ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ, ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

Συμβολικές Αναπαραστάσεις και Κοινή Λογική σε Ανοικτά Πολυ-πρακτορικά Συστήματα

Γεώργιος Κ. Γιαννίκης

Μια διατριβή που εκπονήθηκε για την απονομή
Διδακτορικού Διπλώματος.

Βόλος, 2009

UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF COMPUTER AND COMMUNICATIONS ENGINEERING

Symbolic Representations and Common-sense Reasoning in Open Multi-agent Systems

Georgios K. Giannikis

A thesis submitted for the degree of
Doctor of Philosophy

Volos, 2009

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ, ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

Συμβολικές Αναπαραστάσεις και Κοινή Λογική σε Ανοικτά Πολυ-πρακτορικά Συστήματα

Γεώργιος Κ. Γιαννίκης

Μια διατριβή που εκπονήθηκε για την απονομή
Διδακτορικού Διπλώματος.

Βόλος, 2009

ABSTRACT

This thesis concerns with intelligent autonomous software agents that populate open computational environments, in which they interact for various purposes, e.g. competitively in the case of auctions or resource allocation problems, collaboratively in the case of distributed problem solving or parallel processing, joint planning, etc. We use the term *open* to characterize a computational environment in Hewitt's sense, that is to describe an environment that is dynamic, continuous, unobservable (or, at best, partially observable) and non-deterministic.

Agents in such environments possess, unavoidably, information that is incomplete, imprecise, maybe even incorrect, due to the very fact that the environment is open and, at the very least, agents join and leave it as they please. The information exchanged between agents may be delayed, or distorted by noise during its communication, and in any case, as the environment evolves, it is bound to change.

The interactions among agents in any multi-agent system are typically governed by norms. These may refer to restrictions on communication means among agents, or to particular coordination mechanisms, liveness and safety properties of the system etc. In some application areas, such as e-commerce, additional norms may regulate the agents' behaviours, resulting from agreements into which the agents enter willingly, and possibly from the protocol that governs the e-market. Norms prescribe what each agent is obliged, permitted, prohibited, empowered and so on, to do during its life in the particular environment. *Autonomous* agents decide for themselves which norms to subject itself to and whether to comply with the norms of their environment. This decision-making is all the more challenging when an agent has to perform it in circumstances where its available knowledge is incomplete/imprecise/incorrect.

This thesis addresses the need and requirement for common-sense reasoning agents in open computational environments. We discuss and illustrate our proposals with reference to an e-commerce example. First, based on the example scenario, we identify the requirements for the representation of the norms that govern the environment and the specifications for the environment itself. Nevertheless, our proposals are generally applicable to any case where multiple agents interact and their interaction is governed by some *contract* as a coordination or collaboration mechanism.

Primarily, this thesis examines and motivates the need of agents to fill in information gaps by resorting to assumptions. Agents need to be able to identify and use assumptions dynamically, in any open computational environment, as well as in the particular application context of an e-commerce example. We present a novel approach to dynamic assumption identification and hypothetical non-monotonic reasoning inspired by the syntax and semantics of Default Logic, without however resorting to proof, which is notably computationally hard. We discuss in detail what distinguishes our approach from other work on dynamic assumption based reasoning, namely i.e., we do not resort to a pre-specified pool of assumptions, nor to goal-orientation as a means to identify candidate assumptions. In this way, we claim, an agent is autonomous in deciding which assumptions are appropriate, given its

knowledge-hypotheses requirements at any given time. We propose symbolic and schematic representations to characterize formally the possible knowledge-hypotheses status of an agent, and we use their properties to characterize their dynamics. From these formal characterizations we derive and present the algorithms that support our approach, which we have implemented in a prototype.

Moreover, this thesis addresses other issues of common-sense reasoning agency, such as the need for temporal reasoning and reasoning about actions and their effects or the need for normative conflict management. It is expected that in realistic scenaria an agent will find itself in a situation where it needs to establish, for itself or other agents, what is obliged (permitted, prohibited, empowered etc.) to do at a given point in time; to determine whether, itself or other agents, comply with the agreement; whether deviation is detected from the prescribed behaviour, to determine what, if any, remedial mechanisms might return the transaction to a normal course; and finally, to establish which norms apply or what holds at a given time point, because multiple, possibly conflicting, norms apply.

ΠΕΡΙΛΗΨΗ

Η παρούσα διατριβή αφορά στους ευφυείς και αυτόνομους πράκτορες λογισμικού ως μέλη ενός ανοιχτού υπολογιστικού περιβάλλοντος, στο πλαίσιο του οποίου αλληλεπιδρούν με διάφορους σκοπούς, για παράδειγμα, ανταγωνιστικά στην περίπτωση δημοπρασίας ή προβλημάτων κατανομής πόρων, συνεργατικά στην περίπτωση κατανεμημένης επίλυσης προβλημάτων, συντονισμένου σχεδιασμού ενεργειών κ.α. Χρησιμοποιούμε τον όρο *ανοιχτό* για να χαρακτηρίσουμε ένα υπολογιστικό περιβάλλον κατά την έννοια που αποδόθηκε από τον Hewitt, δηλαδή ένα περιβάλλον δυναμικό, συνεχές, μη παρατηρήσιμο (ή, στην βέλτιστη περίπτωση, μερικώς παρατηρήσιμο) και μη αιτιοκρατικό.

Σε αυτού του είδους τα περιβάλλοντα, οι πράκτορες, αναπόφευκτα, κατέχουν πληροφορία η οποία είναι μη πλήρης, μη ακριβής, και ίσως, μη ορθή, εξαιτίας του γεγονότος πως το περιβάλλον είναι ανοιχτό, και στην ελάχιστη περίπτωση, οι πράκτορες εισέρχονται και αποχωρούν από το περιβάλλον κατά βούληση. Η πληροφορία που ανταλλάσσεται μεταξύ των πρακτόρων μπορεί να καθυστερεί ή να στρεβλώνεται από θόρυβο κατά τη διάρκεια της επικοινωνίας, και σε κάθε περίπτωση, καθώς το περιβάλλον εξελίσσεται, η πληροφορία αλλάζει κατ’ ανάγκη.

Οι αλληλεπιδράσεις μεταξύ των πρακτόρων σε ένα πολύ-πρακτορικό σύστημα, συνήθως, ελέγχονται από κανόνες. Αυτοί μπορεί να αναφέρονται σε περιορισμούς των μέσων επικοινωνίας μεταξύ των πρακτόρων, σε ιδιαίτερους μηχανισμούς συντονισμού, στην ασφάλεια του συστήματος κ.α. Σε μερικές περιοχές εφαρμογής, όπως το ηλεκτρονικό εμπόριο, μπορεί ενδεχομένως, επιπρόσθετοι κανόνες να διέπουν τη συμπεριφορά των πρακτόρων, οι οποίοι προκύπτουν από συμφωνίες στην οποίες ο πράκτορας εισέρχεται κατά βούληση, και πιθανώς από πρωτόκολλα τα οποία διέπουν την ηλεκτρονική αγορά. Οι κανόνες υπαγορεύουν στους πράκτορες τι υποχρεούνται, επιτρέπεται, απαγορεύεται, έχουν εξουσιοδότηση κ.α., να πράξουν κατά τη διάρκεια της παρουσίας τους στο συγκεκριμένο περιβάλλον. Οι *αυτόνομοι* πράκτορες αποφασίζουν για τον εαυτό τους σε ποιούς κανόνες υπάγονται και εάν θα υπακούουν στους κανόνες τους περιβάλλοντος τους. Αυτή η απόφαση αποτελεί, ακόμα μεγαλύτερη πρόκληση, όταν οι πράκτορες θα πρέπει να τη λάβουν σε συνθήκες όπου κατέχουν πληροφορία μη πλήρη/μη ακριβή/μη ορθή.

Αυτή η διατριβή ασχολείται με την ανάγκη και την απαίτηση για πράκτορες κοινής λογικής στα ανοικτά υπολογιστικά περιβάλλοντα. Συζητάμε και απεικονίζουμε τις προτάσεις μας αναφορικά με ένα παράδειγμα ηλεκτρονικού εμπορίου. Αρχικά, βάση του πρότυπου σεναρίου, καθορίζουμε τις απαιτήσεις για την αναπαράσταση των κανόνων που διέπουν το περιβάλλον και τις προδιαγραφές του περιβάλλοντος καθαυτού. Ωστόσο, οι προτάσεις μας είναι γενικότερα εφαρμόσιμες σε κάθε περίπτωση όπου πολλαπλοί πράκτορες αλληλεπιδρούν και οι αλληλεπιδράσεις τους καθορίζονται από μια *συμφωνία* που δρα ως μηχανισμός συνεργασίας και συντονισμού.

Πρωτίστως, η διατριβή εξετάζει και αποτελεί κίνητρο της ανάγκης των πρακτόρων να καλύψουν κενά στην πληροφορία που κατέχουν μέσω υποθέσεων. Οι πράκτορες οφείλουν να είναι ικανοί να προσδιορίζουν και να χρησιμοποιούν υποθέσεις δυναμικά, σε κάθε ανοιχτό υπολογιστικό περιβάλλον,

όπως και στο συγκεκριμένο πλαίσιο εφαρμογής του παραδείγματος του ηλεκτρονικού εμπορίου. Παρουσιάζουμε μια νέα προσέγγιση στο δυναμικό προσδιορισμό των υποθέσεων και τον υποθετικό μη-μονότονο συλλογισμό, η οποία είναι εμπνευσμένη από το συντακτικό και της σημασιολογία της Λογικής Προεπιλογής, χωρίς όμως να καταφεύγουμε στην αποδεικτική διαδικασία, η οποία είναι υπολογιστικά, αξιοσημείωτα, δύσκολη. Συζητούμε με λεπτομέρειες τους παράγοντες που διακρίνουν την προσέγγιση μας από άλλες προσεγγίσεις, συγκεκριμένα δεν καταφεύγουμε σε προκαθορισμένα σύνολα υποθέσεων, ούτε χρησιμοποιούμε τους στόχους ενός πράκτορα ως μέσο προσδιορισμού των υποθέσεων του. Με αυτό τον τρόπο, ισχυριζόμαστε πως, ο πράκτορας είναι αυτόνομος στην απόφασή του για το ποιες υποθέσεις είναι κατάλληλες, βάση των απαιτήσεων του σε γνώση-υποθέσεις σε κάθε χρονική στιγμή. Προτείνουμε συμβολικές και σχηματικές αναπαραστάσεις για να χαρακτηρίσουμε τυπικά την πιθανή κατάσταση γνώσης-υποθέσεων του πράκτορα, και χρησιμοποιούμε τις ιδιότητες αυτών για να χαρακτηρίσουμε τη δυναμική τους. Για αυτούς τους τυπικούς χαρακτηρισμούς σχεδιάζουμε και παρουσιάζουμε αλγορίθμους οι οποίοι υποστηρίζουν την προσέγγιση μας, την οποία έχουμε υλοποιήσει σε ένα πρωτότυπο.

Επιπλέον, η διατριβή αυτή ασχολείται και με άλλα θέματα της κοινής λογικής των πρακτόρων, όπως την ανάγκη για χρονικό συλλογισμό και συλλογισμό σχετικά με τις ενέργειες και το αποτέλεσμα αυτών ή τη διαχείριση των κανονιστικών συγκρούσεων. Είναι αναμενόμενο, σε πραγματικά σενάρια, ο πράκτορας να βρεθεί σε θέση όπου θα έχει την ανάγκη να καθορίσει, για τον ίδιο ή άλλους πράκτορες, τι είναι υποχρεωτικό (επιτρεπτό, απαγορευμένο, θεσμικά ισχύον κ.α.) να πράξει σε δεδομένη χρονική στιγμή, να καθορίσει εάν, ο ίδιος ή άλλοι πράκτορες, συμμορφώνονται με τη συμφωνία, και εάν παρατηρηθεί απόκλιση από την υπαγορευμένη συμπεριφορά, να καθορίσει ποιος, εάν υπάρχει, θεραπευτικός μηχανισμός θα μπορούσε να επιστρέψει τη συναλλαγή στην κανονική πορεία, και εν τέλει, να καθορίσει ποίοι κανόνες πυροδοτούν ή τι ισχύει τη δεδομένη χρονική στιγμή, διότι πολλαπλοί, πιθανώς αλληλοσυγκρουόμενοι, κανόνες θα εφαρμόζονται.

ACKNOWLEDGMENTS

Research indicates the interaction, in various ways, with other people, their ideas and viewpoints. Hence, there are many people I would like to thank due to their participation and contribution in this attempt.

First, my supervisor, Dr Aspassia Daskalopulu, who has been always available for discussions, comments and further directions. I want to thank her for the unending discussions on the philosophical aspects of this research, the chance she gave to me to apprehend a particular viewpoint of Computers, Law and Philosophy and study and research on an exciting area of Computer Science. Also, I want to thank her for being there from the beginning of this attempt, i.e. the preparation and writing of a research and project proposal, up to this time point.

My co-supervisors, Dr Catherine Houstis and Dr Nick Bassiliades, and the rest members of the Doctoral Committee, Dr Grigoris Antoniou, Dr Ioannis Vlahavas, Dr Pavlos Peppas and Dr Spyros Lalis, for the meetings and their careful reading and thoughtful comments. The anonymous reviewers who have affected this work through their insightful comments and feedback. The faculty of the Department of Computer and Communications Engineering, at the University of Thessaly, for funding my participation in international conferences and thus provided my the unique opportunity to meet and exchange ideas with academicians and professionals from throughout the world.

Also, I would like to express my gratitude to the faculty of the Department of Physics, the Department of Informatics and the Lab of Medical Informatics, all at the Aristotle University of Thessaloniki, and especially, to Dr Theodoros Samaras, who have showed me the way.

Finally, despite the difficulties and the concerns of a research attempt, my life during these years has been interesting and pleasant thanks to my family and friends. Special thanks to my wife whose patient and encouragement supported me and inspired me to complete this work. My brother, for his dual role as a family member and a colleague. My parents, who have provided me with everything possible and supported me in all my choices. To them I owe the greatest gratitude.

This research was primarily supported by the European Social Fund and the General Secretariat for Research and Technology, Ministry of Development, Hellenic Republic, through the 3rd Community Support Programme.

Supervisor: Dr Aspassia Daskalopulu, Assistant Professor, University of Thessaly

Co-Supervisors: Dr Catherine Houstis, Professor, University of Thessaly
Dr Nick Bassiliades, Assistant Professor, Aristotle University of Thessaloniki

Doctoral Committee: Dr Grigoris Antoniou, Professor, University of Crete
Dr Catherine Houstis, Professor, University of Thessaly
Dr Ioannis Vlahavas, Professor, Aristotle University of Thessaloniki
Dr Pavlos Peppas, Associate Professor, University of Patras
Dr Nick Bassiliades, Assistant Professor, Aristotle University of Thessaloniki
Dr Aspassia Daskalopulu, Assistant Professor, University of Thessaly
Dr Spyros Lalis, Assistant Professor, University of Thessaly

To my parents and my family.

Στους γονείς μου και την οικογένεια μου.

One does not discover new lands
without consenting
to lose sight of the shore for
a very long time.[1,2]

André Gide

Κανείς δεν ανακαλύπτει νέα εδάφη
αν δεν είναι πρόθυμος
να απομακρυνθεί από την ακτή
για μεγάλο χρονικό διάστημα.

