME: Goal Analysis with first alternative

The drawback of this qualitative analysis is the human interaction involved in label propagation. Since label assignment by human interaction is subjective to the analyst, the top softgoals may receive different labels for the same initial assignment. Another issue with this approach is the conflict that arises in the analysis. Also, ambiguity arises in decision making when a top softgoal receives the same label for all the alternatives. This is illustrated by considering the analysis question "what is the effect of the alternative *Patient Centered Care*?" for the actor, *Patient*. The analysis results are shown in Figure 7.3. From Figure

7.3, it can be seen that the effect of the alternative *Patient Centered Care* is also a *partial satisfaction* of the top softgoal, *Happiness*. There arises an ambiguity in the decision making regarding alternative tasks. Another issue is: what happens if the top softgoal receives the label *Unknown*?

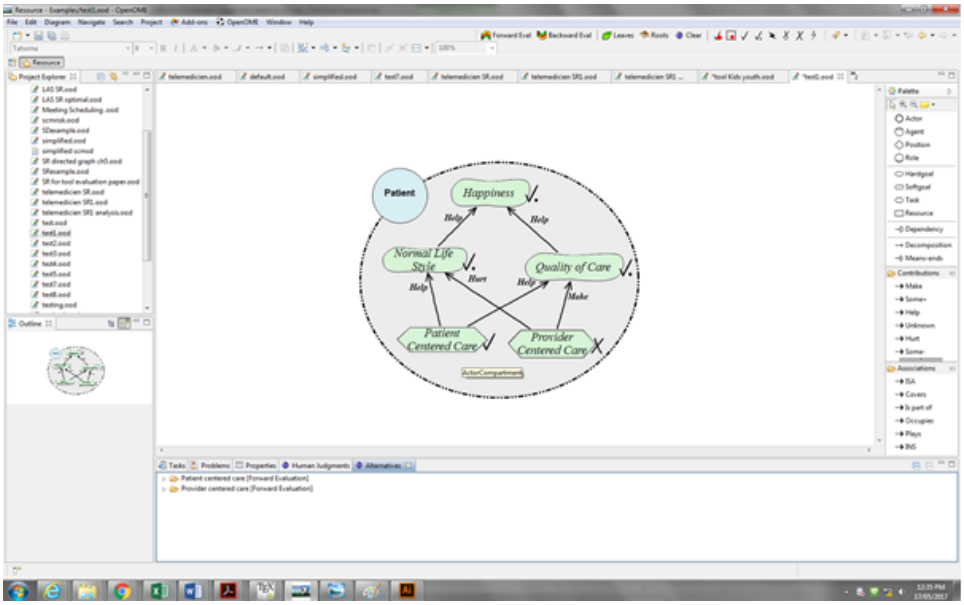

Figure 7.3: OpenOME: Goal Analysis with second alternative

To overcome these problems in the qualitative analysis of the i^* framework, a fuzzy-based quantitative and optimal goal analysis of the i^* framework was proposed in Chapters 3, 4 and 5.

7.3 Optimal Goal Analysis Tool: Quantitative Analysis of the i^* Framework

In the Telemedicine case study, for each actor there are two different alternatives. The task of the analyst is to select an alternative that delivers maximum satisfaction of the top softgoals for each actor. Initially, a fuzzy-based quantitative reasoning of goals for the i^* framework was proposed in Chapter 3.

In this approach, the leaf softgoals are assigned weights in percentage based on their relative preference. The impacts of the goals (or task) on the softgoals represented by *make, help, some+, some-, hurt,* and *break* are given by triangular fuzzy numbers (0.64, 0.80, 1), (0.48, 0.64, 0.80),(0.32, 0.48, 0.64), (0.16, 0.32, 0.48), (0, 0.16, 0.32) and (0, 0, 0.16) correspondingly. Since the impact has a

direct effect on the degree of satisfaction of the softgoals, they are represented by fuzzy numbers to avoid the imprecision associated with decision making. Using the weight of the leaf softgoals and the impact value of the selected alternative on the leaf softgoals, the scores of the leaf softgoals are calculated. These scores are then propagated forward to find the scores of the other softgoals in the hierarchy. For each actor, the top softgoal scores for the alternatives are compared and the alternative that contributes to the maximum satisfaction level is selected. For further details, readers are directed to Chapter 3.

To avoid the subjective preference of the weights assigned to the leaf softgoals and to automate the alternative selection process, an optimal goal analysis procedure using optimisation was proposed in Chapters 4 and 5. Optimisation is used to find the weights of the leaf softgoals. For each actor, an objective function for each alternative in terms of the leaf softgoals is obtained. For details of objective function generation, readers are directed to Chapter 4. These objective functions are solved using the goal programming method to find the weights of the leaf softgoals. These optimal weights of the leaf softgoals are in turn used in the analysis procedure proposed in Chapter 3 to find an alternative that contributes to maximum satisfaction. For the approach, a tool using Java Eclipse integrated with the IBM CPLEX optimisation tool was developed. We illustrate the alternative selection in optimal goal analysis procedure with the Telemedicine case study. The input to the tool is the SR diagram. The objective functions are generated based on the input and are solved by calling the IBM CPLEX optimisation tool. The outputs of the optimisation tool are the weights of the leaf softgoals. Based on these weights, the quantitative analysis is performed. The tool output for the actor, *Patient*, is given in Figures 7.4 and 7.5.

7.4 Empirical Evaluation (Controlled Experiment) to Evaluate the Usability of the Optimal *i** Tool

In this section, details are presented of the controlled experiment carried out to assess the time consumption, functionality, performance, ease of use, result interpretation, automation, user satisfaction of the Optimal goal analysis tool in comparison with the existing OpenOME tool. Table 7.1 shows the key elements of the controlled experiment. To perform this experiment, the guidelines proposed

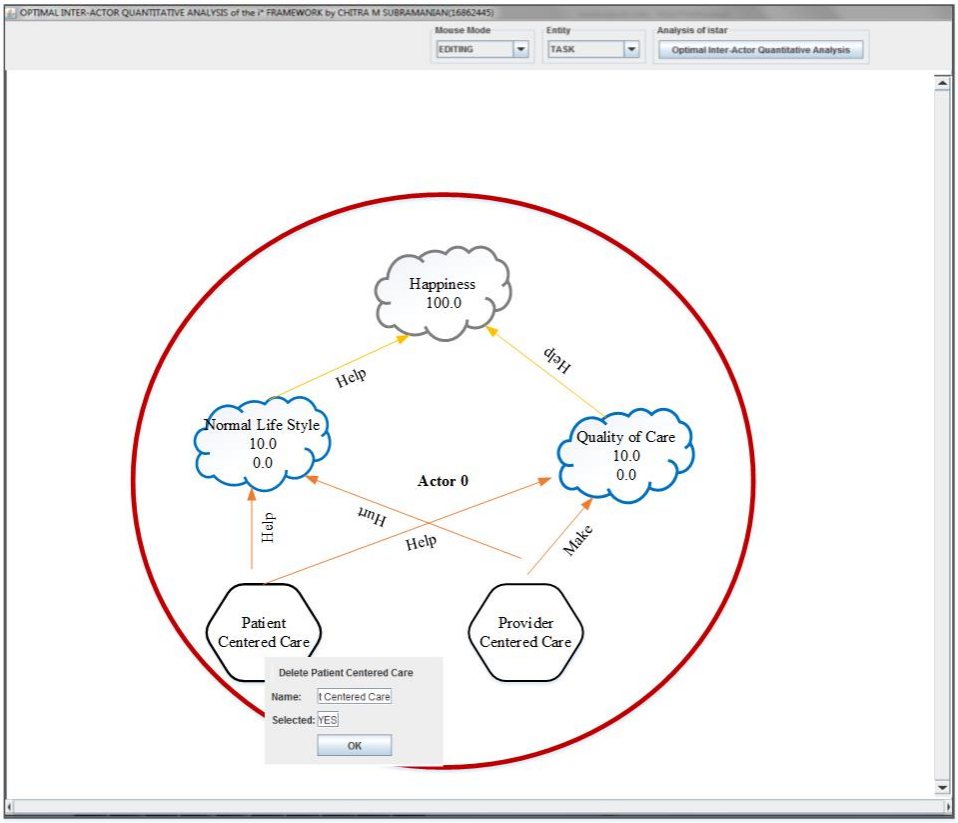

Figure 7.4: Optimal Tool: Goal Analysis (First Alternative)

by Wohlin et al. (Wohlin et al., 2012), on how to define, plan, run, and analyse the results of an experiment in software engineering domain were used.

7.4.1 Research Objectives

The main objective of the controlled experiment was to analyse two tools used for requirements analysis: OpenOME and Optimal goal analysis tools. To achieve the desired objective, seven properties were evaluated during the experiment: time taken for analysis, interpretation of results, user interaction, functionality (suitable for goal analysis), performance (slow, responsive or fast), user's satisfaction with analysis results, and ease of use. The results of the investigation would help researchers and decision makers to select an appropriate tool for requirements analysis of the i^* framework. The experiment was conducted using 20 students from undergraduate and master degrees who have a knowledge of software engineering and/or requirements engineering. The students were asked to test both tools using i^* models taken from existing RE literature. Thus, this

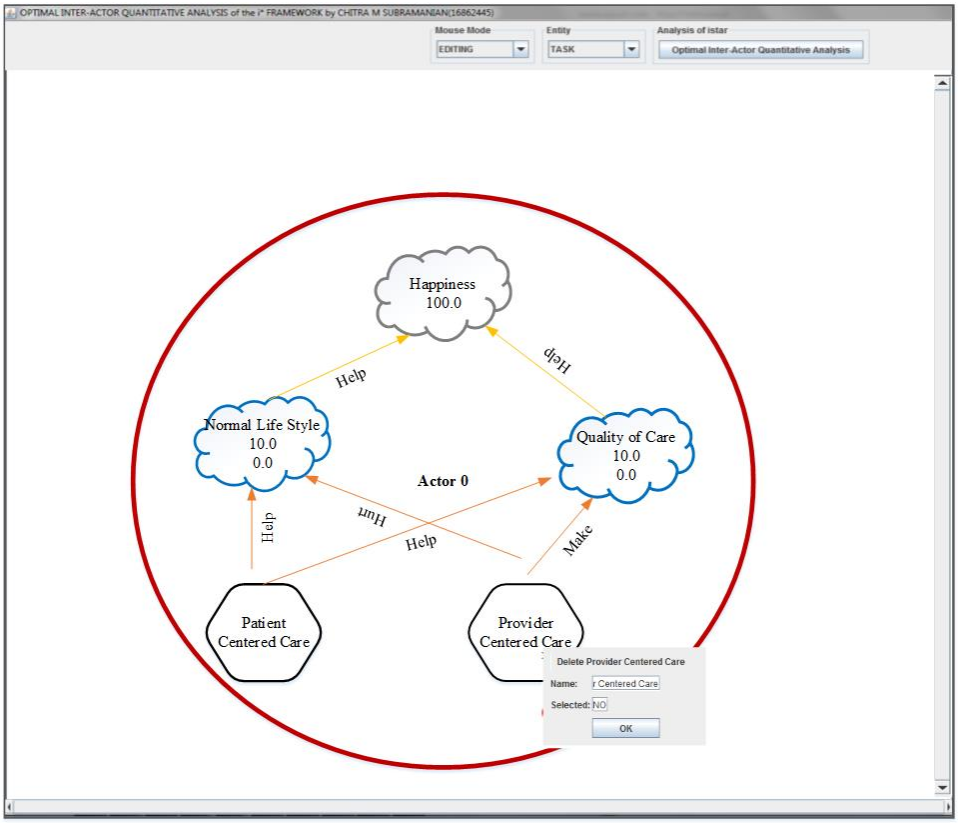

Figure 7.5: Optimal Tool: Goal Analysis (Second Alternative)

experiment is categorised as a blocked subject-object study.

In brief, the objective of the research is defined as:

Analyse the OpenOME tool (qualitative goal analysis) and Optimal goal analysis tool (quantitative optimal goal analysis) with regards to time taken for analysis, interpretation of results, user interaction, functionality (suitable for goal analysis), performance (slow, responsive or fast), user's satisfaction with analysis results, and ease of use from the perspective of decision makers in the context of software engineering students. To achieve the objective of this experiment, answers to the following questions were sought.

RQ1: What is the time (T) taken by the OpenOME and Optimal goal analysis tools, when applied to perform alternative option selection process for a given i^* model?

RQ2: Which tool - the OpenOME or the Optimal goal analysis tool, makes the interpretation (I) of results easier?

RQ3: Which tool - the OpenOME or the Optimal goal analysis tool - is automated (A)?

Table 7.1: Key Elements of Controlled Experiment

Aim	Analyze the tools OpenOME and Optimal goal analysis for alternative option selection with the intent of measuring time, functionality, automation and performance, satisfaction,ease of use
Context	The experiment was carried out using 20 real subjects (who has a knowledge of Requirements Engineering/Software Engineering) with i^* case studies taken from the existing RE literature
Dependent variables	time, result interpretation, automation, functionality, performance, satisfaction and ease of use
Independent variables	OpenOME tool (Qualitative approach) and Optimal goal analysis tool (Quantitative approach)

RQ4: Which tool - the OpenOME or the Optimal goal analysis tool - is more suitable (F) for goal analysis?