Αντρέ Ζιντ

---

[1] L. Minkin, Exits and Entrances: Political Research as a Creative Art, Sheffield Hallam University Press, 1997

[2] P. Dunleay, Authoring a PhD. How to plan, draft, write and finish a Doctoral Thesis or Dessertetion, Palgrave Macmillan, 2003

TABLE OF CONTENTS

PART I PREFACE **1**

1 INTRODUCTION **3**

1.1 MOTIVATION 4
1.2 OBJECTIVES 5
1.3 INTRODUCTORY DEFINITIONS AND CONSIDERATIONS 7
1.4 THESIS OUTLINE 9

PART II BACKGROUND **13**

2 ARTIFICIAL INTELLIGENCE, AGENCY AND LAW **15**

2.1 INTRODUCTION 15
2.2 INTELLIGENT AGENTS AND MULTI-AGENT SYSTEMS 16
2.2.1 Autonomy in the context of Agency 16
2.2.2 Open Systems 18
2.2.3 The Closed World Assumption and the Open World Assumption 19
2.3 AGENCY AND LAW 21
2.3.1 Normative Systems and Deontic Logic 21
2.3.2 e-Contracts 22
2.4 SUMMARY 25

3 COMMON-SENSE REASONING **27**

3.1 INTRODUCTION 27
3.2 REASONING ABOUT TIME, ACTIONS AND EFFECTS 29
3.2.1 Event Calculus 29
3.3 NON-MONOTONIC REASONING 31
3.3.1 Default Logic 32
3.3.1.1 Skeptical and Credulous Reasoning 33
3.3.1.2 Operational Computation of Extensions 34
3.3.2 Variations of Default Logic 38
3.3.2.1 Normal – Semi-normal Default Theories 38
3.3.2.2 Constrained Default Theories 39
3.3.2.3 Pre-constrained Default Theories 40
3.3.2.4 Stratified Default Theories 40
3.3.2.5 Prioritized Default Theories 42
3.4 SUMMARY 43

PART III AN E-CONTRACT FRAMEWORK **45**

4 APPLICATION AREA ANALYSIS **47**

4.1 INTRODUCTION 47
4.2 E-CONTRACT EXAMPLE SCENARIA 48
4.2.1 A three-party business transaction 51
4.3 REQUIREMENTS ANALYSIS AND SPECIFICATION 56
4.4 RELATED WORK 60
4.4.1 E-commerce 61
4.4.1.1 Dynamic Logic 61
4.4.1.2 Event Calculus 62
4.4.1.3 Non-monotonic Reasoning 63
4.4.1.4 Commitments 66
4.4.1.5 Linguistic Aspects of e-Contracts 69
4.4.1.6 Hypothetical Reasoning 70
4.4.2 E-business 71
4.4.3 Virtual Communities 73
4.5 SUMMARY 75

5 REASONING WITH INCOMPLETE KNOWLEDGE......77

5.1 INTRODUCTION......77
5.2 THE REASONING PROBLEM......78
5.3 WHY DEFAULT LOGIC?......81
  5.3.1 Default Logic for Common-sense Reasoning in the Context of Open Environments......82
  5.3.2 Dynamic Default Logic......85
5.4 SUMMARY......89

6 ASSUMPTION-BASED REASONING IN OPEN NORMATIVE ENVIRONMENTS......91

6.1 INTRODUCTION......91
6.2 PRELIMINARIES......92
6.3 THEORY CONSTRUCTION AND INFERENCE......94
  6.3.1 Rule mapping......94
  6.3.2 Inference on the basis of Assumptions......97
  6.3.3 Normal and Semi-Normal Default Theories......103
6.4 PROTOTYPE IMPLEMENTATION......106
6.5 EXAMPLE......109
6.6 SUMMARY......113

7 AUTONOMOUS HYPOTHETICAL NON-MONOTONIC REASONING......115

7.1 INTRODUCTION......115
7.2 ASSUMPTION IDENTIFICATION......117
7.3 THEORY CONSTRUCTION......122
7.4 INFERENCE PROCEDURE......123
7.5 MANAGING THE SPACE OF KNOWLEDGE/HYPOTHESIS......129
  7.5.1 Hypotheses Rationality and Consistency......130
  7.5.2 Hypotheses Restriction......131
  7.5.3 Hypotheses Sequence......133
7.6 EXAMPLES......135
  7.6.1 General Example......136
  7.6.2 Business Example......138
    7.6.2.1 Dealing with information gaps (1H)......138
    7.6.2.2 Commitment to Assumptions (2H)......146
    7.6.2.3 Non-monotonic Reasoning (3H)......147
7.7 SUMMARY......148

8 RELATED WORK......151

8.1 INTRODUCTION......151
8.2 RESEARCH ON THE DYNAMICS OF DEFAULT THEORIES......152
8.3 RESEARCH ON ASSUMPTION-BASED REASONING AND NONMONOTONIC REASONING......152
  8.3.1 Assumption-based Reasoning......153
    8.3.1.1 Static Assumption-based Reasoning......153
    8.3.1.2 Dynamic Assumption-based Reasoning......156
  8.3.2 Non-monotonic Reasoning......160
    8.3.2.1 Closed World Assumption......160
    8.3.2.2 Circumscription......161
    8.3.2.3 Logic Programs......162
    8.3.2.4 Defeasible Logic......163
8.4 RESEARCH ON AUTONOMY-ORIENTED REASONING......163
8.5 SUMMARY......164

9 CONFLICT MANAGEMENT......165

9.1 INTRODUCTION......165
9.2 PRELIMINARIES......167
9.3 NORMATIVE CONFLICT PATTERNS......169
  9.3.1 Primitive Patterns of Normative Conflicts......169
  9.3.2 Other analyses of normative conflicts......172
    9.3.2.1 Policy-based Conflicts......173

9.3.2.2 Role-based Conflicts 173
9.3.2.3 Conflicts related to Interest/Duty 174
9.3.2.4 Exceptions 175
9.3.2.5 Temporal Normative Conflicts 175
9.3.3 Additional Patterns....176
9.3.3.1 Type of action-based conflicts 177
9.3.3.2 Agreement-based conflicts 177
9.3.3.3 Conflicts between assumptions and knowledge 178
9.4 CONFLICT DETECTION 178
9.4.1 Conflict Detection with Default Logic....178
9.4.2 Patterns of Normative Conflicts Represented and Detected via Defaults 183
9.5 CONFLICT RESOLUTION 188
9.6 CONFLICTS AND ASSUMPTIONS....192
9.7 RELATED WORK AND SUMMARY 193

10 OTHER ISSUES FOR COMMON-SENSE REASONING 197

10.1 INTRODUCTION 197
10.2 REASONING WITH TIME, ACTION AND DEONTIC MODALITIES 198
10.3 EXAMPLE 200
10.3.1 Hypothetical Reasoning (1H) 202
10.3.2 Commitment to Assumptions (2H)....204
10.3.3 Non-monotonic Reasoning (3H)....205
10.3.4 Conflict Management 205
10.4 RELATED WORK AND SUMMARY 207

PART IV CONCLUSIONS AND FUTURE WORK 209

11 CONCLUSIONS....211

11.1 SUMMARY....212
11.2 CONTRIBUTIONS AND WORK IN CONTEXT 216
11.3 FUTURE WORK 220

PART V APPENDICES BIBLIOGRAPHY AND INDEX....223

APPENDIX A SURVEY SUMMARY ON E-CONTRACTING FRAMEWORKS....225

APPENDIX B STANDARD DEONTIC LOGIC AND CONFLICT MANAGEMENT 237

BIBLIOGRAPHY 239

LIST OF FIGURES

FIGURE 2.1 E-CONTRACT LIFE CYCLE....24
FIGURE 3.1 SKEPTICAL AND CREDULOUS REASONING....34
FIGURE 3.2 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 1....35
FIGURE 3.3 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 2....36
FIGURE 3.4 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 3....37
FIGURE 3.5 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 4....39
FIGURE 3.6 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 5....40
FIGURE 3.7 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 6....41
FIGURE 3.8 OPERATIONAL COMPUTATION OF EXTENSIONS - EXAMPLE 7....42
FIGURE 4.1 CASE A: CONTRACTS BETWEEN BUYER – RETAILER AND RETAILER – CARRIER....49
FIGURE 4.2 CASE B: CONTRACTS BETWEEN BUYER – RETAILER, RETAILER – CARRIER, RETAILER - WHOLESALER AND WHOLESALER – CARRIER....49
FIGURE 4.3 CASE C: CONTRACTS BETWEEN BUYER – MEDIATOR, MEDIATOR – WHOLESALER, BUYER - WHOLESALER AND WHOLESALER – CARRIER....50
FIGURE 4.4 CASE D: CONTRACTS BETWEEN BUYER – MEDIATOR, MEDIATOR – WHOLESALER, BUYER – WHOLESALER, WHOLESALER – CARRIER, WHOLESALER – IMPORTER AND IMPORTER - CARRIER 51
FIGURE 4.5 A THREE-PARTY BUSINESS TRANSACTION....52
FIGURE 4.6 E-CONTRACT TRANSITION DIAGRAM....55
FIGURE 4.7 PART OF THE E-CONTRACT TRANSITION DIAGRAM....56
FIGURE 5.1 AGENT REASONING IN OPEN ENVIRONMENTS....84
FIGURE 5.2 EXTENSIONS IN AN OPEN ENVIRONMENT....85
FIGURE 5.3 ALGORITHM 1: DEFAULT REASONING IN OPEN ENVIRONMENTS....87
FIGURE 6.1 SINGLE-NORM KH STRUCTURE OF AN AGENT’S MENTAL STATES....95
FIGURE 6.2 MULTI-NORM KH STRUCTURE OF AN AGENT’S MENTAL STATES....98
FIGURE 6.3 PROTOTYPE ARCHITECTURE....106
FIGURE 7.1 EXPANSION/CONTRACTION OF KH STATUS....118
FIGURE 7.2 EXPANSION/CONTRACTION OF KH STATUS FOR RULE R1....122
FIGURE 7.3 EXPANSION/CONTRACTION OF KH STATUS FOR RULE R2....123
FIGURE 7.4 ALGORITHM 6: LIFE-CYCLE OF REASONING IN OPEN ENVIRONMENTS....125
FIGURE 7.5 INITIALIZATION....127
FIGURE 7.6 REPOSITION....128
FIGURE 7.7 REPOSITION....128
FIGURE 7.8 BA’S POSITION FOR NORMS R1 AND R2....140
FIGURE 7.9 BA’S POSITION FOR NORMS R1 AND R2....141
FIGURE 7.10 SA’S POSITION FOR NORMS R1, R2 AND R3....145
FIGURE 7.11 BA’S INITIAL POSITION FOR NORMS R1, R2 AND R3....146
FIGURE 7.12 BA’S REPOSITION FOR NORMS R1, R2 AND R3....147
FIGURE 7.13 BA’S REPOSITION FOR NORMS R1, R2 AND R3....148
FIGURE 9.1 TIME INTERVAL-BASED CONFLICTS....176
FIGURE 9.2 DETECTION OF NORMATIVE CONFLICTS - CASE 1....179
FIGURE 9.3 DETECTION OF NORMATIVE CONFLICTS - CASE 2....180
FIGURE 9.4 DETECTION OF NORMATIVE CONFLICTS - CASE 3....181
FIGURE 9.5 DETECTION OF NORMATIVE CONFLICTS - CASE 4....181
FIGURE 9.6 BUYER - BUYER - SELLER EXAMPLE SCENARIO....191

LIST OF TABLES

TABLE 3.1 THE EVENT CALCULUS PREDICATES 29
TABLE 3.2 THE EVENT CALCULUS DOMAIN-INDEPENDENT AXIOMS 30
TABLE 4.1 NORMATIVE POSITIONS FOR CONTRACTING PARTIES 53
TABLE 4.2 REQUIREMENTS AND SPECIFICATIONS...57
TABLE 6.1 ALGORITHM 2: COMPUTATION OF THE SINGLE-NORM KH STRUCTURE....................................96
TABLE 6.2 ALGORITHM 3: INFERENCE PROCEDURE...99
TABLE 6.3 ALGORITHM 4: COMPUTATION OF EXTENSIONS 100
TABLE 6.4 XML DOCUMENT THAT REPRESENTS THE SINGLE-NORM KH STRUCTURE 107
TABLE 7.1 ALGORITHM 5: COMPUTATION OF THE NEIGHBOUR NODES...121
TABLE 9.1 PRIMITIVE CONFLICT PATTERNS..169
TABLE 9.2 RESOLUTION STRATEGIES FOR NORMATIVE CONFLICT PATTERNS 190
TABLE B.1 CONFLICT PATTERNS WITH AND WITHOUT SDL AXIOMATIZATION 238

LIST OF MAIN ACRONYMS

AI	Artificial Intelligence
AS	Agent Systems
BDI	Belief-Desire-Intention
CDfL	Constrained Default Theory
CSK	Common-Sense Knowledge
CSR	Common-Sense Reasoning
CTD	Contrary To Duty
CWA	Closed World Assumption
DAI	Distributed Artificial Intelligence
DfL	Default Logic
DfT	Default Theory
DDfL	Dynamic Default Logic
FOL	First-order Logic
EC	Event Calculus
K/H	Knowledge/Hypothesis
MAS	Multi-agent Systems
NMR	Non-monotonic Reasoning
NDfT	Normal Default Theory
OAS	Open Agent Systems
ODA	Open Default Assumption
OMAS	Open Multi-agent Systems
OS	Open Systems
OWA	Open World Assumption
PcDfL	Pre-constrained Default Theory
PL	Propositional Logic
SDL	Standard Deontic Logic
SDfT	Stratified Default Theory
SnDfT	Semi-normal Default Theory

Part I
Preface

1 Introduction

An observation on the recent call for papers either for international conferences or special issues in scientific journals in the broader area of Artificial Intelligence and Computer Science signifies a wide and intense interest on issues such as:

- Heterogeneous and dynamic systems. The Web and the Multi-agent Systems are two characteristic examples, though not the only ones, of such environments.
- Autonomous and rational decision making, action and behaviour of system components. Commercial and business applications and applications on virtual communities are some characteristic examples where the need of such reasoning is essential.
- Self-knowledge, self-regulation, self-organization and self-management. Autonomic or Ubiquitous Computing and Adaptive Systems are characteristic examples that behave in this manner.

This thesis addresses such requirements by presenting representation and reasoning approaches and techniques for autonomous rational agents consisting in open computational environments that are governed by norms.

This chapter is organized as follows: section 1.1 describes the motivation of this work; section 1.2 presents the objectives of this thesis and it discusses briefly its contributions; section 1.3, provides a brief introduction to the major definitions and considerations that are either adopted or proposed in this thesis; finally, section 1.4 provides an outline of the rest of this thesis.

1.1 Motivation

This thesis concerns with symbolic knowledge representations and common-sense reasoning techniques for intelligent autonomous software agents that populate open computational environments. An environment is characterized as *open*, in Hewitt’s sense [118], when it is dynamic, continuous, unobservable or partially observable and non-deterministic.

Agents is such environments possess, unavoidably, information that is incomplete, imprecise, maybe even incorrect, due to the very fact that the environment is open and, at the very least, agents join and leave it as they please. The information exchanged between agents may be delayed, or distorted by noise during its communication, and in any case, as the environment evolves, it is bound to change.

The interactions among agents in any multi-agent system are typically governed by norms. These may refer to restrictions on communication means among agents, or to particular coordination mechanisms, liveness and safety properties of the system etc. In some application areas, such as e-commerce, additional norms may regulate the agents’ behaviours, resulting from agreements into which the agents enter willingly, and possibly from the protocol that governs the e-market. Norms prescribe what each agent is obliged, permitted, prohibited, empowered and so on, to do during its life in the particular environment. Autonomous agents decide for themselves which norms to subject itself to and whether to comply with the norms of their environment. This decision-making is all the more challenging when an agent has to perform it in circumstances where its available knowledge is incomplete/imprecise/incorrect.

As a result, the specific characteristics of such environments comprise the motivation of the work presented in this thesis. Particularly, in such environments:

- An agent will need to establish at a given time point, the state of its transaction with other agents, in order to check and regulate its own behaviour with respect to the commitments it has engaged itself towards other agents, and plan its activities accordingly.

- It is expected that agents will find themselves in a situation where multiple, possibly conflicting, norms apply. The agent will need some way to detect that such conflicts exist, so that subsequently, it may deploy some resolution mechanism, in order to infer what it should do at a given point in time.
- Unavoidably in open environments agents have incomplete knowledge about their world, and about other agents, yet they must somehow plan their activities, and they must somehow preserve their autonomy and rationality, i.e. decide for themselves which behaviour serves their private or shared goals in the best way. We believe that the degree of agent autonomy is related to the extent to which an agent is ‘free’ to make assumptions about anything it does not know about, either these concern the future or the past and present. We see that, it is essential to support, first, dynamic assumption identification and usage, without a priori restrictions on the agent, and without resorting to proof, which is prohibitive computationally, and second, non-monotonicity when information that becomes available at some time point confirms or disproves assumptions made at previous times, and consequently conclusions drawn on their basis.

1.2 Objectives

This thesis addresses the need and requirement of agents for common-sense reasoning and knowledge representation in open computational environments. We discuss and illustrate our proposals with reference to an e-commerce example scenario, although our proposals are generally applicable to any case where multiple agents interact and their interaction is governed by some *contract* as a coordination or collaboration mechanism.

First, we aim to analyse the application area and to specify requirements for both knowledge representation and reasoning with norms that govern the environment. Towards this scope we identify several specifications that the environment and its components should meet. The rest of this thesis, concerns either with addressing directly some of these requirements via the introduction of novel approaches for representation and novel techniques for reasoning or with discussing the way our proposals are also capable in addressing other requirements. Through this attempt we have surveyed related literature. Thus, a critical review of other research approaches towards e-contracting, dynamics of logic theories, hypothetical reasoning, non-monotonic reasoning and autonomous reasoning is presented throughout this work.