RQ5: How fast (P) are OpenOME and Optimal goal analysis tools when applied to perform alternative option selection process?

RQ6: Which tool - the OpenOME or the Optimal goal analysis tool - gives user satisfaction (S) results?

RQ7: Which tool - the OpenOME or the Optimal goal analysis tool - is easier to use (U)?

7.4.2 Experimental Design

7.4.2.1 Context Selection

The experiment was carried out in a laboratory environment, and hence it was executed off line (i.e., the experiment was not conducted in an industrial setting). The participants in the experiment were undergraduate and postgraduate students from our university. As pointed out by Carver et al. (Carver et al., 2003), this research took advantage of the benefit of considering students for experiments. The experiment is classified as specific as it targeted the goal analysis/requirement analysis process in OpenOME tool and Optimal goal analysis tool. The experiment addressed a real problem as it sought to determine the differences between two tools when tested on the same set of i^* models.

7.4.2.2 Formulation of Hypotheses

The following hypotheses were established based on the research questions (RQ1 to RQ7) articulated in Section 6.1 for the controlled experiment:

- Time Null Hypothesis (H_T0): There is no considerable difference between the OpenOME and Optimal goal analysis tools in the view of the time taken to perform the goal analysis of a given i^* model.

  $\delta(T(OpenOME\ tool)) = \delta(T(Optimal\ goal\ analysis\ tool))$

- Time Alternative Hypothesis (H_T1): There is a considerable difference between the OpenOME and Optimal goal analysis tool in the view of the time taken to perform the goal analysis of a given i^* model.

  $\delta(T(OpenOME\ Tool)) \neq \delta\ (T(Optimal\ goal\ analysis\ tool))$

- Interpretation Null Hypothesis (H_I0): There is no considerable difference between the OpenOME and Optimal goal analysis tools in terms of interpretation of results from the goal analysis of a given i^* model.

  $\delta(I(OpenOME\ Tool)) = \delta(I(Optimal\ goal\ analysis\ tool))$

- Interpretation Alternative Hypothesis (H_I1): There is a considerable difference between the OpenOME and Optimal goal analysis tools in terms of interpretation of results from the goal analysis of a given i^* model.

  $\delta(I(OpenOME\ Tool)) \neq \delta(I(Optimal\ goal\ analysis\ tool))$

- Automation Null Hypothesis (H_A0): There is no considerable difference between the OpenOME and Optimal goal analysis tools in terms of user interaction involved in the goal analysis of a given i^* model.

  $\delta(A(OpenOME\ Tool)) = \delta(A(Optimal\ goal\ analysis\ tool))$

- Automation Alternative Hypothesis (H_A1): There is a considerable difference between the OpenOME and Optimal goal analysis tools in terms of user interaction involved in the goal analysis of a given i^* model.

  $\delta(A(OpenOME\ Tool)) \neq \delta(A(Optimal\ goal\ analysis\ tool))$

- Suitability Null Hypothesis (H_F0) : The functionality of the OpenOME tool is the same as that of the Optimal goal analysis tool.

  $\delta(F(OpenOME\ Tool)) = \delta(F(Optimal\ goal\ analysis\ tool))$

- Suitability Alternative Hypothesis (H_F1): The functionality of the OpenOME tool is not the same as that of the Optimal goal analysis tool.

  $\delta(F(OpenOME\ Tool)) \neq \delta(F(Optimal\ goal\ analysis\ tool))$

- Performance Null Hypothesis (H_P0): The performance of the OpenOME tool is the same as that of the Optimal goal analysis tool.

  $\delta(P(OpenOME\ Tool)) = \delta(P(Optimal\ goal\ analysis\ tool))$

- Performance Alternative Hypothesis (H_P1): The performance of the OpenOME tool is not the same as that of the Optimal goal analysis tool.

  $\delta(P(OpenOME\ Tool)) \neq \delta(P(Optimal\ goal\ analysis\ tool))$

- Satisfaction Null Hypothesis (H_S0): There is a no considerable difference between the OpenOME and Optimal goal analysis tools in terms of user satisfaction with the results from the goal analysis of a given i^* model.

  $\delta(S(OpenOME\ Tool)) = \delta(S(Optimal\ goal\ analysis\ tool))$

- Satisfaction Alternative Hypothesis (H_S1): There is a considerable difference between the OpenOME and Optimal goal analysis tools in terms of user satisfaction with the results from the goal analysis of a given i^* model.

  $\delta(S(OpenOME\ Tool)) \neq \delta(S(Optimal\ goal\ analysis\ tool))$

- Ease of use Null Hypothesis (H_U0): There is no considerable difference between the OpenOME and Optimal goal analysis tools in terms of usability of the tool.

  $\delta(U(OpenOME\ Tool)) = \delta(U(Optimal\ goal\ analysis\ tool))$

- Ease of use Alternative Hypothesis (H_U1): There is a considerable difference between the OpenOME and Optimal goal analysis tools in terms of usability of the tool.

  $\delta(U(OpenOME\ Tool)) \neq \delta(U(Optimal\ goal\ analysis\ tool))$

7.4.2.3 Selection and Measurement of Variables

As with any controlled experiment in the domain of software engineering, the independent and dependent variables of the controlled experiment were identified. The independent variables are the tools OpenOME and Optimal goal analysis. Seven dependent variables were identified based on the research questions stated

in Section 7.4.1: time, result interpretation, automation, functionality, performance, satisfaction and ease of use. In the controlled experiment, the variables were measured using tabular structure for comparison of these two tools. After working with each tool, the test subjects were asked to fill in a questionnaire.

7.4.2.4 Selection of Subjects and Objects

The subjects of this controlled experiment were undergraduate and master students who have a knowledge of software engineering and/or requirements engineering. The experiment was not made compulsory in order to avoid biasing the results. The findings would be more rigorous if subjects were not compelled to participate. In response to our invitation, 20 students agreed to participate in the experiment. All the subjects were given a set of documents: a copy of the consent form, participation information sheet, details about the experiment, test model diagrams and a questionnaire.

The same objects were used to test both tools. The experiment was carried out using two i^* models adapted from existing RE literature: Telemedicine (Yu, 2002) and Youth Counseling (Horkoff and Yu, 2009).

7.4.3 Execution

Prior to the controlled experiment, a brief explanation was given to all subjects about the purpose of goal analysis and goal frameworks. Moreover, a short instruction on how to work with tools, OpenOME and Optimal goal analysis tool was provided. The experiment was conducted in a laboratory room where each computer was installed with the OpenOME tool and Optimal goal analysis tool. The subjects executed the two given goal models, using the OpenOME tool and Optimal goal analysis tool. The session took an hour including the introductory presentation and providing instructions for the use of the tools. After executing the two models using both tools, each participant completed the questionnaire to measure the ease of use of both tools.

7.4.4 Results and Analyses

The results obtained from the experiment were statistically analysed by following T-test, and Chi-2 (denoted χ^2) test (Wohlin et al., 2012) to answer the research questions RQ1 to RQ7. This section presents the analyses of the results obtained from the experiment.

Table 7.2: Average time consumption for OpenOME tool and Optimal tool

	OpenOME Tool	Optimal Tool	Difference (OpenOME, Optimal Tool)
Actual time-consumption(in secs)	5.166s	2.333s	2.833s

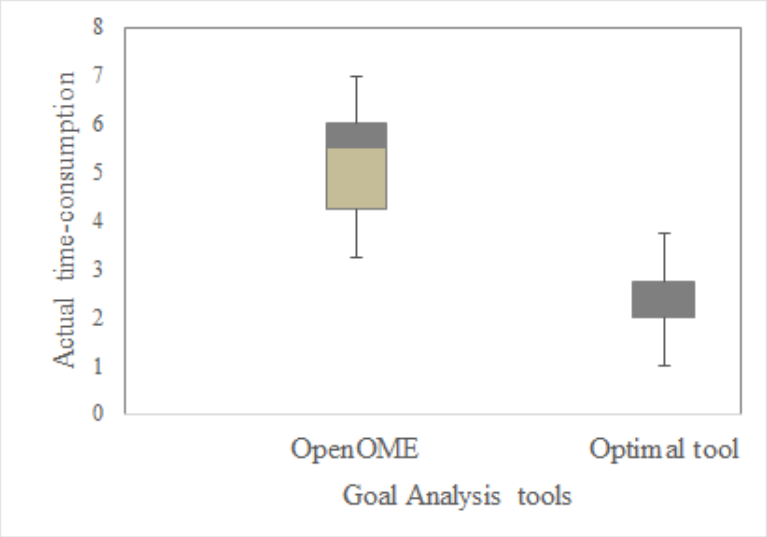

Figure 7.6: Boxplot of the actual time consumption

7.4.4.1 RQ1: What was the time (T) taken by the OpenOME and Optimal goal analysis tools when applied to perform an alternative option selection process for a given *i** model?

The time taken by each tool to perform goal analysis was measured by recording the start time and end time. This time included time to draw the model and time for performing goal analysis. Table 7.2 presents the average time-consumption. The boxplot in Figure 7.6 compares the actual time-consumption for performing goal analysis of the Telemedicine case study using the OpenOME tool and the Optimal tool. From Table 7.2, it can be seen that the difference in average time-

Table 7.3: T-test values for null hypothesis H_T0

$\bar{OpenOME}$	$\bar{Optimal}$	$S^2_{OpenOME}$	$S^2_{Optimal}$	S_p	t_0
5.2	2.35	0.8	0.24	0.7852	12.5

Table 7.4: Chi-Square test values for null hypothesis H_I0

Easy interpretation of results	OpenOME tool	Optimal tool
No. of response	2	18
%	10%	90%
χ^2	12.8	
p-value	0.000347(<0.05)	

consumption between the OpenOME tool and Optimal tool collected from 20 subjects is 2.833s, which corresponds to a 55% reduction. This is also shown in Figure 7.6, where the median value is smaller for the Optimal tool than for the OpenOME tool. Hence, the time taken to perform goal analysis using the Optimal tool is less than that for the OpenOME tool.

To test the null hypothesis (H_T0), we applied a T-test. The test results mean, standard deviation and variance are shown in Table 7.3. From (Wohlin et al., 2012), we found that the null hypothesis could be rejected if $|t_0| > t_{\alpha/2,n+m-2}$. By using the T-test, we obtained $|t_0| = 12.5$. The p-value is less than 0.00001. The result is significant at p <0.05; therefore, the first null hypothesis (H_T0) was rejected. Hence, it was concluded that the time taken using the Optimal tool is less than that for the OpenOME tool.

7.4.4.2 RQ2: which tools between OpenOME and Optimal goal analysis, interpretation (I) of results was easier?

The subjects worked with both the tools using the Telemedicine case study and were asked to answer the following question: Which tool did they found easy (or difficult) to interpret results of goal analysis? Among the 20 subjects, 18 subjects found Optimal tool easier to interpret final results than OpenOME tool. Hence, 90% of the participants found the Optimal tool easier. Since the data were not distributed normally with respect to the variable 'easy interpretation of results' (Table 7.4), we chose Chi-square test (χ^2) (Wohlin et al., 2012) to investigate the second null hypothesis (H_I0) by comparing the number of responses in favour of Optimal tool to the total number of responses. Obviously, there was a statistically critical difference, as p value = 0.000347 (<0.05). Therefore, the second null hypothesis (H_I0) was rejected and it was concluded that the interpretation of results is easier with the Optimal tool when compared with the OpenOME tool.

Table 7.5: Chi-Square test values for null hypothesis H_A0

Involve user interaction	OpenOME tool	Optimal tool
No. of response	18	2
%	90%	10%
χ^2	12.8	
p-value	0.000347(<0.05)	

Table 7.6: Chi-Square test values for null hypothesis H_F0

Functionality	OpenOME tool	Optimal tool	Equally
No. of response	1	4	15
χ^2	20.3		
p-value	0.00003.9(<0.05)		

7.4.4.3 RQ3: Which tool - the OpenOME or the Optimal goal analysis tool - was automated (A)?

To find the automation of the tools, the subjects were asked to answer the following question: Did the tool require involvement of user interaction during goal analysis? Table 7.5 shows the responses obtained from the subjects. It is observed from Table 7.5 that 18 subjects noted that OpenOME tool involved user interaction whereas 2 subjects noted that Optimal tool involve user interaction. As the data were not distributed normally and due to the nature of variable, we applied Chi-square test (χ^2) to investigate the third null hypothesis (H_A0) by comparing the number of responses in favour of the Optimal tool to the total number of responses. It was apparent that there is a statistically significant difference, as p value = 0.000347 (<0.05). Therefore, the third null hypothesis (H_A0) was rejected and it was concluded that the Optimal tool is automated and did not require any user interaction.