Primarily, this thesis aims to examine and motivate the need of agents to fill in information gaps by resorting to assumptions. Agents need to be able to identify and use assumptions dynamically, in any open computational environment, as well as in the particular application context of an e-commerce example. Towards this scope, we present a novel approach to dynamic assumption identification and hypothetical non-monotonic reasoning inspired by the syntax and semantics of Default Logic, without however resorting to proof, which is notably computationally hard. We discuss in detail what distinguishes our approach from other work on dynamic assumption based reasoning, namely that we do not resort to a pre-specified pool of assumptions, nor to goal-orientation as a means to identify candidate assumptions. We, also, propose symbolic and schematic representations to characterize formally the possible knowledge-hypotheses status and knowledge-hypotheses space of an agent, and we use their properties to characterize their dynamics. From these formal characterizations we derive and present the algorithms that support our approach, which we have implemented in a prototype. Furthermore, we concern, particularly, with the management of an agent’s knowledge/hypotheses space and reasoning procedure. In this way, we claim, an agent, first, is autonomous in deciding which assumptions are appropriate to fill in information gaps, and, second, is rational in its reasoning, given its knowledge-hypotheses requirements at any given time.

Moreover, this thesis aims to address other issues of common-sense reasoning agency, such as the need for temporal reasoning and reasoning about actions and their effects or the need for normative conflict management. It is expected that in realistic scenaria an agent will find itself in a situation where it needs to establish, for itself or other agents, what is obliged (permitted, prohibited, empowered etc.) to do at a given point in time; to determine whether, itself or other agents, comply with the agreement; whether deviation is detected from the prescribed behaviour, to determine what, if any, remedial mechanisms might return the transaction to a normal course; and finally, to establish which norms apply or what holds at a given time point, because multiple, possibly conflicting, norms apply. To this end, we claim and show how our proposals towards hypothetical non-monotonic reasoning, first, facilitate conflict management, i.e. conflict detection and resolution, and, second, is capable for reasoning with time, actions and deontic modalities.

To sum up, this thesis aims to address the requirement for reasoning in open environments by presenting a proposal where agents: have self-knowledge of the limits of their knowledge; are capable of self-regulation, i.e. to develop for themselves the norms according to which they regulate their behaviour; and are, also, capable of self-management, i.e. to make their own reasoning and to rely on their own knowledge and strategy.

1.3 Introductory Definitions and Considerations

Autonomous Agents: In this thesis we take the most recent perspective on autonomy and address it relevant to an agent's reasoning process, within the context of an open computational system. In this context, our agent is expected to make inferences about which beliefs to adopt about its environment, other agents and norms in force, which goals to commit to, and which actions to perform, in the presence of incomplete or inconsistent information, and it is expected to be independent from external intervention in this reasoning process. Specifically, in this thesis we examine the relation between an agent's ability to identify and employ assumptions independently, and its autonomy. We claim that an agent that answers the reasoning problem, addresses also the autonomy problem and present techniques that enable agents to 'develop for themselves the laws and strategies according to which they regulate their behaviour (in the spirit of [234]) and to 'make their own inferences and reasoning and to rely on their own conclusions' (in the spirit of [40]). A detailed discussion is available in chapters 2 and 7.

Open Multi-agent Systems: In this thesis, our motivation is influenced by Hewitt's and de Jong's work on Open Systems. They see agents that [122]:

> “…need to have self-knowledge in order to function effectively in Open Systems in order to understand its own abilities as well as the limits of its knowledge and power. As knowledge is added incrementally to a subsystem, it must relate the new knowledge to its existing knowledge. Any subsystem can have only partial knowledge of the overall system, and partial power to affect other subsystems.”

We accept, that agents in Open Multi-agent Systems possess, unavoidably, incomplete information about the environment and other agents in it, have self-knowledge of the limits of their knowledge and need to reason in the presence of information gaps via self-management and self-regulation. A detailed discussion is available in chapter 2 and 7.

Operational Computation of Extensions: In this thesis, in order to facilitate common-sense reasoning in an open world where the environment and whatever is associated with it are matters of continuous change, we do not require or expect from the agent to either accept worlds that belong to the distance future or reject them, in advance. It seems rational and realistic to us, to require from an agent to reason in a process-oriented and step-wise manner and on the basis of its current knowledge and some plausible and rational assumptions. Such an agent would compute all possible worlds, examine the state that these worlds reveal (consistent/inconsistent, rational/irrational, eligible/non-eligible, desirable/non-desirable etc) and finally decide to either to follow or not a specific course of action. In other words, we see that common-sense reasoning in open computational systems calls for an iterative reasoning process where the agent gains knowledge about its current state in the available world, commit itself to this world, produce entailments either on factual or on hypothetical basis and re-examine its world for any changes due to exogenous or endogenous factors. A detailed discussion is available in chapters 3 and 5.

Open Default Assumption: The *Open Default Assumption* is the presumption that the truth-value of a statement that is not currently known may considered to be *true* if this does not cause an inconsistent view of the world. It refers, not only, to the ability of agents to identify and employ assumptions dynamically, on the basis of their current knowledge, but also to their ability to manage their inference on the basis of these assumptions and any available knowledge at previous or future time points. It is inspired by the syntax and semantics of DfL and uses some of its several variations along with *Dynamic Default Logic*, i.e. a variation of DfL presented here, towards dynamic hypothetical non-monotonic reasoning. A detailed discussion is available in chapter 5.

Dynamic Default Logic: *Dynamic Default Logic* accepts the Open Default Assumption. With Dynamic Default Logic it is possible to reformulate, appropriately and when needed, the initially given rules. As a result, inferencing is possible on a totally hypothetical basis via the dynamic identification and employment of appropriate candidate assumptions, i.e. *extended* versions of extensions are possible to be computed. A detailed discussion is available in chapter 5.

e-Contracts: The term *contract* is used to refer both to a legally binding agreement between two or more parties and to the *document* that records such an agreement. A contract creates mutual legal relations between the parties involved and determines what actions are obligatory/permissible/forbidden to be performed by the parties. The term *e-contract* is used to refer to the electronic document that is determined by reference to an individual computer-generation transaction. The term *e-contracting* is used to encompass all activities concerned with this transaction and e-contracts [9, 54, 150]. The whole contractual life cycle comprises of two different phases, namely *contract formation* and *contract performance*. In this thesis we broadly refer to three application sub-areas, where electronic agreements play a central role: e-commerce, i.e. purchase contracts, business process modelling and automation, i.e. cooperation contracts, and virtual communities, i.e. social contracts. A detailed discussion is available in chapters 2 and 4.

1.4 Thesis Outline

This thesis consists of four core sections. The first section, i.e. chapters 2 and 3, provides a review of the background of this work. In chapter 2, we discuss intelligent agents, multi-agent interactions, and the systems where agents interact. These systems are considered as open computational environments that are governed by norms. In chapter 3, we discuss logic languages and reasoning patterns that we exercise in this thesis in order to address the issue of common-sense reasoning in open norm-governed multi-agent environments. Moreover, during this review we also discuss research questions that arise in these settings and motivate our viewpoint.

The next sections present the main contributions of this thesis. Specifically, the second section, i.e. chapter 4, provides an analysis of the application area of this thesis. First, we introduce several example scenaria of this area. These examples motivate a discussion about the requirements and specifications for common-sense reasoning in open environments, and to this end we introduce and discuss a list of requirements for representation languages and specifications for the development of tools. This chapter concludes, with a critical review on previous logic-based approaches towards e-contract modelling and performance monitoring.

The third section, i.e. chapters 5, 6, 7 and 8, introduces our approach to common-sense reasoning with incomplete knowledge. In chapter 5, we discuss the reasoning problem by identifying several research questions, and our viewpoint for reasoning within the context of open environments. Chapter 6 presents a first approach to assumption-based reasoning in open normative systems and discusses issues towards implementation by presenting algorithms and a prototype. Chapter 7 discusses the limitations of the first approach and presents a second one towards autonomous hypothetical non-monotonic reasoning. This chapter, mainly concerns with the management of an agent’s knowledge space and reasoning procedure. We claim that, in this way we enhance the agent’s autonomous and rational behaviour and enable its reasoning within open environments when it possesses incomplete knowledge. Finally, chapter 8 consists of a critical review on various approaches, found in the literature, and the way these approaches address the issues discussed in the previous chapters 5, 6 and 7, such as the dynamics of logic theories, assumption-based reasoning, non-monotonic reasoning, and autonomous agency.

The fourth section, i.e. chapters 9 and 10, explain the way our primary proposal on hypothetical non-monotonic reasoning is also suitable to address other various issues of interest, such as conflict management and temporal reasoning. Specifically, in chapter 9, we address the agent’s need to detect normative conflicts and to deploy some resolution mechanism, in order to infer what it should do at a given point in time. Chapter 10 discusses how our proposal facilitates temporal reasoning, reasoning about actions and their effects and reasoning with deontic modalities.

Finally, chapter 11 provides a summary and discussion of the thesis core chapters, presents a listed view of the contributions of this thesis and its relation to other research approaches, and concludes with directions for future research on both theoretical and practical issues discussed in this thesis.

Part II
Background

2 Artificial Intelligence, Agency and Law

2.1 Introduction

Russell and Norvig in [215] note about the different fields of Artificial Intelligence (AI):

> “The main unifying thing is the idea of an intelligent agent. We define AI as the study of agents that receive percepts from the environment and perform action.”

The notion that relates agents with AI is *rationality*. Various definitions for AI where based on this relation [215]:

- “The study of mental faculties through the use of computational models.” [42]
- “The study of the computations that make it possible to perceive, reason and act.” [253]
- “Computational Intelligence is the study of the design of intelligent agents.” [197]
- “AI … is concerned with intelligent behaviour in artifacts.” [186]

In the next sections we discuss rational agents (or, in other words, intelligent agents), multi-agent interactions, and the environments where agents interact. We see agent environments as open systems that are governed by norms and discuss research questions that arise in these settings.

2.2 Intelligent Agents and Multi-agent Systems

The concept of *autonomy* is central to the notion of agent-hood and has been addressed by many researchers in the AI community. Wooldridge and Jennings in [255, 254] note:

> "An agent is a computer system that is situated in some environment and that is capable of autonomous action in this environment in order to meet its design objectives."

This definition reflects various properties of software agents. An agent *perceives* the environment in which it is situated, and *interacts* with it by performing certain *actions* towards its design *goals*. Additionally, following this definition, results that a rational agent is an agent that does the "right thing" given what it knows [215].

Of course, agents interact with other agents. Today's AI, i.e. Distributed Artificial Intelligent (DAI), concerns with the study and the applications of Multi-agent Systems (MAS). A MAS is an electronic society of software agents which *communicate* and interact with each other. Each agent (or, groups of agents) represents/acts on behalf of different representatives, i.e. agents have different *roles*. This means that, agents may have the same interests and act in *cooperation*, or may have different (possible conflicting) interests and act as *competitors*. In both cases interactions and goal achievement requires *negotiation*. Moreover, in such environments, agents may need to be equipped with higher *utilities* (utility factor) beside their goals, or even with what is called an *internal state*, i.e. *beliefs*, *desires* and *intentions* (BDI).

2.2.1 Autonomy in the context of Agency

Etymologically the term 'autonomy' (auto=self + nomos=law) refers to the ability of an entity to choose its own norms and regulate its own behaviour accordingly. In common usage the term 'autonomy' is defined as the quality or state of being self-governing (especially the right of self-government), self-directing freedom (especially moral independence), and a self-directing state [67]**.** Since autonomy is a key characteristic of an agent, all researchers in the MAS community address this issue. A study of the relevant literature during the past twenty years, suggests that autonomy has been examined from various perspectives, initially in relation to an agent's goals, and more recently in relation to an agent's reasoning. Specifically, in [21, 32, 78, 168, 255, 160, 169, 38, 40, 161, 18, 224] autonomy is defined as freedom from *external* intervention or control, where 'external' refers to humans and/or other agents. In early work [78, 81, 160] this freedom from external intervention is specifically examined in relation to an agent's ability to *choose* its own goals and to act towards them independently. Barber and Martin in [19] distinguish external interventions that affect an agent's *environment*, from those that affect its *beliefs*, and moreover from those that affect its *decision-making process*, and consider independence from the latter 'as the most salient dimension of the concept of autonomy'; according to Barber and Martin an agent's decision-making process establishes how the agent should pursue a particular goal, i.e. it is action-oriented. Castelfranchi [40] and Verhagen [246] take a more general view of autonomy and relate it to an agent's *reasoning*, i.e. to its ability *to make its own inferences and to rely on its own conclusions*, where such inferences may result in choosing an action to perform, or a belief or goal to adopt, or establishing and evaluating motives for a particular course of action. Moreover, Castelfranchi [39] considers the concept of *initiative* as important and relevant to an agent's autonomy. Finally, in [50, 247] autonomy is examined in relation to an agent's ability to choose which *norms to subject itself to*, and to decide whether to comply with them or not.

In this thesis we take the most recent perspective on autonomy and address it relevant to an agent's reasoning process, within the context of an *open* system. In this context, our agent is expected to make inferences about which beliefs to adopt about its environment, other agents and norms in force, which goals to commit to, and which actions to perform, in the presence of incomplete or inconsistent information, and it is expected to be independent from external intervention in this reasoning process. A detailed discussion is available in chapters 5 and 7.

2.2.2 Open Systems

Open Systems (OS), in the sense of Hewitt and colleagues [122, 123, 118, 119, 120, 121], that consist mainly of computers, users and software, are characterized of the following properties:

- **Continuous Change and Evolution**: Always new components, such as computers, users and software are being added in the systems. Thus, systems should be able to change their components and even more important, to evolve their components in order to perform their work.
- **Asynchrony and Late Arriving Information**: Asynchrony enables the components to operate on the basis of local circumstances. When late-arriving information becomes available, components are able to take into account the new information and affect their decision making process.
- **Inconsistent Information**: Information that becomes available to the components, either form outside or from inside the system, may be inconsistent.
- **Arm's-length relationships**: The internal operation, organization and state of computers, users and software are not always known to other computers, users and software.
- **Decentralized control**: The above properties call for some kind of decentralized and distributed decision making mechanism.
- **Negotiation**: Components, i.e. computers, users or software, can not control or use directly the resources of another component. A negotiation mechanism is imperative to support the interchange of resources among the components.

AS and MAS are considered to be open systems, i.e. Open Agent Systems (OAS) and Open Multi-agent Systems (OMAS), respectively. Thus, OAS and OMAS are: dynamic (agents may join or leave the system at any given time), non deterministic, continuous, unobservable (or, at best, partially observable), heterogeneous, and finally, their members do not share a global utility, i.e. members may work towards different (possible conflicting) directions.

In this thesis, our motivation is influenced by Hewitt's and de Jong's work on OS. They see agents that [122]:

> "…need to have self-knowledge in order to function effectively in Open Systems in order to understand its own abilities as well as the limits of its knowledge and power. As knowledge is added incrementally to a subsystem, it must relate the new knowledge to its existing knowledge. Any subsystem can have only partial knowledge of the overall system, and partial power to affect other subsystems."

We accept, that agents in OMAS possess, unavoidably, incomplete information about the environment and other agents in it, have self-knowledge of the limits of their knowledge and need to reason in the presence of information gaps via self-management and self-regulation.

2.2.3 The Closed World Assumption and the Open World Assumption

In order to support reasoning with incomplete knowledge we could adapt one of the general presumptions called the Closed World Assumption (CWA) [207] or Open World Assumption (OWA).

Under the CWA the following presumption is conceded:

> "An atomic formula is assumed false, unless it is explicitly known to be true."

An agent that uses its incomplete representation essentially admits into its knowledge base negative literals $\neg A$ that correspond to assumptions it makes under CWA about the falsity of certain atomic formulae A if A cannot be proved from its current knowledge base and continues its inference.

Under the OWA the following presumption is conceded:

> "An atomic formula is assumed to be unknown by default."

In other words, the OWA limits an agent's inference ability only on the basis of statements that are explicitly known to the agent to be true, contrary to the CWA that expands an agent's inference ability by considering statements that are not currently known to be true as false.

For example, according to the knowledge base:

$$\{ \mathrm{Has(Eagle,Feathers)},\ \mathrm{Lays(Penguin,Egges)},\ \mathrm{Is}(x,\mathrm{Bird}) \leftarrow \forall x\ (\mathrm{Has}(x,\mathrm{Feathers}) \wedge \mathrm{Lays}(x,\mathrm{Eggs}))\}$$

for an agent, in order, to entail that the eagle or the penguin is a bird, must decide on whether an eagle lays eggs or a penguin have feathers. Under the CWA answers to these questions are negative, i.e., Has(Penguin,Feathers), Lays(Eagle,Eggs) truth values are set to *false*. Under the OWA answers to these questions are unknown, i.e., Has(Penguin,Feathers), Lays(Eagle,Eggs) truth values are set to *unknown*.

Hewitt and de Jong in [122] note:

> "…the 'closed world assumption' is intrinsically contrary to the nature of Open Systems. … Systems based on the 'closed world assumption' typically assume that they can find all the instances of a concept that exist by searching their local storage. In contrast we desire that subsystems be accountable for having evidence for their belief and explicitly aware of the limits of their knowledge."