7.4.4.4 RQ4: Which tool - the OpenOME or the Optimal goal analysis tool - was more suitable (F) for goal analysis?

To find the functionality of the tools, the subjects were asked to answer the following question: Was the tool suitable for goal analysis? Table 7.6 summarizes the responses collected from test subjects with respect to the functionality of goal analysis by OpenOME tool and Optimal tool. Table 7.6 clearly demonstrates that

Table 7.7: Chi-Square test values for null hypothesis H_P0

Tool	Likert Scale		
	Slow	Responsive	Fast
OpenOME	5	11	4
Optimal tool	0	8	12
χ^2	15.2618		
critical vale at 0.05	5.991		

15 subjects noted that both tools were suitable for goal analysis. Of the remaining 5 subjects, 4 stated that the Optimal tool was more suited for goal analysis than the OpenOME tool. In order to obtain better understanding of which tool is more suitable for goal analysis, the fourth null hypothesis was tested statistically. Since the data were not distributed normally, and due to the nature of variables, we applied Chi-square test (χ^2) to investigate the fourth null hypothesis (H_F0). It was apparent that there is a statistically significant difference, as p value = 0.00003.9 (<0.05). Therefore, the fourth null hypothesis (H_F0) was rejected and it was concluded that the Optimal tool is more suited for goal analysis in comparison with OpenOME tool.

7.4.4.5 RQ5: Which tool - the OpenOME or the Optimal goal analysis tool - had better performance when used to conduct goal analysis?

To measure the performance, after working with both the tools, each participants was asked to indicate his/her judgement using Likert scale. The results of this post-questionnaire are given in Table 7.7. For the Optimal tool, the majority of the participants believed that it is fast in comparison with the OpenOME tool. Hence, it could be said that the Optimal tool has better performance than does the OpenOME tool. To test the fifth null hypothesis (H_P0), again the Chi-Square test with two variables was applied. After applying the test, χ^2 was found to be 15.2618 which is greater than the critical value (5.991). Hence, the fifth null hypothesis (H_P0) was rejected. Thus, it was concluded that the Optimal tool performs better than the OpenOME tool.

Table 7.8: Chi-Square test values for null hypothesis H_S0

Tool	Likert Scale		
	Not Satisfied	Partially Satisfied	Fully Satisfied
OpenOME	10	2	8
Optimal tool	0	2	18
χ^2	18.9154		
critical vale at 0.05	5.991		

Table 7.9: Chi-Square test values for null hypothesis H_U0

Tool	Likert Scale		
	Easy	medium	hard
OpenOME	12	4	4
Optimal tool	17	3	0
χ^2	5.00493		
critical vale at 0.05	5.991		

7.4.4.6 RQ6: Which tool - the OpenOME or the Optimal goal analysis tool - gave better user satisfaction (S) results?

To test the satisfaction of the goal analysis results, the subjects were asked to record their satisfaction using a Likert scale. The post-questionnaire results are given in Table 7.8. The majority of the participants had better satisfaction with the Optimal tool than with OpenOME tool. Hence, it could be said that the Optimal gives better satisfaction results than the OpenOME tool. To test the sixth null hypothesis (H_S0), again the Chi-Square test with two variables was applied. After applying the test, χ^2 was found to be 18.9154 which is greater than the critical value (5.991). Hence, the sixth null hypothesis (H_S0) was rejected. Thus, it was concluded that the Optimal tool gave better results in terms of satisfaction than did the OpenOME tool.

7.4.4.7 RQ7: Which tool between OpenOME and Optimal goal analysis was easier to use (U)?

To test the usability of the goal analysis tools, the subjects were asked to record their judgement using a Likert scale. The post-questionnaire results are given in Table 7.9. As can be seen from Table 7.9, the opinions of the participants seem to differ regarding the usability of the OpenOME tool and the Optimal

tool. In order to obtain a better understanding of which tool is more easy to operate and understand, the seventh null hypothesis was tested statistically. Since the data were not distributed normally, and due to the nature of the variable, the Chi-square test (χ^2) to investigate the seventh null hypothesis (H_U0) was applied. It is apparent that there is a no significant difference, as χ^2 is 5.00493 (<5.991). Therefore, the seventh null hypothesis (H_U0) was accepted and it can be concluded that both the Optimal tool and the OpenOME tool are easy to use and understand.

7.5 Discussion

Based on the statistical analysis from Section 6.4, the Optimal tool was found to be better than the OpenOME tool in terms of the time taken for goal analysis, interpretation of results, automation, functionality, performance, and satisfaction. The reason for this is that Optimal tool is quantitative in nature and did not involve any subjective preferences in the goal analysis. However, in terms of ease of use, subjects found both tools easy to use and understand. By overall comparison, it was concluded that the Optimal goal analysis tool is significantly better than the OpenOME tool in performing goal analysis of the i^* framework.

7.5.1 Threats to Validity

The threats to the validity of the experiment are analysed in this section. The main threat for this experiment was the number of subjects (N=20). This number is comparatively small: however, the knowledge of the subjects who contributed to this experiment was significant. It is worth noting that the results were obtained in a situation where the OpenOME and Optimal tools were applied to small goal models. An essential factor to consider when performing controlled experiments is to carry out experiments in an acceptable amount of time. However, most of the real-life industrial software include larger goal models. Hence, the results of the experiments could not be generalized to all industrial software. These results might be valid only when small goal models are used. Moreover, the Optimal tool was static in performing optimisation. Therefore, the evaluation of Optimal tool in situations of large goal models and dynamic optimisation of goal models could be considered for further empirical investigation of this approach.

7.6 Chapter Summary

In this chapter, we presented a description of the empirical evaluation carried out in order to evaluate and compare the two goal analysis tools, namely OpenOME tool and Optimal tool. The two tools were compared and analysed to find which is more suitable for goal analysis based on the time taken for goal analysis, interpretation of results, automation, functionality, performance, satisfaction, and ease of use. The statistical analysis of the results obtained from the controlled experiment indicates the superiority of the Optimal tool over OpenOME in terms of time taken for goal analysis, interpretation of results, automation, functionality, performance, and satisfaction. The subjects found that both tools were easy to use. The experiment was carried out with 20 subjects from undergraduate and master studies, who have a knowledge of software engineering and/or requirements engineering. The Telemedicine and Youth Counselling case studies from existing literature were used as test goal models. The future direction for this research is to use dynamic optimisation using the Optimal tool and apply this experiment to larger goal models.

This chapter completes the description of the research carried out in this thesis. In the next chapter, we highlight the main problems that we wanted to solve and summarise our key contributions in addressing those problems. We will also discuss some major limitations in our approach and then propose future work to deal with them.

Chapter 8

Contributions, Limitations and Future work

The success of any software system depends on the degree to which it satisfies the purpose for which it was intended. The term Requirements Engineering (RE) was introduced to refer to the process which determines the purpose of a software system and documents it in a form which is amenable to comprehensible scrutiny, communication followed by implementation. RE is considered to be an inevitable activity of software development cycle, since poor requirements are perceived to be a predominant cause of software problems. The initial recognition of the so-called requirements problems led to development of modeling languages for the definition and analysis of requirements. One of the most critical problems in RE is requirements analysis. Although, many approaches have been proposed for requirements analysis, automated requirements analysis is still a significant challenge in Requirements Engineering (RE).