In other words, agents under the CWA are *atheist* agents which dictate that an information gap should be treated as negative information. In the same spirit, agents under the OWA are *agnostic* agents which treat information gaps as potentially positive or negative information. In many realistic scenaria, however, this indecisiveness is undesirable and, furthermore, it is important to be able to make assumptions about the *truth* (rather than the falsity) of certain formulae, i.e., to treat information gaps for what they are (absence of definite information) as potentially positive information.

In this thesis, in order to facilitate reasoning with incomplete and inconsistent knowledge, we adopt agents of none of the above two kinds. We see that agents should be able to identify and employ appropriate hypotheses dynamically, especially when the agents are engaged in business transactions and may violate a norm that regulates their behaviour, inadvertently. Specifically, we are interested in situations where the agent needs to make *specific* assumptions about the *truth*, rather than the falsity, of certain formulae. We claim that such reasoning may be useful in two cases:

- *Best-guess reasoning*: An agent cannot know the future, yet it may need to plan its activities on the basis of hypotheses that concern the future.

- *No-risk reasoning*: An agent may not know everything about the past and present, i.e., the history of its environment, other agents and itself so far, yet it may need to plan its activities on the basis of hypotheses that concern the past and present.

2.3 Agency and Law

Several approaches have been proposed in order to provide formal tools for the specification, modelling, monitoring and execution of MAS. Normally, approaches are based on ideas and follows directions of the design-by-contact approach on software engineering. We classify these approaches into three application areas: (i) normative systems, (ii) electronic institutions and (iii) virtual organizations. Typically, all approaches involve a representation of deontic notions (such as obligation, permission, prohibition and power), their associated meta-level notions (such as violation, sanction, compliance and normative conflict) and domain-independent concepts such as time, actions and their effects. In this section we discuss, briefly, issues that concern these approaches, and specifically, e-contracts which is the application area of this thesis.

2.3.1 Normative Systems and Deontic Logic

Normative systems can be considered as the intersection of Law and AI, and therefore, OMAS may be viewed as instances of normative systems. Jones and Sergot in [135] note about normative systems:

> “We use the term to refer to any set of interacting agents whose behaviour can usefully be regarded as governed by norms. Norms prescribe how the agent ought to behave and specify how they are permitted to behave and what their rights are. Agents may be human individuals or collection of human individuals, or computer systems or collection of computer systems. Normative systems include systems of law, abstract models of computer systems, and hybrid systems consisting of human and computer agents in interaction.”

In the same spirit Carmo and Jones in [36] note:

> “… sets of agents whose interactions are norm-governed; the norms prescribe how the agents ideally should and should not behave what they are permitted to do and what they have right to do. Importantly, the norms allow for the possibility that actual behaviour may at times deviate from the ideal, i.e. that violations of obligations, or of agents’ rights, may occur.”

Jones and Sergot argued for the need for a formal representation of such environments, i.e. a representation capable to express the distinction between the actual and the ideal. Appropriate, representations, first, should provide a way to formalize and reason about what it is obligatory, permitted or prohibited for an agent, and, second, should address issues such as *normative violation*, *contrary to duty structures,* i.e. the specification of a primary obligation, along with the specification of a secondary obligation that obtains if the primary one is violated [44, 200], and *institutionalized power* [136]. To this end the use of Deontic Logic was proposed [135, 200, 136, 36].

Deontic Logic is a form of modal logic used to reason about norms and was first presented by Ernst Mally (1926), while later, in 1951, von Wright established deontic notions as are known today in Standard Deontic Logic (SDL) [249, 250, 176, 177]. Deontic Logic introduces one primitive operator to represent expressions such as “*it is obligatory*”, and two dual operators that derive from the primitive one to represent expressions such as “*it is permitted*” and “*it is forbidden*”.

According to Standard Deontic Logic the following inter-definability relations hold among operators for Obligation (O), Permission (P) and Prohibition (F):

It is obligatory that a: Oa

It is permitted that a: $Pa \equiv \neg O \neg a$

It is prohibited that a: $Fa \equiv \neg Pa \equiv O \neg a$

2.3.2 e-Contracts

During the last decade a rapid growth on research activity related to normative systems, electronic institutions or virtual communities has occurred. Such domains were considered as open normative electronic environments that provide a framework for brokering, negotiation, agreement establishment and subsequently agreement monitoring. Agreements, the so called *contracts*, are reached in order to regulate the activity of autonomous agents. Their scope depends on the exact application area, i.e [35]:

- *Purchase Contracts* are used in e-commerce applications within *e-marketplaces*.
- *Cooperation Contracts* are used to model and manage enterprise business processes, also known as workflows.
- *Social Contracts* are considered as a set of norms, rules, commitments or conventions that coordinate and manage a virtual society.

Generally, the term *contract* is used to refer both to a legally binding agreement between two or more parties and to the *document* that records such an agreement. Generally, a contract creates mutual legal relations between the parties involved and determines what actions are obligatory/permissible/forbidden to be performed by the parties. In Information Technology, contracts are considered as electronic documents that describe, regulate and enable automation of business processes. The term *e-contract* is used to refer to the electronic document that is determined by reference to an individual computer-generation transaction. Researchers agree that contracts are instances of documents classes called *contract templates*. Those templates are *generic* documents containing obligatory and optional contract clauses, that each one addresses a specific point of interest of the business interaction. The term *e-contracting* is used to encompass all activities concerned with this transaction and e-contracts [9, 54, 150].

The whole contractual life cycle comprises of two different phases, namely *contract formation* and *contract performance* (Figure 2.1). During the former, parties communicate with each other, exchange needed information and negotiate the terms and conditions of their agreement. During the latter, once an agreement has been established; contract execution takes place, during which violations of contract clauses may occur, thus raising the need for reparatory mechanisms to be deployed, if any such are stipulated in the agreement.

Consequently there are two classes of service tools that are involved with contractual activity:

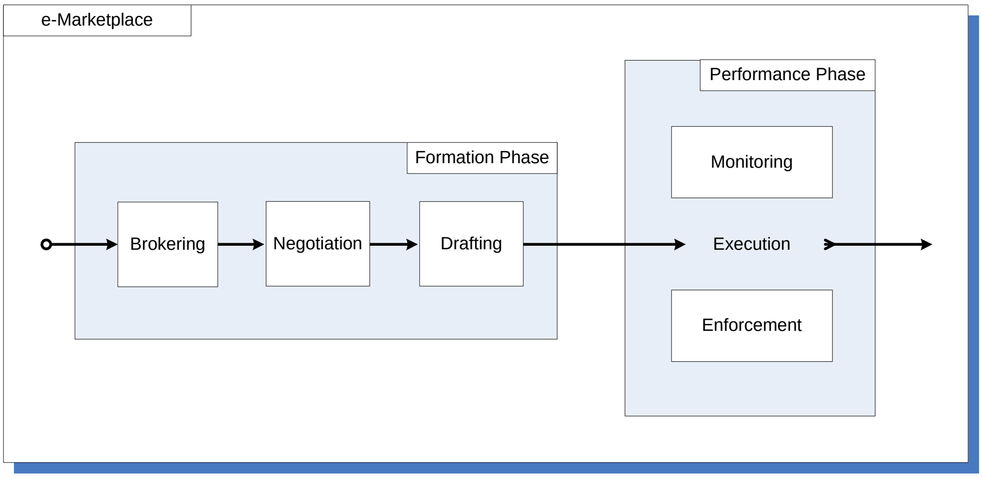


Figure 2.1 E-contract life cycle

- Contract Formation Service Tools:
  - Brokering tool, where a partner matching service is enabled.
  - Negotiation tool, where negotiation with the matched partner and the document lodging take place.
  - Drafting tool, where contract establishment occurs.
- Contract Performance Service Tools:
  - Execution tool, where actions planning, behaviour advisory and execution occur.
  - Monitoring tool, where monitoring of parties' compliance with the agreement occurs and factual or potential states of the transaction are examined for counter intuitive and conflicting situations.
  - Enforcement tool, where, in case of contract violations, suggestions for recovery actions or advices for the possible consequences are provided, while, in cases of conflicts a resolution strategy is chosen and applied.

It is obvious that the whole contractual procedure is too complicated and consists of different, iterative and interrelated multi-party interactions. Because of this, the separation of the drafting procedure is imperative. In [57, 54] two sequential contract drafting procedures were proposed. Here, we redefine the role, the attributes and the procedures that were formerly ascribed in the two-level proposed contract drafting workflow:

- Contract drafting at the *macro-level*. During this phase the drafting tool aims to formulate the overall structure of the document by choosing all or some of the predefined clauses of the contact template and optionally by adding new clauses as enhancements. The role of an e-contract is to define a legal interaction framework. Therefore, the formulated electronic contractual document should be:
    - *Well-structured*: A predefined form should be met.
    - *Expressive*: All contractual parties, goods, services and interactions should be recorded.
    - *Complete*: All possible circumstances should be considered.
    - *Consistent*: All contractual clauses should be coherent.
    - *Compendious*: No redundant clauses should be met.
- Contract drafting at the *micro-level*. During this phase the drafting tool focuses in drafting a well-formed and performable document. Due to this, all previously selected contractual clauses should be represented and combined in a formal way. In [57, 54] a non exhausted list of clauses features were presented. Specifically, contract provisions should be:
    - *Descriptive*: To define particular contract terms and items.
    - *Prescriptive*: To determine contractual parties' behaviours.
    - *Procedural*: To specify procedures that need to be followed when specific states are established or when a specific state needs to be established.
    - *Algebraic*: To calculate contract parameters and avoid redundancy.
    - *Effective*: To specify conditions under which other provisions apply.

2.4 Summary

In this chapter, we discussed intelligent agents, multi-agent interactions, and the systems where these interactions take place. Systems were considered as open computational environments that are governed by norms. Especially, we referred: to the role of autonomy in the context of agency; the special characteristics of open systems; two approaches (CWA and OWA) that deal with information gaps in these settings and the reasons that motivate us to differentiate. Finally, we provided, in brief, the essential characteristics of normative systems and introduced what e-contracts and e-contracting are.

3 Common-sense Reasoning

3.1 Introduction

Elio in [75] concludes that:

> "…it is through careful, reasoned *rationality* that we discover the 'truth' and through unreflective *common sense* that we make judgments that are 'sound enough' for dealing with matters that arise in the everyday world."

McCarthy in a series of papers, such as [171, 170, 174], works towards the understanding the common-sense capability. He divides common-sense capability in *common-sense knowledge* (CSK) and *common-sense reasoning* (CSR). CSK concerns the information that is available about a scenario and its appropriate representation in order to common-sense reason with and about it. Key issues about CSK are [174, 183]:

- the ability to identify fundamentals objects/entities that are involved in the scenario and the ability to determine their properties.
- the ability to situate the scenario on a temporal basis and the ability to represent the properties of the world that change over time.
- the ability to represent actions or events, their preconditions and their effects (direct or indirect effects, context-sensitive effects, nondeterministic effects, delayed effects, triggered effects) on the world.
- the ability to capture the *Common-sense Law of Inertia*, i.e. the axiom that the world stays the same unless it is affected by some action or event.
- the ability to represent the space of the scenario.

- the ability to represent the mental states, such as knowledge, beliefs, goals, likes and dislikes, intentions and abilities, of partners or competitors (human or software agents) in the scenario.

McCarthy notes that the ability to use CSK depends on the ability to use CSR. In most realistic scenaria, rarely we have complete knowledge and static world. Key issues about CSR are [174, 183]:

- the ability to fill in gaps in available information.
- the ability to take decisions with incomplete and uncertain knowledge.
- the ability to revise believes and decisions when complete knowledge becomes available.

In this thesis, supplementary to all of the above issues, we espouse the following description of CSR as presented by Mueller in [183]:

> “Commonsense reasoning is a process that involves taking information about certain aspects of a scenario in the world and making inferences about other aspects of the scenario based on our commonsense knowledge, or knowledge of how the world works. Commonsense reasoning is essential to intelligent behavior and though. It allows us to fill in the blanks, to reconstruct missing portions of a scenario, to figure out what happened, and to predict what might happen next.”
>
> “A method for automated commonsense reasoning must support several types of commonsense reasoning. The first is temporal projection or *prediction*, in which we start with an initial state and some events and then reason about the state that results from the events. … The second type for reasoning is *abduction*, in which we start with an initial state and a final state and then reason about the events that lead from the initial state to the final state. … The third type of reasoning is *postdiction*, in which we start with some events that lead to a state and then reason about the state prior to the events.”

In the next sections we discuss logic languages and reasoning patterns that we exercise in this thesis in order to address the issue of CSR.

3.2 Reasoning about Time, Actions and Effects

Many different logic languages are met in the AI literature for temporal projection and reasoning in dynamic domains, such as, the McCarthy's Situation Calculus [172], the Kowalski's and Sergot's Event Calculus [149], the Thielscher's Fluent Calculus [240], and the action languages C/C+ [103, 101, 151], amongst others. In the next section we discuss logic representation in Event Calculus.

3.2.1 Event Calculus

The Event Calculus (EC), introduced by Kowalski & Sergot in 1986 [149], is a formalism for representing and reasoning about actions and their outcomes. The basic elements of this logic language are *time points*, *fluents* and *actions* (or else *events*).

A time point `time` corespond to each state of the world, while `time < time + 1` indicates that time point `time` is before time point `time + 1`. A fluent `fluent` is a fact or predicate whose value can be altered over time. Actions, denoted as `action`, are considered all possible events that occur at some point time.

Table 3.1 The Event Calculus Predicates

PREDICATE	INTERPRETATION
Initiates(action, fluent, time) / Terminates(action, fluent, time) Initiates(event, fluent, time) / Terminates(event, fluent, time)	fluent starts/stops to hold after action occurs at time. fluent starts/stops to hold after event occurs at time.
HoldsAt(fluent, time)	fluent holds at time.
Happens(action, time)	action occurs (instantaneously) at time.
Clipped(time1, fluent, time2) / Declipped(time1, fluent, time2)	fluent is terminated/activated between time1 and time2.

In this thesis we adopt the EC formalism as explained in [180, 228]. This work is based on examples taken from literature and presents how EC applies in various domains in order to represent actions with indirect or non-deterministic effects, compound or concurrent actions and continuous change. Specifically, we adopt and adapt the so called simple EC, which introduces some basic predicates such as (Table 3.1):

- Initiates(action, fluent, time) represents that fluent fluent starts to hold after action action occurs at time point time,
- Terminates(action, fluent, time) represents that fluent fluent stops to hold after action action occurs at time point time,
- HoldsAt(fluent, time) represents that fluent fluent holds at time point time,
- Happens(action, time) represents that action action occurs at time point time,
- Clipped(time1, fluent, time2) represents that fluent fluent is terminated between time points time1 and time2,
- Declipped(time1, fluent, time2) represents that fluent fluent is activated between time points time1 and time2.

In [180, 228] six domain independent axioms (Clipped, Declipped, HoldsAt and ¬HoldsAt) were defined as shown in Table 3.2:

Table 3.2 The Event Calculus domain-independent axioms

Axiom	INTERPRETATION
Axiom 1	$\text{Clipped}(\text{time1, fluent, time2}) \equiv \exists \text{action,time } [\ (\text{Happens}(\text{action, time}) \wedge \text{Terminates}(\text{action, fluent, time}) \wedge \text{time1} \leq \text{time} < \text{time2}\ ]$
Axiom 2	$\text{Declipped}(\text{time1, fluent, time2}) \equiv \exists \text{action,time } [\ (\text{Happens}(\text{action, time}) \wedge \text{Initiates}(\text{action, fluent, time}) \wedge \text{time1} \leq \text{time} < \text{time2}\ ]$
Axiom 3	$\text{HoldsAt}(\text{fluent, time2}) \leftarrow (\text{Happens}(\text{action, time1}) \wedge \text{Initiates}(\text{action, fluent, time1}) \wedge \text{time1} < \text{time2} \wedge \neg\text{Clipped}(\text{time1, fluent, time2})$
Axiom 4	$\neg\text{HoldsAt}(\text{fluent, time2}) \leftarrow (\text{Happens}(\text{action, time1}) \wedge \text{Terminates}(\text{action, fluent, time1}) \wedge \text{time1} < \text{time2} \wedge \neg\text{Declipped}(\text{time1, fluent, time2})$
Axiom 5	$\text{HoldsAt}(\text{fluent, time2}) \leftarrow (\text{HoldsAt}(\text{fluent, time1}) \wedge \text{time1} < \text{time2} \wedge \neg\text{Clipped}(\text{time1, fluent, time2}))$
Axiom 6	$\neg\text{HoldsAt}(\text{fluent, time2}) \leftarrow (\text{HoldsAt}(\text{fluent, time1}) \wedge \text{time1} < \text{time2} \wedge \neg\text{Declipped}(\text{time1, fluent, time2}))$

The first definition for HoldsAt above reflects the establishment of a fluent as a result of an action, while the second one reflects the Common-sense Law of Inertia. The predicates Happens, Initiates and Terminates are domain dependent therefore they are defined circumstantial.

3.3 Non-monotonic Reasoning

Entailment in most formal logics, such as the Propositional Logic (PL) and the First-order Logic (FOL), or reasoning patterns, such as *Deductive Reasoning*, is *monotonic*. *Monotonicity* says that, if $KB \models a$ then $KB \land b \models a$, i.e. new knowledge that is added in the knowledge base KB can only produce new entailments.

However, in CSR, sometimes it is not necessary that new knowledge validates all previous entailments. This property is called *non-monotonicity* and the corresponding reasoning pattern is called *Non-monotonic Reasoning* (NMR). Non-monotonicity says that, new knowledge that is added in the knowledge base may invalidate previous entailments.

Strongly related with the pattern of NMR, are the two other reasoning patterns called *Default Reasoning* (or reasoning by default) and *Defeasible Reasoning*. Reasoning by default is the ability to jump to a non-monotonic conclusion, when there is no absolute knowledge about the world. Sometimes it is necessary to make certain assumptions in order to proceed with inference. For instance, reasoning by default enables us to capture rules such as “A bird typically flies, in the absence of the knowledge of the contrary”. Such inference on the basis of certain assumptions, also afford us the ability: (i) to relate certain conclusions to certain assumptions, and (ii) to retract certain conclusions when required. Thus, *Defeasibility* is the ability to invalidate previous drawn conclusions when it is desirable.

In this thesis, we are engaged in such CSR, i.e. assumptions are made in order to facilitate reasoning with incomplete knowledge, certain defeasible conclusions are drawn on the basis of these assumptions, and finally, when new knowledge becomes available, that renders previous made assumptions or drawn conclusions false, certain assumptions and conclusions are being retracted. For example, consider that on the basis of currently available knowledge and in the absence of knowledge of the contrary the name of a cartoon bird is assumed to be ‘Tweety’. In this case, we may conclude that the bird is a canary and is able to fly. But, at a later time point we found out that the name is ‘Mubble’, so we are no longer allowed to rely on this assumption and consequently, we are required to retract any conclusion on its basis, and conclude that the bird is a penguin and does not fly.

Many approaches have been proposed during the last thirty years to non-monotonic reasoning such as Reiter's Default Logic [208], McCarthy's Circumscription [173], Moore's Autoepistemic Logic [182], Gelfond's and Lifschitz's Logic Programs (with stable model or answer set semantics) [83, 84], and Nute's Defeasible Logic [187]. In the next section we discuss reasoning with Default Logic.

3.3.1 Default Logic

Default Logic (DfL) [208], introduced by Reiter in 1980, is arguably the most notable formulation for default reasoning (cf. [14, 154]). A *default rule* (henceforth *default*) has the form[3]:

$$\frac{P : J1,J2,\ldots Jn}{C}$$

where P is the *prerequisite*, $J=\{J1,J2,\ldots Jn\}$ is a set of *justifications* and C is the derived *consequent*. The sets P, J and C contain PL or closed FOL formulae. The semantics of a default rule is: If P holds and the assumption J is consistent with our current knowledge, then C may be inferred. For instance, for the following default rule:

$$\frac{\mathrm{Bird} : \neg \mathrm{Flies}, \mathrm{Swims}}{\mathrm{IsPenguin}}$$

its interpretation is as follows: for Bird if it is consistent to assume that it does swims and does not flies then we may conclude that Bird is a penguin. In the case of a first-order language, i.e. logic formulae that contain free variables, the rule has the following form:

$$\frac{\mathrm{Specie}(x,\mathrm{Bird}) : \neg \mathrm{Flies}(x), \mathrm{Swims}(x)}{\mathrm{Is}(x,\mathrm{Pengiun})}$$

Let us call these rules *default schemata*. Default schemata represent a set of possible defaults that emerge when computing the possible ground substitutions that assign values to the free variables that appear in the default schema. In other words, it is considered that free variables of default schemata are universally quantified over the

---

[3] Prerequisite-free default rules are defaults of the form $\mathrm{true} : J1,J2,\ldots Jn / C$. Justification-free default rules are defaults of the form $P : \mathrm{true} / C$.

whole schema. For instance, for the above default schema and given the facts that $\text{Specie(Tweety,Bird)}$ and $\text{Specie(Mumble,Bird)}$ the corresponding defaults are:

$$\frac{\begin{array}{c}\text{Specie(Tweety,Bird)}\\ :\\ \neg\text{Flies(Tweety), Swims(Tweety)}\end{array}}{\text{Is(Tweety,Pengiun)}}$$

and

$$\frac{\begin{array}{c}\text{Specie(Mumble,Bird)}\\ :\\ \neg\text{Flies(Mumble), Swims(Mumble)}\end{array}}{\text{Is(Mumble,Pengiun)}}$$

A Default Theory (DfT) is a pair of the form (W, D), where W is a set of PL or FOL formulae that represent currently available knowledge, and D is a set of defaults. Rules may be used to compute *extensions* E of the default theory. A default is applicable to a deductively closed set of formulae $W \subseteq E$ if and only if $P \in E$ and $\neg J_1, \ldots, \neg Jn \notin E$. In this case, the set E is called *extension* of the default theory. Extensions are the most complicated concept of Reiter's default theory because it is hard to determine an accurate belief set for which justifications should be consistent. In the Reiter's initial paper for DfL [208] three important properties of extensions were referred. In particular, an extension E of a default theory (W, D):

- should contain W,
- should be deductively closed and
- for a default rule of the form $P\colon J1, J2, \ldots Jn / C$, if $P \in E$ and $\neg J1, \ldots, \neg Jn \notin E$ then $C \in E$.

3.3.1.1 Skeptical and Credulous Reasoning

For an agent that reasons on the basis of a DfT, extensions represent possible world views. Whenever multiple extensions are computed, for a DfT, these represent multiple possible world views. Depending on its chosen action an agent is committed to a particular extension.

There are two classical approaches to perform such inference on the basis of DfL. In the first one, the *skeptical reasoning*, a formula is entailed by a theory, if it is entailed by all its extensions. This is a strict approach and requires the computation of all possible extensions and the subsequent check to determine if this formula belongs in all of the extensions. In the second approach, the *credulous reasoning*, a formula is entailed by a theory, if it is entailed by at least one extension. In this case, there is no need to compute all possible extensions.

For example, consider the following computed extensions as shown in Figure 3.1:

- Extension #1: { C1, C2, C3 }
- Extension #2: { C1, C2 }
- Extension #3: { C1, C3 }
- Extension #4: { C1 }

According to the skeptical reasoning approach an agent may only infer that C1 holds, because it appears in all possible computed extensions. On the contrary, according to the credulous reasoning approach, an agent may infer that: C1 holds when it chooses extensions 1, 2, 3 and 4; C2 holds when it chooses extensions 1 and 2; and finally, that, C3 holds when it chooses extensions 1 and 3.

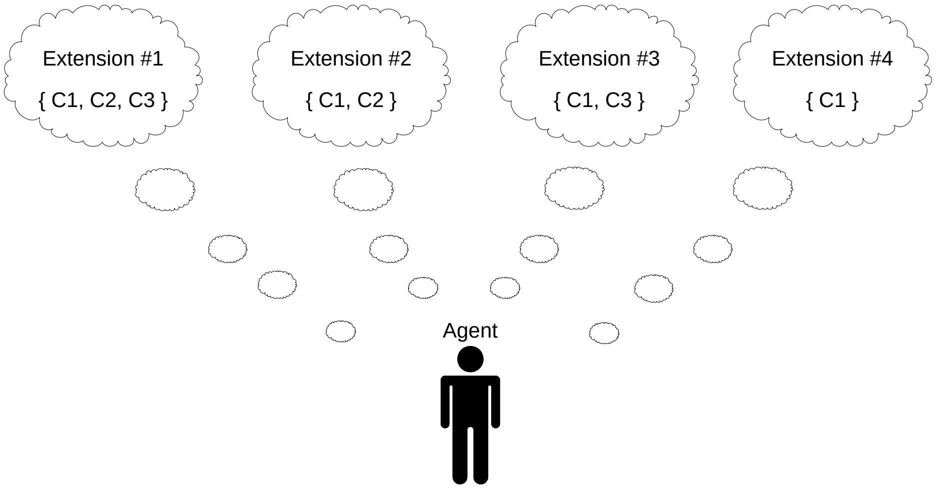


Figure 3.1 Skeptical and credulous reasoning

3.3.1.2 Operational Computation of Extensions

In this thesis, we adopt an operational process-oriented technique for the computation of extensions as presented by Antoniou in [12, 14]. Antoniou proposed an operational definition of extensions and an incremental technique for their computation, maintaining consistent sets of formulae whose conditions part (prerequisites and justifications) is interpreted conjunctively and the conclusions part (consequent) is interpreted disjunctively, as in sequent calculus [132]. Thus, an agent that derives conclusions on the basis of assumptions, by applying defaults, constructs the extension of its grounded DfT incrementally. Let Π represent a default reasoning process by recording the order in which defaults from D apply. At each step i of the reasoning process $\Pi(i)$, i.e. after the application of each default $P{:}J1,\ J2,\ldots,Jn/C$, the extension computed is a set of ground sentences $In(i)=In(i-1) \cup \{C\}$, and the set of assumptions employed, which should not turn out to be true, is $Out(i) = Out(i-1) \cup \{\neg J1\ldots, \neg Jn\}$. As a result, $\Pi(i)= \Pi(i-1) \cup \{Di \mid Di \text{ is the default rule which applied at step } i\}$. Initially $In(0)=W$, $Out(0)=\varnothing$ and $\Pi(0)=\varnothing$ for $i=0$. The default reasoning process $\Pi(i)$ is *successful* iff $In(i) \cap Out(i)= \varnothing$, otherwise it is *failed*. Moreover, the process $\Pi(i)$ is *closed* iff every default rule that belongs in the D set and is applicable in $In(i)$ already occurs in $\Pi(i)$. According to [14] a set of formulae E is a DfT extension if there is a closed and successful process $\Pi(i)$ of the DfT such that $E=In(i)$.

Example 1

In order to illustrate this interpretation consider the following DfT (W, D):

W={ P1, P2, P3 }

D={ D1 ≡ P1:J1/C1, D2 ≡ P2:J2/C2 }

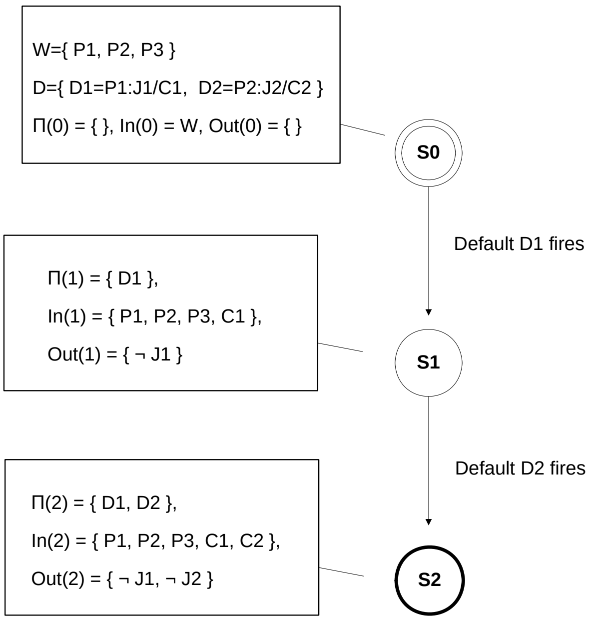


Figure 3.2 Operational computation of extensions - Example 1

For this theory the following sets are computed $In(2)=\{P1,\ P2,\ P3,\ C1,\ C2\}$, $Out(2)=\{\neg J1,\ \neg J2\}$, where $\Pi(2)=\{D1,\ D2\}$[4]. An abstract description of the computation of extensions as a state

---

[4] We may also consider the process $\Pi(2)=\{D2,\ D1\}$.

diagram is shown in Figure 3.2, where a double line node denotes the initial state where $\Pi(0)=\{\}$; and a bold line node denotes that process $\Pi(i)$ is a closed and successful process, i.e. $\Pi(i)$ is an extension, as defined by Antoniou.

Example 2

Consider the following DfT (W, D):

W={ P1, P2, P3 }

D={ D1 ≡ P1:J1/C1, D2 ≡ P2: ¬C1/C2 }

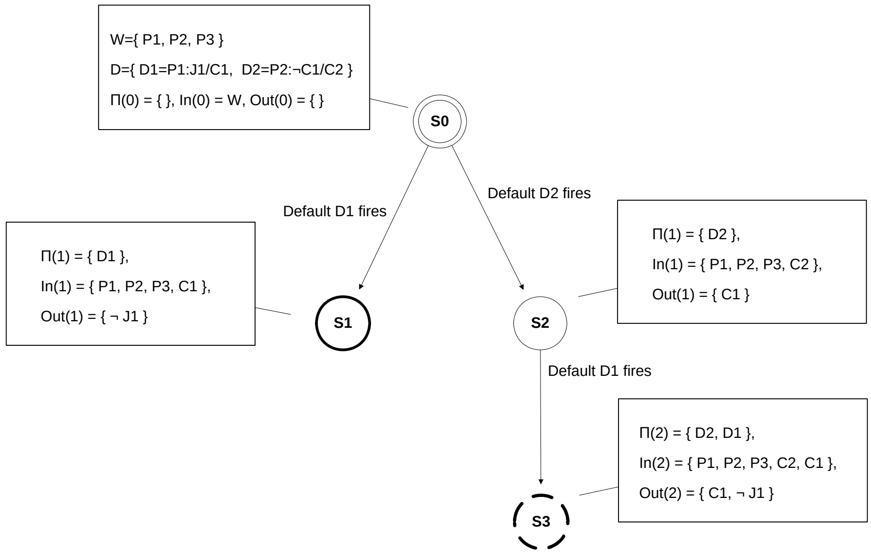


Figure 3.3 Operational computation of extensions - Example 2

For this theory the following sets are computed: $In(1)=\{P1, P2, P3, C1\}$, $Out(\Pi)=\{\neg J1\}$ for $\Pi(1)=\{D1\}$; and $In(2)=\{P1, P2, P3, C2, C1\}$, $Out(2)=\{C1, \neg J1\}$ for $\Pi(2)=\{D2, D1\}$. An abstract description of the computation of extensions as a state diagram as shown in Figure 3.3, where a double line node denotes the initial state where $\Pi(0)=\{\}$; a bold line node denotes that process $\Pi(i)$ is a closed and successful process, i.e. $\Pi(i)$ is an extension; and, finally, a discontinuous bold line node denotes that process $\Pi(i)$ has failed, as defined by Antoniou. At state $S3$ the process $\Pi(2)$ is a failure due to the fact that $In(2) \cap Out(2) \neq \varnothing$. Thus, there is only one extension for the DfT.

Example 3

Now, consider the following DfT (W, D):

W={ P1, P2, P3 }

$D=\{ D1 \equiv P1{:}J1/C1,\ D2 \equiv P2{:}J2/C2 \}$

where $C1$ and $C2$ are formulae that render the knowledge base inconsistent when both of them hold simultaneously. In this case there are various computations (possible extensions) for In and Out sets: $In(1)=\{P1, P2, P3, C1\}$, $Out(1)=\{\neg J1\}$ for $\Pi(1)=\{D1\}$; $In(1)=\{P1, P2, P3, C2\}$, $Out(1)=\{\neg J2\}$ for $\Pi(1)=\{D2\}$; and $In(2)=\{P1, P2, P3, C1, C2\}$, $Out(2)=\{\neg J1, \neg J2\}$, for $\Pi(2)=\{D1, D2\}$[5]. According to Antoniou, the processes $\Pi(1)=\{D1\}$ and $\Pi(1)=\{D2\}$ are not extensions of the DfT, due to the fact that these processes are not closed processes. On the contrary, the processes $\Pi(2)=\{D1, D2\}$ and $\Pi(2)=\{D2, D1\}$, are considered to be extensions (successful and closed processes) of the DfT. But in this case, an inconsistency arises in an agent's world.