Goal-Oriented Requirements Engineering (GORE) is an emerging paradigm in RE. Similar to traditional and object-oriented paradigms, GORE also evolved to overcome problems that arise in requirements analysis. Among the GORE frameworks, the i^* framework facilitates the analysis of an enterprise with an emphasis on socio-technical domains which includes stakeholders, their goals, dependencies and alternatives. The i^* framework can be used as a tool for the modeling and reasoning of organizational environments and their information systems. The existing literature works show that the i^* framework supports only qualitative analysis of requirements with some limitations. Therefore, it is important to investigate methods that help to improve the analysis of requirements. Our work aims to fill that gap as well as to automate the analysis of requirements

in the i^* framework. In this chapter, we conclude the work that has been carried out and suggest directions for future related research.

8.1 Summary of contributions

Our major objective was to provide an optimal quantitative support for goal analysis in the i^* framework. We followed a fuzzy-based quantitative approach to perform goal analysis to address the ambiguity problems that arise in qualitative analysis. In other words, we used fuzzy numbers to represent the stakeholders' linguistic representation of requirements. The following research questions were raised in Chapter 1:

1. How to represent the linguistic description of requirements (by stakeholders) in goal analysis?

2. How to effectively represent the subjective preference of quantitative values used in the goal analysis?

3. How to effectively implement requirements of an opposing nature?

4. What type of tool can assist in the process of goal analysis?

We developed a fuzzy-based optimal goal analysis framework, addressing each of the research questions as summarized below.

The linguistic representation of stakeholders' preferences regarding requirements can be easily represented by fuzzy numbers (research question 1). We employed fuzzy numbers to represent the contribution of goals/tasks to goals/softgoals (Chapter 3). Moreover, the i^* framework involved inter-actor dependencies. Therefore, we proposed a fuzzy based quantitative goal analysis using inter-actor dependencies in the i^* framework. We defined our own membership functions for the contributions of goal/softgoal to softgoal which were represented by *help, make, some+, some-, hurt,* or *break*. Once the goal model was obtained, the leaf softgoals were assigned percentage values based on their relative importance. For each alternative, these leaf softgoal values along with the contribution of fuzzy values were propagated to the top softgoals. The alternatives that gave maximum contribution to the top softgoal was selected. To prove the feasibility of the approach, a simulation was implemented in VC++ and evaluation was carried to with case studies from existing literature.

The novel aspect of our framework was the management of subjective preferences (weights) to the leaf softgoals (research question 2). Specifically, the subjective weights to the leaf softgoals were handled using an optimisation technique (Chapter 4) namely multi-objective optimisation. For each actor, the objective functions for each alternative based on its impact on the leaf softgoals was generated. Using the scalarization method, the multiple objective functions were combined into an objective function. This single objective function was then solved using the MATLAB genetic algorithm to obtain the weights of the leaf softgoals. These weights were then used in the analysis procedure described in Chapter 3 to perform goal analysis of the i^* framework. An interesting point of optimisation was sensitivity analysis. Sensitivity analysis was conducted to check the system behaviour for change in input parameter. The input parameter in the objective function was the impact of the tasks on the leaf softgoals. The lower and upper bounds of each impact variable were varied to find the range for which there was no change in the optimal solution.

The goal analysis in the i^* framework was performed by propagating the impact and weight values throughout the entire hierarchy of an actor. The optimisation model developed in Chapter 4 was partial as it was derived based on the leaf softgoals. A complete optimisation model was then built based on all the softgoals and leaf softgoals of an actor (Chapter 5). The multi-objective goal programming concept was used to build complete optimisation model. As in Chapter 4, for each actor, for each alternative an objective function was generated. The multiple objective functions were solved using the IBM ILOG CPLEX optimizer to find the weights of the leaf softgoals. These weights were then used in the analysis procedure described in Chapter 3 to perform goal analysis of the i^* framework. A tool (research question 4) in the Java Eclipse environment was developed to perform the inter-actor goal analysis.

Usually, most of the real-world business problems encounter the simultaneous optimisation of many competitive objective functions (research question 3). To deal with such problems, we proposed a method for goal analysis based on game theory in Chapter 6. For each actor, the opposing multi-objective functions were obtained and were solved to obtain the objective function values. Using these objective function values and the alternatives of an actor, a payoff matrix was obtained. Using the zero-sum game theory approach, the payoff matrix was solved to identify the optimal alternatives. The approach was implemented in the Java Eclipse environment integrated with the IBM ILOG CPLEX optimizer.

Finally, we performed an empirical evaluation to assess the efficiency and

effectiveness of our optimal goal analysis framework. The evaluation was performed against the existing OpenOME tool for the i^* goal analysis. The evaluation's results demonstrated that the approach was effective in comparison with OpenOME tool. The evaluation's results also lead to some potential future work that are discussed in the following section.

8.2 Limitations and Future work

Our research can be further improved by addressing some of the potential limitations where further research and extensions to the i^* goal analysis can be performed.

8.2.1 Additional parameters for the i^* evaluation

Our fuzzy based inter-actor quantitative approach has used only softgoal interdependencies in the goal analysis of i^* framework. In an i^* goal model, there are also other types of dependencies including goal dependency, resource dependency and task dependency. These dependencies also contribute to the analysis of goals in a strategic dependency model of an i^* framework. Therefore, potential future work may involve an extension to the goal analysis to include all forms of dependencies.

The goals contributing to the softgoals evaluation can also be associated with other parameters such as cost and time for the development of the goal, and the risk or other factors involved in the development. In the optimal goal analysis procedure, these factors can also be included in the evaluation of the goals in order to determine the goals with maximum softgoal satisfaction and minimum cost factors.

8.2.2 Implementation Issues

In the implementation of optimal quantitative goal analysis, the contributions or impacts of the task to goals namely *make*, *help*, *some+*, *some-*, *hurt* and *break* are represented by fuzzy numbers. The membership functions for these fuzzy numbers can be selected in a subjective way. Different persons may wish to select different membership functions to represent the same idea and also choice depends on the context in which it is used. In the implementation of our approach, we established

fixed values for the contributions. The tool can be modified to choose the fuzzy values for the contributions.

The main focus of our work is on optimisation of the i^* frameworks to automate goal analysis. The implementation of our optimal algorithm is limited in human involvement for generating objective functions. In the tool implementation of optimal goal analysis, we have performed static way of generating objective functions for a given i^* model. However, in practice, this has to be performed dynamically. Therefore, this implementation should be thoroughly modified for general use.