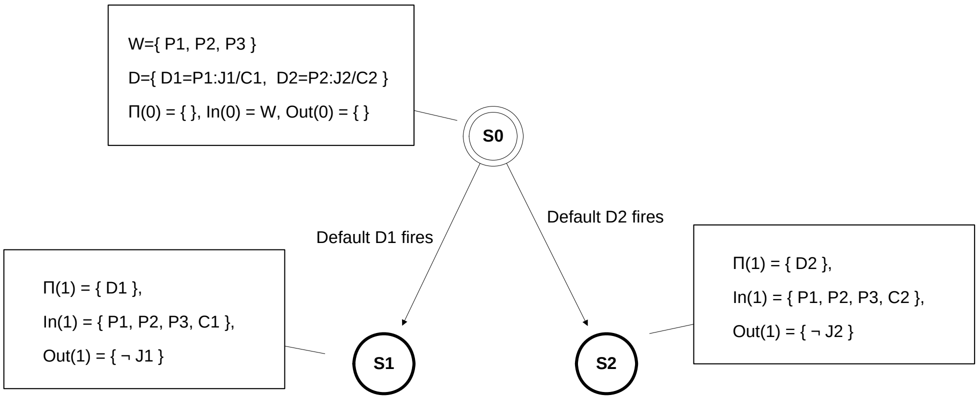


Figure 3.4 Operational computation of extensions - Example 3

In this thesis, in order to facilitate CSR in an open world where the environment and whatever is associated with it are matters of continuous change, we should not require or expect from the agent to either accept worlds that belong to the distance future or reject them, in advance. It seems rational and realistic to us, to require from an agent to reason in a process-oriented step-wise manner on the basis of its current knowledge. Such an agent would compute all possible worlds, examine the state that these worlds reveal (consistent/inconsistent, rational/irrational, eligible/non-eligible, desirable/non-desirable etc) and finally decide to either to follow or not a specific course of action. Specifically, with respect to the above example 3, our agent may accept the processes $\Pi(1)=\{D1\}$ and $\Pi(1)=\{D2\}$ as theory extensions, on the criteria that: (i) the next step in inference, i.e. applying defaults $D2$ or $D1$ respectively, if followed, would

---

[5] We may also consider the process $\Pi(2)=\{D2, D1\}$.

turn itself into an irrational agent, and (ii) to ground itself in a protracted period of idleness is not the reason of its development (Figure 3.4). Thus, the agent may place itself in either one of the two states (s_1 or s_2) and continue with inference from this state on a new basis, i.e. a new current world that probably contains different factual and prescriptive knowledge. We further discuss this issue and motivate our choice on extensions when we discuss the overall reasoning problem in chapter 5.

3.3.2 Variations of Default Logic

Since the publication of the Reiter's initial paper for Default Reasoning, many variations of Default Logic, such as the Normal Default Theory (NDfT) [208], the Semi-normal Default Theory (SnDfT) [77], the Justified-DfL (JDfL) [163], the Constrained-DfL (CDfL) [225], the Rational-DfL (RDfL) [178], the Cumulative-DfL (CuDfL) [28], the Pre-constrained-DfL (PcDfL) [226, 13], the Stratified-DfL (SDfL) [45, 46] and the Prioritized-DfL (PDfL) [29, 30], amongst others, have been proposed for various reasons and applications. In this subsection we discuss some of them in turn.

3.3.2.1 Normal – Semi-normal Default Theories

A Normal Default Theory (NDfT) [208] is a DfT where all defaults are normal. A default rule is called *normal* if its justification coincides with its consequent, i.e. a normal default has the form[6]:

$$\frac{P : C}{C}$$

The semantics of this default rule is: If P holds and is consistent with our current knowledge to assume C, then C may be inferred. Note that, every closed NDfT always has an extension.

A Semi-normal Default Theory (SnDfT) [77] is a DfT where all defaults are semi-normal. A default rule is called *semi-normal* if all its justifications imply its conclusion, i.e. a semi-normal default has the form:

$$\frac{P : J \wedge C}{C}$$

---

[6] Supernormal defaults are prerequisite-free normal default rules of the form $\mathrm{true}: C / C$.

Note that, every SnDfT does not always have an extension.

3.3.2.2 Constrained Default Theories

A Constrained Default Theory (CDfT) [225] is a DfT that avoids running into inconsistencies and ensures the joint consistency of all justifications involved in reasoning. A default is applied only if its justifications and consequences are consistent with the background theory, i.e., $In(i) \cup \neg Out(i)$. This is tantamount to saying that the possible world models inferred by the agent contain, besides previous knowledge, both the consequents and the assumptions of the applied defaults.

Example 4

Consider the following CDfT (W, D):

W={ P1, P2, P3 }

D={ D1 ≡ P1:J1/C1, D2 ≡ P2:J2/C2, D3 ≡ P3:J3/C3 }

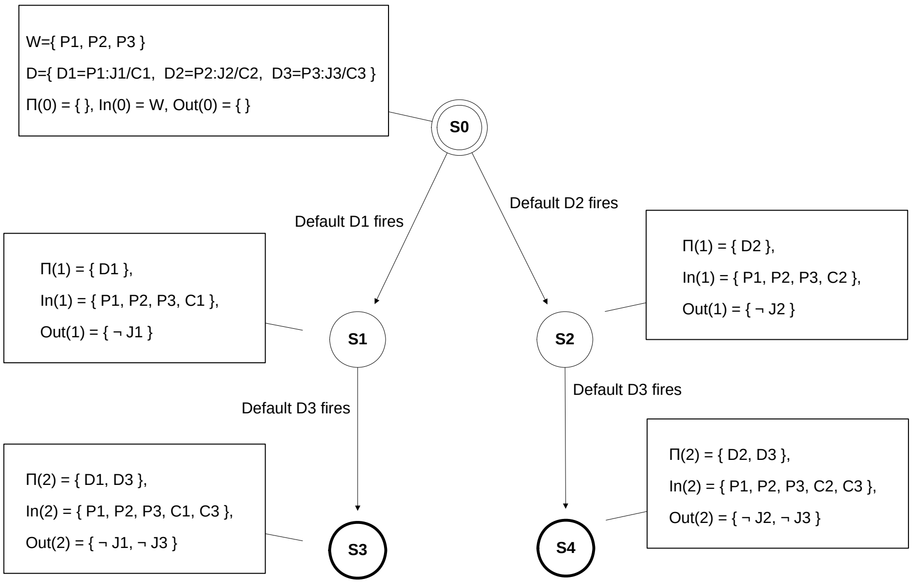


Figure 3.5 Operational computation of extensions - Example 4

where the pairs C1 - C2 and J1 - J2, are formulae that render the knowledge base inconsistent when both members of the pairs hold simultaneously. For this CDfT there are two accepted extensions: $In(2)=\{P1, P2, P3, C1, C3\}$, $Out(2)=\{\neg J1, \neg J3\}$ for $\Pi(2)=\{D1, D3\}$; and $In(2)=\{P1, P2, P3, C2, C3\}$, $Out(2)=\{\neg J2, \neg J3\}$ for $\Pi(2)=\{D2, D3\}$; due to the restrictions, i.e. one due to the joint consistency of assumptions, and one due to the consistency of the computed world (Figure 3.5).

3.3.2.3 Pre-constrained Default Theories

A Pre-constrained Default Theory (PcDfT) is a triple of the form (W, D, PC), where (W, D) is a CDfT and PC is set of formulae that are considered as the initial constraints of the theory [226]. For the first step of the process, i.e., for $i=1$, $In(0)=W$ and $Out(0) = PC$.

Example 5

Consider the following PcDfT (W, D):

W={ P1, P2, P3 }

D={ D1 ≡ P1:J1/C1, D2 ≡ P2:J2/C2, D3 ≡ P3:J3/C3 }

PC={ J3 }

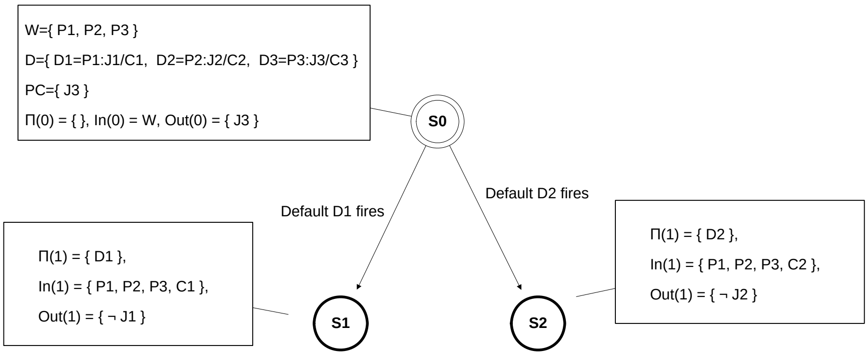


Figure 3.6 Operational computation of extensions - Example 5

where the pairs $C1$ - $C2$ and $J1$ - $J2$, are formulae that render the knowledge base inconsistent when both members of the pairs hold simultaneously and $J2$ is a formulae acting as a pre-constrain of the theory. For this PcDfT there are two accepted extensions: $In(1)=\{P1, P2, P3, C1\}$, $Out(1)=\{ \neg J1\}$ for $\Pi(1)=\{D1\}$; and $In(1)=\{P1, P2, P3, C2\}$, $Out(1)=\{ \neg J2\}$ for $\Pi(1)=\{D2\}$; due to the restrictions, i.e. one due to the joint consistency of assumptions, one due to the consistency of the computed world and, finally, one due to the pre-constrain (Figure 3.6).

3.3.2.4 Stratified Default Theories

A DfT is called Stratified Default Theory (SDfT) [45, 46] iff there exists a stratification function s that assigns a natural number to each default and composes strata from the initial D set. If the consequent of the rule D' (default rule in case of DfL) is used by another rule D'' then we apply D' before D'' i.e., $s(D') \leq s(D'')$. The formal characterization of this property for any defaults D' and D'' is as follows:

$$\text{if } prop(cons(D')) \cap prop(pre(D'') \cup just(D'')) \neq \emptyset \text{ then } s(D') \leq s(D'')$$

$$\text{if } prop(cons(D')) \cap prop(cons(D'')) \neq \emptyset \text{ then } s(D') = s(D'')$$

where $pre(D')$, $just(D')$ and $cons(D')$ denote the prerequisites, justifications and consequents part of D', respectively, and $prop(D')$ denote the set of all atoms occurring in D' [13]. According to this definition D' and D'' are mapped into different strata because $s(D') \leq s(D'')$ holds. In this way, an agent ensures that D' applies before D'' and no knowledge is lost.

Example 6

Consider the following SDfT (W, D):

$W=\{ P1, P2, P3 \}$

$D=\{ D1 \equiv P1{:}J1/C1,\ D2 \equiv P2\wedge C1{:}J2/C2,\ D3 \equiv P3\wedge C2{:}J3/C3 \}$

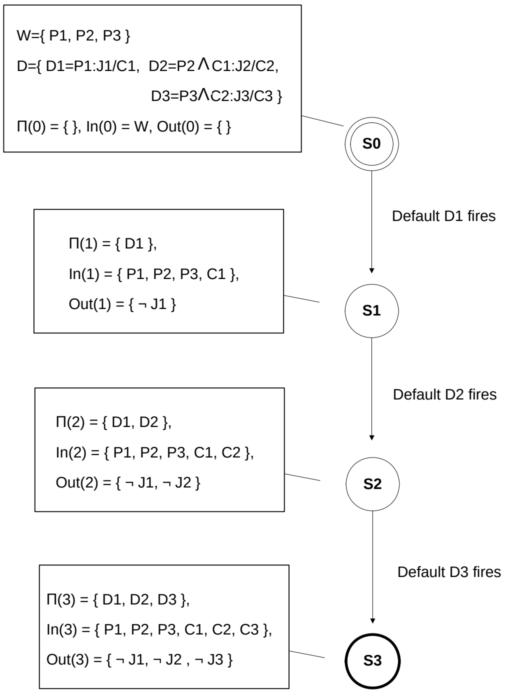


Figure 3.7 Operational computation of extensions - Example 6

where the pairs $C1$ - $C2$ and $J1$ - $J2$, are formulae that render the knowledge base inconsistent when both members of the pairs hold simultaneously. For this SDfT there is only one accepted order to apply the defaults, i.e. there is only one possible extension: $In(3)=\{P1, P2, P3, C1, C2, C3\}$, $Out(2)=\{\neg J1, \neg J2, \neg J3\}$ for $\Pi(2)=\{D1, D2, D3\}$; due to the fact that defaults $D1$, $D2$ and $D3$ are placed in different strata according to the stratification function s (Figure 3.7).

3.3.2.5 Prioritized Default Theories

Brewka in [29, 30] defines a Prioritized Default Theory (PDfT) as a triple (W, D, name), where name is a function that assigns names to default rules *D*. The extension of a PDfT is derived in the same way as in a DfT. Priorities over defaults can either define preference on extensions that are, eventually, preferred transaction plans when dealing with the query “Which norm should I apply?”, or define preference on normative relations that already hold, that is an answer to the query “Which normative relation should I concede?” based on the priorities of defaults that entailed these normative relations.

Example 7

To illustrate this interpretation consider the following PDfT (W, D):

W={ P1, P2, P3 }

D={D1 ≡ P1:J1/C1, D2 ≡ P2:J2/C2, D3 ≡ P2:¬C1/C3}

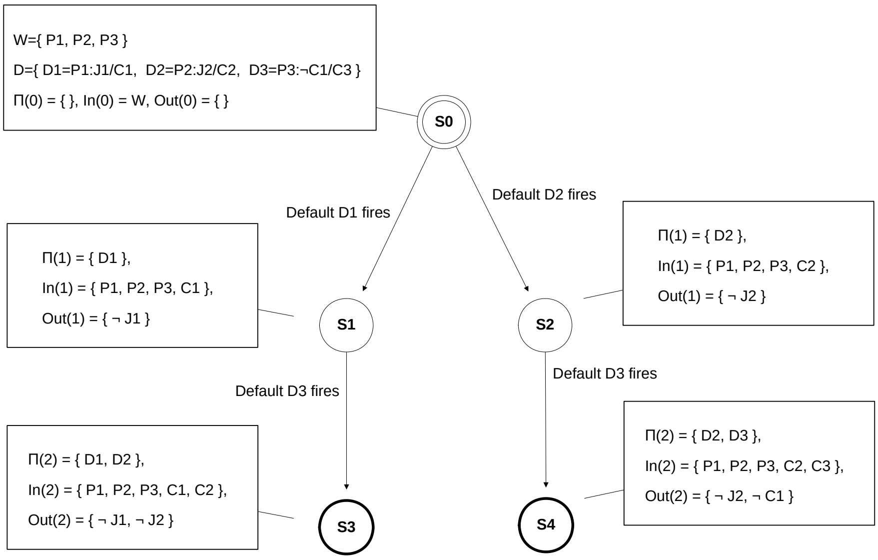


Figure 3.8 Operational computation of extensions - Example 7

where the priority relation among default is D1<D2<D3 [7]. For this PDfT there are two possible sequences for the application of defaults, i.e. there are two possible extensions: In(2)={P1, P2, P3, C1, C2}, Out(2)={ ¬J1, ¬J2} for Π(2)={D1, D2}; and In(2)={P1, P2, P3, C2, C3}, Out(2)={ ¬J2, ¬C1} for Π(2)={D2, D3} (Figure 3.8). Note that, the preference on defaults determines the

---

[7] < notation describes a priority relation. a< b means that the right part (b) takes priority over the left part (b) of this relation, i.e. b is preferred to a.

order that these defaults apply in the theory. Moreover, the priority relation that holds among the defaults determines a preference over the two possible extensions, i.e. extension computed for $\Pi(2)=\{D2, D3\}$ is the preferred extension over the extension computed for $\Pi(2)=\{D1, D2\}$.

3.4 Summary

In this chapter, we discussed logic languages and reasoning patterns that we adopt and adapt in this thesis in order to address the issue of CSR in open norm-governed multi-agent environments. First, we referred to the meaning of CSR in AI and Computer Science, and then, we discussed specific issues that are related to the representations and reasoning techniques which will be presented in the following chapters. Specifically, we: showed the EC language that we use in order to reason with time, actions and their effects; explained Reiter's DfL and Antoniou's work towards the computation of extensions; and reminded some of the major variations of DfL. Moreover, during this chapter we discussed research questions that arise in such settings and motivate our viewpoint.

Part III
An e-Contract Framework

4 Application Area Analysis

4.1 Introduction

In this chapter we analyze the application area of this thesis, i.e. electronic agreements. We broadly refer to three application sub-areas, where electronic agreements play a central role: e-commerce, business process modelling and automation, and virtual communities. Specifically, first we discuss various example scenaria and present a set of requirements that a representation of electronic agreements should meet, in order to facilitate the development of tools for contract performance monitoring. We then review research efforts related to contract representation and contract performance monitoring that have emerged during the last decade and are based on logic and present the open issues that each approach deals with and the characteristic techniques that have been employed to this scope. Typically approaches involve a representation of deontic notions (such as obligation, permission, prohibition and power), their associated meta-level notions (such as violation, sanction, compliance and normative conflict), mental notions (such as beliefs, desires, intentions and trust) and domain-independent concepts such as time, actions and their effects. Furthermore, we comment which of the noted requirements are met.

The rest of the chapter is organized as follows: section 4.2 introduces several example scenaria of the application area of this thesis; section 4.3 introduces and discusses a list of requirements for representation languages and specifications for the development of computational tools; section 4.4 provides a critical review on previous logic-based approaches that deal with purchase, cooperation and social contracts; and finally, section 4.5 provides a summary.

4.2 e-Contract Example Scenaria

In this thesis, for the purposes of illustration, we consider various example scenaria of business transactions between different parties of interest (e.g. a Buyer, a Retailer, a Wholesaler, a Mediator, an Importer and a Carrier). Business transactions take place in electronic marketplaces that are populated by software agents. Each agent acts as a representative for one contracting party. The multi-party scenario outlines the manner in which a business interaction is being carrying out, taking into account normative positions that hold among contracting parties, actions that can be performed during the commercial transaction and their effects on the different states of this transaction.

Let the set `Agents={Agent1, Agent2, Agent3,.....}` denote distinct identifiers for the various agents, and the set `Roles={BA, RA, WA, CA, MA, IA, ...}` denote distinct roles that agents may assume in the e-market (where `BA`, `RA`, `WA`, `CA`, `MA`, `IA` denote *buyer*, *retailer*, *wholesaler*, *carrier*, *mediator* and *importer* respectively). Assume, now, that a buyer requests a product. The buyer, in order to order successfully the requested product, should provide to its representative agent (agent: `Agent1`, role: `BA`) all essential knowledge for its request. For instance, `Agent1` who acts as a `BA` is aware of the amount of the requested goods, their type, the time and the form of the delivery and the bounds of the acceptable price. Correspondingly, the buyer's agent communicates with the retailer's agent (agent: `Agent2`, role: `RA`) and settles down an agreement that fulfils its request. Furthermore, the agreement is enhanced with additional information such us address of delivery, the way of payment, guarantee of normal execution, the way of possible compensation in case of violations and the commission in case of mediating. Note that during this work we do not aim to examine the way that agents negotiate, but we focus on the contract drafting phase, its electronic representation and contract performance monitoring. The retailer's agent is, either, able to satisfy immediately its contract with the buyer's agent by selling goods from its stock or defines a new request from its point of view towards possible wholesaler partners (agent: Agent4, role: WA).

Taking into account the first option (case A) the business transaction embodies two contracting parties, i.e. the buyer (agent: Agent1, role: BA) and the retailer (agent: Agent2, role: RA). Apart from this agreement, another agreement should be established to complete the whole transaction. The new agreement is related to the transportation of the requested goods from one contractual party to another. Thus, the retailer should communicate with a carrier agent (agent: Agent3, role: CA) and establish another agreement with it for the timely and safe delivery of goods towards the buyer (Figure 4.1).

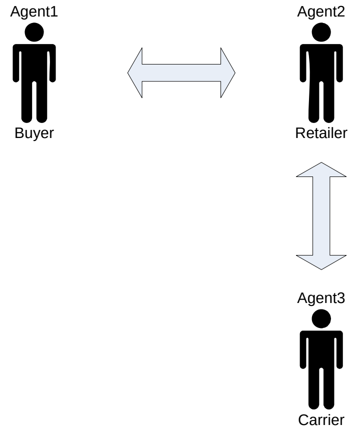


Figure 4.1 Case A: Contracts between Buyer – Retailer and Retailer – Carrier

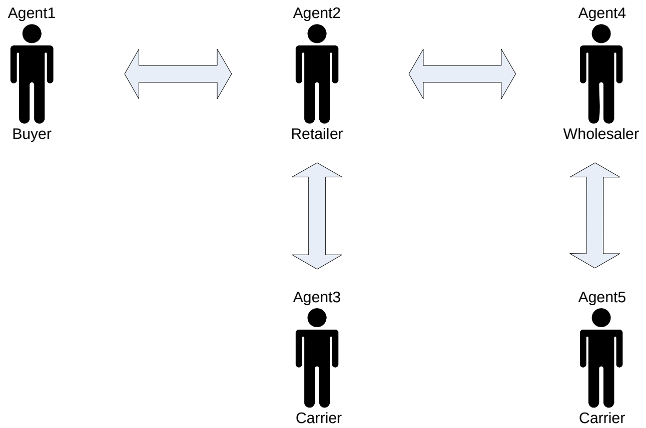


Figure 4.2 Case B: Contracts between Buyer – Retailer, Retailer – Carrier, Retailer - Wholesaler and Wholesaler – Carrier

Based on the second option Agent2 has a dual role. It can act either as a retailer (agent: Agent2, role: RA) (case B) – that is to establish a new agreement with the wholesaler, purchase requested goods in any form, transform them in the desirable package and resell them to the buyer, or act as a mediator (agent: Agent2, role: MA) (case C) – that is to mediate among the buyer and the wholesaler in order to come to a new agreement and request a commission. According to the first option the business transaction embodies five contractual parties, connected with contracts as shown in Figure 4.2. Here, two transportation contracts have been established: the first between the wholesaler and a carrier (agent: Agent5, role: CA) and the second, between, the retailer and another carrier (agent: Agent3, role: CA), in order to deliver goods from the wholesaler to the retailer and from the retailer to the buyer, respectively. According to the later option the business transaction embodies four contractual parties, connected with contracts as shown in Figure 4.3. Here, one transportation contract have been established between the wholesaler and a carrier (agent: Agent5, role: CA) in order to deliver goods from the wholesaler to the buyer directly.

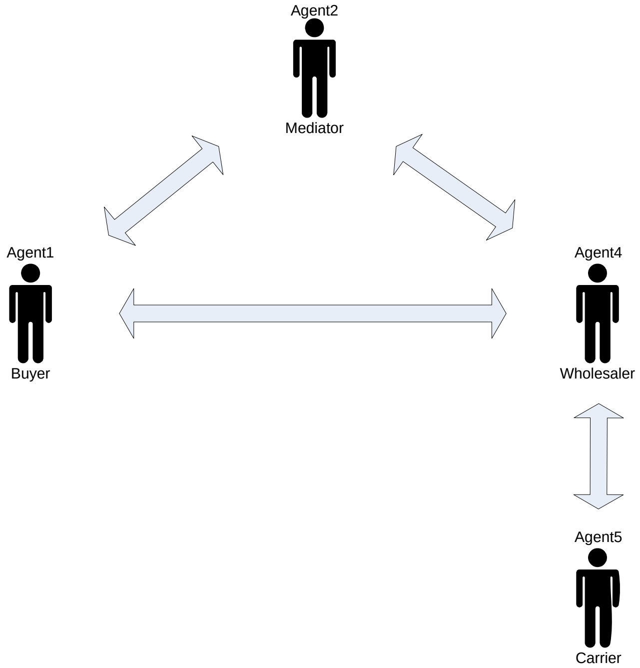


Figure 4.3 Case C: Contracts between Buyer – Mediator, Mediator – Wholesaler, Buyer - Wholesaler and Wholesaler – Carrier

Of course, we may extend this scenario with more contracting parties, e.g. an importer (agent: Agent6, role: IA) and another carrier (agent: Agent7, role: CA) as shown in Figure 4.4. On the basis on the above analysis for a realistic commercial agreement, it is clear that the whole process could be separated in sub-processes of two contracting parties. In other words, we may focus on a two-party contract that includes all the essential characteristics of a contractual agreement and is a representative example of complex agreements that take place in e-marketplaces. In the next section, we present in more detail a case study three-party scenario, which contains two independent two-party contracts.

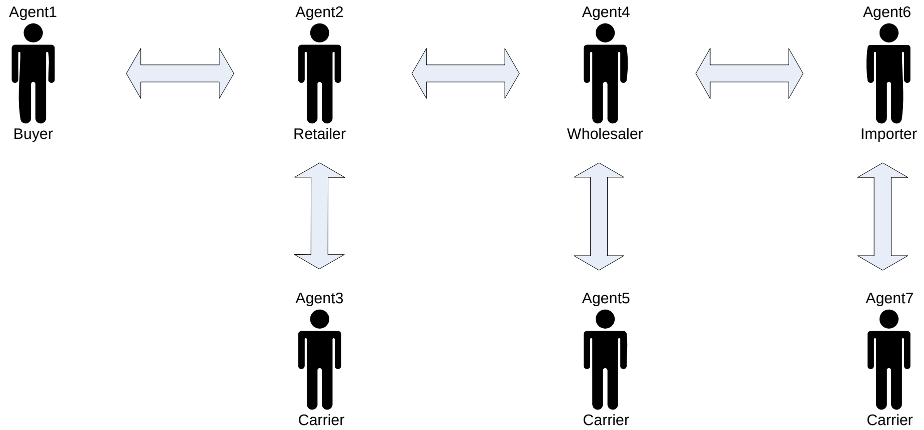


Figure 4.4 Case D: Contracts between Buyer – Mediator, Mediator – Wholesaler, Buyer – Wholesaler, Wholesaler – Carrier, Wholesaler – Importer and Importer - Carrier

4.2.1 A three-party business transaction

For the purposes of illustration consider an electronic marketplace, populated by software agents that establish and perform e-contracts on behalf of some real world parties (Figure 4.5).

Let the set Agents={Agent1, Agent2, Agent3,.....} denote distinct identifiers for the various agents, and the set Roles={BA, SA, CA, ...} denote distinct roles that agents may assume in the e-market (where BA, SA, CA, denote *buyer*, *seller* and *carrier* respectively). A buyer (agent: Agent1, role: BA) communicates with a seller (agent: Agent2, role: SA) and establishes an agreement with it for purchasing a certain product. Consequently, the seller communicates with a carrier agent (agent: Agent3, role: CA) and establishes another agreement with it for the timely and safe delivery of goods toward the buyer.

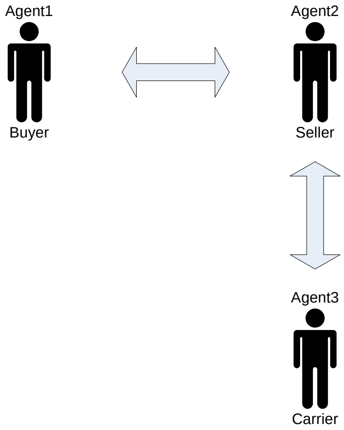


Figure 4.5 A three-party business transaction

The first agreement (between BA and SA) is to be conducted on the following terms[8]: SA should see to it that the goods be delivered to BA within 10 days from the date BA's order happens. BA, in turn, should see to it that payment be made either in cash on delivery or within 21 days from the date it receives the goods, via a deposit in a bank account, at an additional cost. The agreement may specify sanctions and possible reparations in case the two agents do not comply with their obligations, which we refer here explicitly. If SA does not deliver on time, then a fixed amount is to be deducted from the original price of the goods for each day of delay and it should see to it that delivery be made by a new deadline, say within the next 3 days. If BA does not perform payment on delivery, then an additional cost is added to the original price of the goods and payment is arranged to take place within 21 days from the date BA receives the goods, via a deposit in a bank account, at this additional cost. If BA does not perform payment within 21 days, then a 2% surcharge is to be added to the price of the goods for each day of delay. In the same spirit, the second agreement (between SA and CA) specifies obligations, deadlines and possible sanctions/reparations in case of violations. In Table 4.1, we present a non-complete list of normative positions, such as obligations, prohibitions, permissions and institutionalized powers, that are raised among contracting parties in such a business transaction:

---

[8] For the purposes of simplicity and without loss of generality, we discuss the example scenario without noting the distinction between agents and their roles. We consider that each agent having a single role acts on behalf of its representative and we refer to agents by mentioning their roles.

Table 4.1 Normative positions for contracting parties

NORMATIVE POSITION	BUYER	SELLER	CARRIER
Obliged	- to accept requested goods in case of successful delivery - to pay the seller in case of successful product delivery via the carrier agent for the case of the cash on delivery option or via a deposit in a bank account within 21 days	- to formulate goods in the desired form as requested from the buyer and agreed in the contract - to arrange the way and the time of the delivery, i.e. to settle down a new agreement with a carrier that meets the same requirements of its contract with the buyer - to ensure quality of service, e.g. the on time delivery, in the desired form and in acceptable condition - to pay the carrier in the pre-agreed time point/period in the case of a successful delivery of products to the buyer	- to deliver the requested goods - to ensure quality of service, e.g. the on time delivery, in the desired form and in acceptable condition - to accept payment on behalf of the seller in the case of the cash on delivery option
Permitted	- to ask for discount or return goods (the whole amount or a part of it) in the case of contrary to duty actions or delinquencies - to cancel the agreement with the seller at an	- to ask for discount in case of contrary to duty actions or delinquencies, - to cancel its agreement with the retailer or the carrier at an early time	- to cancel its agreement with the seller at an early time

	early time		
Prohibited	- not to accept products after successful delivery - to cancel the agreement at a delayed time	- to cancel the agreement with the buyer or the carrier without informing the third contractual party - to cancel its agreement with the buyer or the carrier at a delayed time	- to deliver/resell the transported products to another buyer - not to accept payment on behalf of the seller in the case of the cash on delivery option - to cancel its agreement with the seller at a delayed time
Institutional Empowered	- to require product delivery in case of ordering - to return the delivered product - to negotiate the seller's (and the carrier's) contrary to duty actions or delinquencies, such as delayed delivery or bad condition of goods	- to require payment for its services, e.g. in the case of a successful delivery of products - to entrust and authorize a third party (e.g. the carrier) to deliver products - to entrust and authorize another party (e.g. the carrier or a bank) for product payment - to negotiate the seller's (and the carrier's) contrary to duty actions or delinquencies, such as delayed payment or bad condition of delivered goods	- to require payment for its services, e.g. in the case of a successful delivery of products - to deliver a product on behalf of the seller - to get paid on behalf of seller in case of the cash on delivery option - to negotiate the retailer's contrary to duty actions or delinquencies, such as delayed payment

Following [55], we may take an informal, process view of the business transaction that is regulated by the two agreements. Each state offers a (possibly partial) description of the factual and normative propositions that hold true in it. A transition between states corresponds to an event that takes place, i.e., an action that one of the parties performs or omits to perform. An abstract description of the business exchange as a state diagram is shown in Figure 4.6, where a double line node denotes the initial state; a bold line node denotes a total successful ending of the agreement; a discontinuous bold line node denotes a total unsuccessful ending of the transaction; and finally, a single line node denotes a midway state of the business transaction.

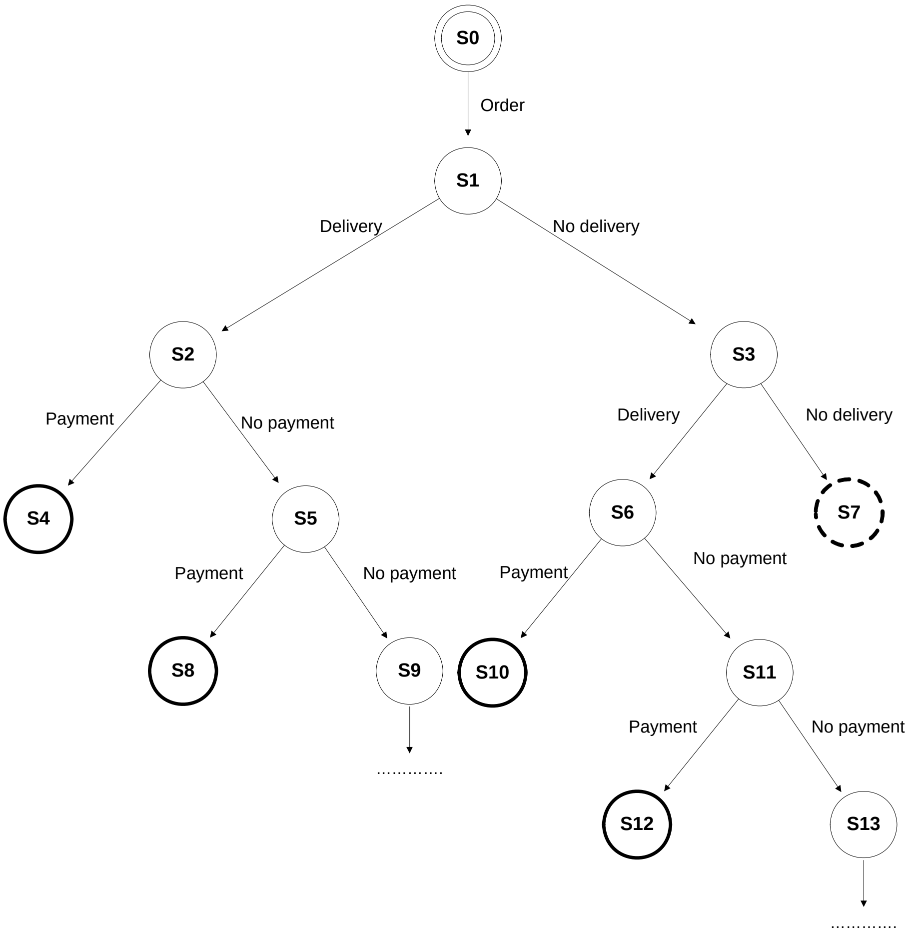


Figure 4.6 E-contract transition diagram

A more detailed part of such a description of the business exchange as a state diagram is shown in Figure 4.7. Initially, at time point T0, the transaction is in state S0 where the two agreements have been established and no events have occurred yet. If BA places an order at some time after T0, the transaction will move to a state S1, where SA is obliged towards BA to deliver goods within 10 days. Also, CA’s obligation towards

SA, to deliver goods to BA on SA's behalf within 10 days, is active. If CA delivers within the specified time bounds, then the business exchange will move to a state S2, where CA's obligation (and SA's obligation towards the BA for delivery, which is related to it) is successfully discharged, and BA's obligation towards SA to pay becomes active (as does SA's obligation to pay CA). If, when the transaction is at state S1, CA does not deliver on time, then the transaction will move to some state S3, where SA must compensate BA as specified by their agreement (and CA must compensate SA as specified by their agreement). In the same manner we may discuss other states of the business exchange.