Another concern with the tool is regarding sensitivity analysis of the i^* framework. In Chapter 4, we have performed only a simulation for sensitivity analysis of i^* framework using visual VC++. As a future line of research, the optimal goal analysis tool can be extended to incorporate sensitivity analysis. We intend to perform a sensitivity analysis by changing the proportion of the alternatives in the proposed game-theory-based approach.

The interface used in the optimal goal analysis tool is basic and simple because of limited time. We intend to improve the interface of the tool to make it more user friendly and easier to use.

We would also like to perform a more extensive study on the applicability of our optimisation and game theory approach to other models such as Non-Functional Requirements framework, Goal-Oriented Requirements Language framework, and Tropos framework.

We would like to analyze approaches based on Machine Learning to determine whether these methods can be of use in modifying or extending the approach described in this research work. In the analysis of real-time software systems, requirements are sometimes incompletely specified (or not known). To determine the incompleteness of a given set of requirements, inductive learning algorithms can be used in our proposal. To deal with vagueness and uncertainty in data, rough set based approaches can also be used in our proposal.

Bibliography

(no date). Descartes architect. http://www.isys.ucl.ac.be/descartes. (last accessed: March 2017).

(no date). Ibm ilog cplex optimization studio. http://www.ibm.com/developerworks/downloads/ws/ilogcplex. (last accessed: July 2016).

(no date). Introduction to lp solve 5.5.2.0. http://lpsolve.sourceforge.net. (last accessed: June 2015).

(no date). Jung (java universal network/graph framework). http://jung.sourceforge.net. (last accessed: June 2016).

(no date). Openome, an open-source requirements engineering tool. https://se.cs.toronto.edu/trac/ome. (last accessed: March 2017).

(no date). *i** tools. http://www.istarwiki.org. (last accessed: March 2017).

Affleck, A. and A. Krishna (2012). Supporting quantitative reasoning of non-functional requirements: A process-oriented approach. In *Proceedings of the International Conference on Software and System Process*, pp. 88–92. IEEE Press.

Affleck, A., A. Krishna, and N. R. Achuthan (2013). Optimal selection of operationalizations for non-functional requirements. In *Proceedings of the Ninth Asia-Pacific Conference on Conceptual Modelling-Volume 143*, pp. 69–78. Australian Computer Society, Inc.

Affleck, A., A. Krishna, and N. R. Achuthan (2015). Non-functional requirements framework: A mathematical programming approach. *The Computer Journal 58*(5), 1122–1139.

Amyot, D., S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton, and E. Yu (2010). Evaluating goal models within the goal-oriented requirement language. *International Journal of Intelligent Systems 25*(8), 841–877.

Ashamalla, A., G. Beydoun, and G. Low (2017). Model driven approach for real-time requirement analysis of multi-agent systems. *Computer Languages, Systems & Structures 50*, 127–139.

Bond, A. H. and L. Gasser (2014). *Readings in distributed artificial intelligence*. Morgan Kaufmann.

Bresciani, P., A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems 8*(3), 203–236.

Burgess, C. and A. Krishna (2009). A process-oriented approach for the optimal satisficing of non-functional requirements. *Trustworthy Software Development Processes*, 293–304.

Burgess, C., A. Krishna, and L. Jiang (2009). Towards optimising non-functional requirements. In *Quality Software, 2009. QSIC'09. 9th International Conference on*, pp. 269–277. IEEE.

Caramia, M. and P. Dell'Olmo (2008). *Multi-objective management in freight logistics: Increasing capacity, service level and safety with optimization algorithms*. Springer Science & Business Media.

Carver, J., L. Jaccheri, S. Morasca, and F. Shull (2003). Issues in using students in empirical studies in software engineering education. In *Software Metrics Symposium, 2003. Proceedings. Ninth International*, pp. 239–249. IEEE.

Charnes, A. and W. W. Cooper (1961). *Management models and industrial applications of linear programming*, Volume 1. JSTOR.

Chung, L., B. A. Nixon, E. Yu, and J. Mylopoulos (2012). *Non-functional requirements in software engineering*, Volume 5. Springer Science & Business Media.

Dardenne, A., S. Fickas, and A. Van Lamsweerde (1991). Goal-directed concept acquisition in requirements elicitation. In *Proceedings of the 6th international workshop on Software specification and design*, pp. 14–21. IEEE Computer Society Press.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation 6*(2), 182–197.

Démuth, A. (2013). Game theory and the problem of decision–making. *Edition Cognitive Studies, ISBN*, 978–83.

Deng, H. and A. Molla (2008). Multicriteria analysis for evaluating and selecting e-markets in business-to-business e-business. In *Proceedings of the 2008 International MultiConference of Engineers and Computer Scientists, Hong Kong*.

Dybå, T., R. Prikladnicki, K. Rönkkö, C. Seaman, and J. Sillito (2011). Qualitative research in software engineering. *Empirical Software Engineering 16*(4), 425–429.

Eric, S. (2011). *Social modeling for requirements engineering*. MIT Press.

Franch, X. (2006). On the quantitative analysis of agent-oriented models. In *International Conference on Advanced Information Systems Engineering*, pp. 495–509. Springer.

Gani, A. N. and S. M. Assarudeen (2012). A new operation on triangular fuzzy number for solving fuzzy linear programming problem. *Applied Mathematical Sciences 6*(11), 525–532.

Giorgini, P., J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani (2002). Reasoning with goal models. In *International Conference on Conceptual Modeling*, pp. 167–181. Springer.

Goncalves, J. and A. Krishna (2015). Dynamic non-functional requirements based model-driven agent development. In *Software Engineering Conference (ASWEC), 2015 24th Australasian*, pp. 128–137. IEEE.

Goncalves, J. and A. Krishna (2016a). Incorporating change management within dynamic requirements-based model-driven agent development. *The Computer Journal*.

Goncalves, J. Z. and A. Krishna (2016b). Optimal requirements dependent model-driven agent development. In *Transforming Healthcare Through Information Systems*, pp. 135–151. Springer.

Grau, G., X. Franch, and S. Avila (2006). J-prim: A java tool for a process reengineering i* methodology. In *Requirements Engineering, 14th IEEE International Conference*, pp. 359–360. IEEE.

Grau, G., X. Franch, and N. A. Maiden (2005). Redepend-react: an architecture analysis tool. In *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, pp. 455–456. IEEE.

Hadigheh, A. G. and T. Terlaky (2006). Sensitivity analysis in linear optimization: Invariant support set intervals. *European Journal of Operational Research 169*(3), 1158–1175.

Harman, M. (2007). The current state and future of search based software engineering. In *2007 Future of Software Engineering*, pp. 342–357. IEEE Computer Society.

Heaven, W. and E. Letier (2011). Simulating and optimising design decisions in quantitative goal models. In *Requirements Engineering Conference (RE), 2011 19th IEEE International*, pp. 79–88. IEEE.

Horkoff, J. and E. Yu (2009). Evaluating goal achievement in enterprise modeling–an interactive procedure and experiences. In *IFIP Working Conference on The Practice of Enterprise Modeling*, pp. 145–160. Springer.

Horkoff, J. and E. Yu (2016). Interactive goal model analysis for early requirements engineering. *Requirements Engineering 21*(1), 29–61.

Johnson, L. A. and D. C. Montgomery (1974). *Operations research in production planning, scheduling, and inventory control*, Volume 6. Wiley New York.

Kaiya, H., H. Horai, and M. Saeki (2002). Agora: Attributed goal-oriented requirements analysis method. In *Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on*, pp. 13–22. Ieee.

Kelly, A. (2003). *Decision making using game theory: an introduction for managers*. Cambridge University Press.

Lázaro, M. and E. Marcos (2006). An approach to the integration of qualitative and quantitative research methods in software engineering research. *PhiSE 240*.

Lee, K.-H. and S.-W. Lee (2015). Applying game theoretic approach to goal-driven requirements trade-off analysis for self-adaptation. In *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, Volume 3, pp. 330–335. IEEE.

Letier, E. and A. Van Lamsweerde (2004). Reasoning about partial goal satisfaction for requirements and design engineering. In *ACM SIGSOFT Software Engineering Notes*, Volume 29, pp. 53–62. ACM.

Liaskos, S., R. Jalman, and J. Aranda (2012). On eliciting contribution measures in goal models. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pp. 221–230. IEEE.

Liaskos, S., S. A. McIlraith, S. Sohrabi, and J. Mylopoulos (2010). Integrating preferences into goal models for requirements engineering. In *Requirements Engineering Conference (RE), 2010 18th IEEE International*, pp. 135–144. IEEE.

Liaskos, S., S. A. McIlraith, S. Sohrabi, and J. Mylopoulos (2011). Representing and reasoning about preferences in requirements engineering. *Requirements Engineering 16*(3), 227.

Mairiza, D., D. Zowghi, and V. Gervasi (2014). Utilizing topsis: A multi criteria decision analysis technique for non-functional requirements conflicts. In *Requirements Engineering*, pp. 31–44. Springer.

Miller, T., B. Lu, L. Sterling, G. Beydoun, and K. Taveter (2014). Requirements elicitation and specification using the agent paradigm: the case study of an aircraft turnaround simulator. *IEEE Transactions on Software Engineering 40*(10), 1007–1024.

Morandini, M., L. Penserini, A. Perini, and A. Susi (2007). Refining goal models by evaluating system behaviour. In *International Workshop on Agent-Oriented Software Engineering*, pp. 44–57. Springer.

Mylopoulos, J., L. Chung, S. Liao, H. Wang, and E. Yu (2001). Exploring alternatives during requirements analysis. *IEEE Software 18*(1), 92–96.

Mylopoulos, J., L. Chung, and B. Nixon (1992). Representing and using non-functional requirements: A process-oriented approach. *IEEE Transactions on software engineering 18*(6), 483–497.

Mylopoulos, J., L. Chung, and E. Yu (1999). From object-oriented to goal-oriented requirements analysis. *Communications of the ACM 42*(1), 31–37.

Nair, P. R. (2013). E-supply chain management using software agents. *CSI Communications 37*(4), 13–16.

Oliveira, L. S. d. and S. F. Saramago (2010). Multiobjective optimization techniques applied to engineering problems. *Journal of the brazilian society of mechanical sciences and engineering 32*(1), 94–105.

Pavan, P., N. A. Maiden, and X. Zhu (2003). Towards a systems engineering pattern language: Applying i* to model requirements-architecture patterns. In *STRAW*, pp. 134–141.

Pohl, K. (2010). *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated.

Sadiq, M. and S. Jain (2014). Applying fuzzy preference relation for requirements prioritization in goal oriented requirements elicitation process. *International Journal of System Assurance Engineering and Management 5*(4), 711–723.

Sasieni, M. W., A. Yaspan, and L. Friedman (1959). *Operations research–methods and problems*. Wiley.

Thomas, D. J. and P. M. Griffin (1996). Coordinated supply chain management. *European journal of operational research 94*(1), 1–15.

Van Lamsweerde, A. (2000). Requirements engineering in the year 00: a research perspective. In *Proceedings of the 22nd international conference on Software engineering*, pp. 5–19. ACM.

Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, pp. 249–262. IEEE.

Van Lamsweerde, A. (2004). Goal-oriented requirements enginering: a round-trip from research to practice [enginering read engineering]. In *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*, pp. 4–7. IEEE.

Van Lamsweerde, A. (2009). Reasoning about alternative requirements options. In *Conceptual Modeling: Foundations and Applications*, pp. 380–397. Springer.

Van Lamsweerde, A. and E. Letier (2004). From object orientation to goal orientation: A paradigm shift for requirements engineering. In *Radical Innovations of Software and Systems Engineering in the Future*, pp. 325–340. Springer.

Wang, X.-T. and W. Xiong (2011). An integrated linguistic-based group decision-making approach for quality function deployment. *Expert Systems with Applications 38*(12), 14428–14438.

Waters, M., L. Padgham, and S. Sardina (2015). Improving domain-independent intention selection in bdi systems. *Autonomous Agents and Multi-Agent Systems 29*(4), 683–717.

Wohlin, C., P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Yazdania, S., Y.-T. Linb, W. Caic, and E. Huanga (2017). A game theory perspective on requirement-based engineering design.

You, Z. (2004). *Using meta-model-driven views to address scalability in i* models*. Ph. D. thesis, University of Toronto.

Yu, E. (2001). Agent-oriented modelling: software versus the world. In *International Workshop on Agent-Oriented Software Engineering*, pp. 206–225. Springer.

Yu, E. (2002). Agent-oriented modelling: software versus the world. *Agent-Oriented Software Engineering II*, 206–225.

Yu, E. (2011). Modelling strategic relationships for process reengineering. *Social Modeling for Requirements Engineering 11*, 2011.

Yu, E. S. and J. Mylopoulos (1994). Understanding" why" in software process modelling, analysis, and design. In *Software Engineering, 1994. Proceedings. ICSE-16., 16th International Conference on*, pp. 159–168. IEEE.

Yu, E. S. and J. Mylopoulos (1995). From er to armodelling strategic actor relationships for business process reengineering. *International Journal of Cooperative Information Systems 4*(02n03), 125–144.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control 8*(3), 338–353.

Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoningi. *Information sciences 8*(3), 199–249.

Zou, W., Y. Zhu, H. Chen, and B. Zhang (2011). Solving multiobjective optimization problems using artificial bee colony algorithm. *Discrete dynamics in nature and society 2011*.