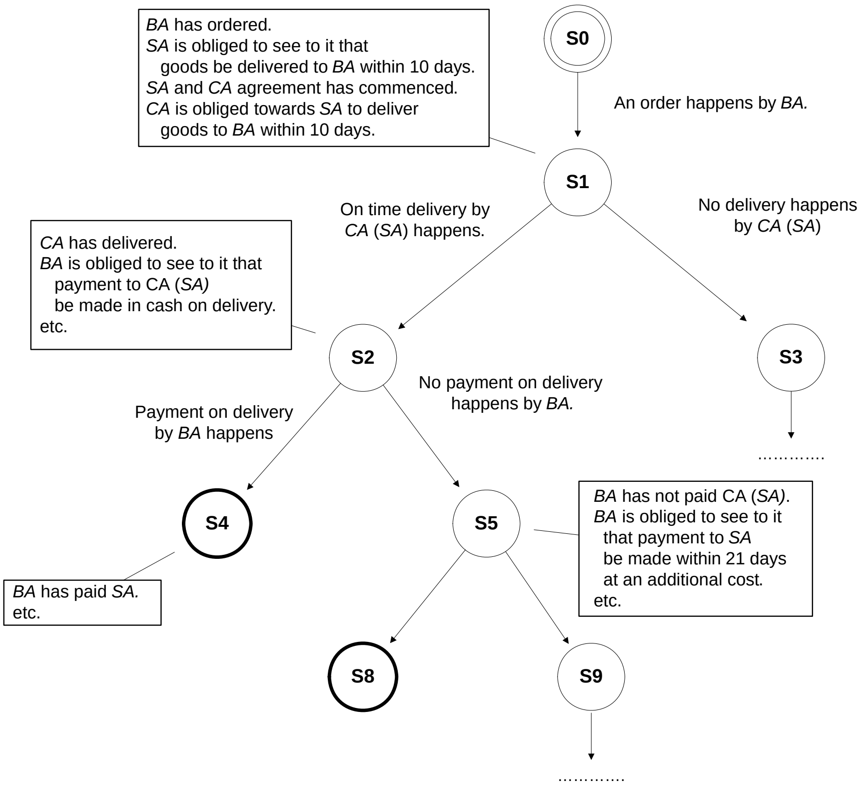


Figure 4.7 Part of the e-contract transition diagram

4.3 Requirements Analysis and Specification

Although the three-party business transaction is superficially simple, it puts on view several important features that are met on more sophisticated and realistic scenaria of transactions, and motivates a discussion about the requirements of appropriate electronic agreement representations in order to facilitate the development of tools that support CSR in OS, such as the e-marketplaces, business environments or virtual communities, and specifically for tools that support contract performance monitoring.

Table 4.2 presents a list[9] of requirements for representation languages and specifications for the development of tools that are essential for CSR in OMAS. Next we discuss each one in turn.

Table 4.2 Requirements and specifications for common-sense reasoning in open environments

ID	TITLE
R1	Ontology representation
R2	Temporal information representation
R3	Deontic Modalities
R4	Legal and Physical ability
R5	The representation of normative violation
R6	Contrary to Duty Structures
R7	Normative conflict representation and resolution
R8	Auxiliary calculations
R9	Default, Defeasible and Nonmonotonic Reasoning
R10	Autonomous and Adaptive Reasoning

R1. Ontology representation: The concepts of the application domain as well as their relations need to be represented explicitly, so that such information may be used during inference.

R2. Temporal information representation: We need a proper formalization of time and temporal information, as noted by many researchers, for instance [248]. Here are some examples that show that temporal reasoning is required. Expressions such as:
“BA places an order at time point T”, or

---

[9] This list is neither exhaustive nor its items are presented in a certain order.

"SA is obliged towards BA to deliver ordered goods within 10 days from the date BA's order happens", or

"At state S1 the CA agent is obliged to delivery goods to the BA agent by time point T+10",

show that we are not only interested in the actual time points at which an action happens or a (normative or descriptive) proposition holds but also in deadlines. Indeed, specifically for norms [166] note the distinction between their so called *internal* and *external* times. We may also need to represent periodic information, for instance:

"RA is obliged to perform installment payment each month".

Specific issues that arise concern the commonsense law of inertia, the representation of the indirect effects of actions, the representation of non-deterministic actions, concurrent actions and so on.

R3. Deontic Modalities: We need to determine what normative relations obtain between parties during a business transaction. Deontic Logic studies such notions, i.e., obligations, permissions, prohibitions and their interrelation [43, 176]. Deontic Logic allows us to disconnect what is the case from what ought to be the case. Such distinction enables us to determine explicitly whether the actual behavior of contractual parties complies with the prescribed behavior. [134, 252, 177].

R4. Legal and Physical Ability: We need to distinguish between the legal/institutional and the actual/physical ability of involved parties to perform actions in order to meet their obligations [165, 136]. Such notions are essential, because they affect which actions are considered as valid and consequently which actions' effects obtain the domain [136].

R5. The representation of Normative Violation: In realistic domains, such as electronic marketplaces, along with the notion of obligation comes the notion of violation. Their relation is obvious. Given a specifications of the agents' obligations during a business transaction, which typically involve deadlines, the e-contract representation ought to facilitate the automated determination of agreement violations. There are many ways in which an agent may violate its obligation to perform an action A by time T, e.g. the agent may perform A but not within the deadline, or the agent may not perform A at all, or the agent may perform some other action B which renders performance of A impossible etc. Similar concerns arise in the case of prohibitions, which, again, may be violated in more than one ways.

R6. Contrary to Duty Structures: Contrary to duty structures are the specification of a primary obligation, along with the specification of a secondary obligation that obtains if the primary one is violated. One may see them as a priori (to contract violation) determinations of recovery mechanisms [44, 200].

R7. Normative Conflict representation and resolution: According to [222] a conflict arises when “(possibly) valid norms establish incompatible qualifications for the same concrete state”. A norm set may be either inconsistent, if a contradiction is logically derivable from it, or potentially inconsistent, if it may lead to contradiction in an upcoming state. In similar spirit in [127, 128] moral conflicts are defined as states where an agent ought to do an action A and, at the same time, it ought to do another action B, but it is impossible to do both. Such situations are often met in business transactions where agents either are in conflict and need a resolution or face a potential conflict and need a plan to overcome this situation or to deal with it in a self-serving manner.

R8. Auxiliary calculations: We need to be able to define and use formulae and procedures that enable the dynamic calculation and re-recalculation of domain concepts, such as deadlines based on relative times, or amount of money for payment.

R9. Default, Defeasible and Nonmonotonic Reasoning: The need for reasoning by default, defeasibly and non-monotonically in legal domains is strongly argued in many research papers, i.e., [223, 66, 41, 114, 199] among others. Different dimensions and interpretations of this kind of reasoning have been discussed in various approaches with respect to the underlying logic that each approach adopts.

R10. Autonomous and Adaptive Reasoning: In norm-governed OMAS, that replicate real world scenaria, reasoning in the presence of incomplete or inconsistent knowledge is usually the case. In such circumstances an agent either needs to plan its *future* activities based on several cases about events/actions that will occur, and that its partners' actions will be valid (*best-guess reasoning*) or even though it may not know everything about the *past* and *present*, it may need to infer information, in order to protect itself from an undesirable situation in the future (*no-risk reasoning*). Thus an agent, first, needs to posses self-knowledge, and, second, needs to self-manage and self-regulate its reasoning by employing appropriate assumptions that fill in information gaps. When more information becomes available, possibly rendering some of these assumptions false, the agent must be able to retract conclusions drawn previously, on the basis of these assumptions. In other words, for an agent in order to perform autonomous and adaptive reasoning, which is also adjusted to its current knowledge, some kind of NMR based on assumptions is required.

4.4 Related Work

In this section we present research efforts that are based on logic and have been proposed during the last decade. The research community that has been concerned with electronic agreements has focused not only on e-commerce applications but also on business process modeling and automation and social norms that govern virtual communities. Our aim is to commit to this paper the fundamental features that each proposed framework contributes. We classify all gathered research approaches in subsections based on the application domain (e-commerce, e-business, virtual communities) and the specific characteristics/features that distinguish them from other frameworks. In the subsection 4.4.1 we record approaches that refer to e-commerce applications, while in subsections 4.4.2 and 4.4.3 we record approaches that refer to applications for business process modeling and automation and virtual communities.

4.4.1 E-commerce

4.4.1.1 Dynamic Logic

In this part we discuss efforts presented in [219, 55, 56, 218, 217]. Those approaches are grouped together due to the common inspiration, which originate from Meyer's dynamic logic formalisation of deontic notions [175], to address reasoning with time and actions.

Santos and Carmo, in [219], propose a set of deontic operators in order to specify the intended set of behaviours that are related with contractual parties. Deontic operators are combined with dynamic operators to represent actions. Furthermore operators present a temporal dimension through their semantics. Desirable behaviours of contractual parties are based on the concept of obligation. Obligations were defined by a special kind of norm, in the juridical context, called prescription. According to [249] a prescription is a command or permission, settled by someone in an authoritative position, towards agents with the intention of inducing or allowing them certain behaviours or conducts. Obligations are examined from the point view of their fulfillment and/or their violation through an a posteriori verification of the actual behaviour. On the whole, the proposed logic mainly addresses compliance with the agreement that is requirements R2, R3 and R5. This work is the first approach in the analysis and representation of contractual obligations and set the basis for subsequent proposals.

In [55, 56] a Modal Action Logic combined with Deontic Logic operators approach was proposed by Daskalopulu *et al.*. A contract is modelled as a process whose state at a given time is determined by the legal relations that stand between contractual parties. Transitions between states are affected by parties' actions. Depending on whether parties' actions comply or violate contractual behaviours, the resulting state is defined as acceptable or unacceptable. An unacceptable state either sets the abnormal ending of the business transaction or is unacceptable in a tolerable way because reparation is possible. Moreover, reparations associated with the violation of obligations are studied and a suitable representation of contrary to duty structures in fault tolerant systems is proposed. To sum up, this approach mainly concerns requirements R2, R3, R5 and R6.

In [217, 218, 216] an electronic agent-based contract framework layered on top of existing B2B frameworks is presented by Sallé *et al.*. The framework was designed to support the whole life-cycle of the contract, which consists of three phases: (i) drafting, (ii) formation, and (iii) fulfilment. In this work, contracts were defined as sets of statements of participant's intentions. The contract specifies the behaviour of contractual parties in ideal worlds as well as in sub-ideal worlds where parties' do not fulfil their commitments. Contract structure was separated in two main parts: (i) an informative section that contains information such as identification number, identities and roles, validity period and a normative system of reference, and (ii) a behavioural specification section which is a set of informative statements that describes the expected behaviour of participants. All contractual obligations are associated with sanctions. This characteristic gives the agent the advantage of a deliberative decision on fulfilling or not a normative statement based on positive or negative effects. Two types of sanction norms were proposed. Endogenous sanctions, i.e., contrary to duty structures, and exogenous sanctions that apply when violations with no specific endogenous sanction occur. To sum up, this approach mainly concerns requirements R2, R3, R5 and R6.

4.4.1.2 Event Calculus

Here we discuss two frameworks presented in [79, 143]. Both approaches adopt a contract representation in Event Calculus (EC) for temporal reasoning and reasoning with actions and their effects [149, 228]. EC is also used in [261, 211, 212] but we discuss these approaches later in subsection 4.4.1.4 because they have another important distinguishing feature.

In [79] Farrell *et al.* present an ontology and a tool to capture issues that are related with contract state tracking for Service Level Agreements. The presented framework is implemented using the Java programming language and is constructed on an XML-based formalization of the Event Calculus, called ecXML. Their main intention is the implementation of a tool, called Event Calculus State Tracking Architecture (ECSTA), which is able to track the effects of various events on different contractual states and to define what normative relations hold between parties on those states. Moreover, a detailed discussion about notions such as obligation, permission (vested permission) and institutionalized power (vested power) and their role in the business transaction is presented. Based on the above analysis, three types of contractual norms are proposed: contract management norms, obligation norms and privilege norms. We discuss the third type which concerns with actions that are permitted to be performed and are not explicitly recorded in the contract. According to Farrell *et al.* any action that is not permitted is considered to be an illegal action. This fact leads us to the conclusion that there is no need for explicit prohibition norms or in other words the absence of permission is considered as the presence of prohibition. On the whole, requirements R2, R3, R4 and R8 are explicitly discussed; we believe that requirements R5 and R6 are also met by this approach, although the authors do not explicitly discuss them.

Knottenbelt and Clark, in [143], introduce a simple Event Calculus representation of contracts and a BDI architecture that supports contract performance. The proposed architecture was built on top of the AgentSpeak(L) [202] agent architecture and enables agents to respond to events in a reactive or a proactive manner based on their active contracts and temporal conditions. During this work two types of contracts were studied. Short-term contracts like the one presented in our example scenario and long-term contracts that define requirements of short-term contract drafting. Communication between agents is possible by exchanging events. An event is considered as the act of sending messages. A well-formed message consists of a time stamp, an identifier, the identifier of the message to which it is a reply, a sender, a receiver, content, context, and the interaction protocol. During messaging exchange an agent is able to evaluate a contract by placing a query on the Event Calculus HoldsAt predicate [149, 228]. To sum up, this approach deals with requirements R2, R3 (only obligations are discussed), R4 (only institutionalized power is discussed) and R5. Moreover, the authors claim that conflict detection and resolution is possible through the work presented in [31].

4.4.1.3 Non-monotonic Reasoning

In this section we discuss approaches that deal with defeasible and non-monotonic reasoning with e-contracts [112, 204, 113, 111, 22, 104, 105, 191, 189]. In what follows we present the main points of those proposals. Note that although the underlying logic language and theory are different, these approaches present many common features due to the interrelation of the adopted logics.

In a series of papers, such as [112, 204, 113, 111, 22] Grosof *et al.* presented a comprehensive approach to the representation of business rules and a series of tools that are integrated in the WWW framework. Specifically, in [112] a declarative approach to the representation of e-contracts rules that is based on Courteous Logic Programs (CLP) is introduced. CLP is an extension of Ordinary Logic Programs (OLP) with prioritized conflict handling. The central purpose of this work is to present declarative contract semantics, to handle potential conflicts with priorities, to represent contract rules with an XML-based encoding and to present a prototype called Common Rules. This works is mainly concerned with the contract negotiation phase and particularly with a suitable contract rule representation for communication during this phase. Moreover, in [112] an XML formalism of CLP rules called Business Rules Markup Language (BRML) and a prototype implementation named Common Rules was also introduced. This work was extended in [204] and an auction-based negotiation tool called ContractBot was introduced. Here a contract representation in CLP rules that consists of two subsets is presented. The first subset, called, proto-contract, contains rules that determine facts and conditions of the overall transaction, such as ways of delivery, payment or reparation, while the other subset contains negotiation-level rules, that describes of what and how will be negotiated. In [111] an overview of all previous efforts is available plus an extension, of the previous work on business rules representation, which is based on Situated Courteous Logic Programs (SCLP) is also introduced. SCLP is the Situated extension of CLP that is characterized of features such as non-monotinicity, that are negation as failure and prioritized conflict handling as presented above and furthermore, procedures for querying on contracts and representing actions. Note that conflict detection is facilitated with the use of mutual exclusions statements, which are statements (pair of literals) that determine contradictory or inconsistent transaction states. On the whole, this approach deals well with requirements R5, R7, R8 and R9, but no temporal representation was adopted in order to facilitate reasoning with time.

In [104, 105] an architecture to represent and reason about e-contracts is introduced by Governatori *et al.*. The system is called DR-Contract and extends the DR-Device architecture (a system for defeasible reasoning on the Semantic Web [20],) with the Defeasible Deontic Logic of Violation (DDLV) [104, 107]. The aim of this approach is to analyze the expected behaviour of the contractual parties and to identify what normative relations arise from an e-contract. Contracts are considered to comprise provisions that determine obligations, permissions, entitlements and other mutual normative positions that hold among contractual parties. Contract clauses are separated in two different types: (i) definitional clauses that define contractual concepts such as “who is a privileged customer” or “what is a special order”, and (ii) normative clauses that contain deontic notions and intend to regulate the whole transaction. The underlying logic that is adopted is Nute’s Defeasible Logic [187]. According to this theory four types of knowledge are considered: (i) facts, (ii) strict rules, which are rules in the classical sense (iii) defeasible rules, which are rules that can be defeated by other rules, and, finally, (iv) superiority relations, which define priority relations among rules. Another point worth mentioning is the fact that this approach also deals, in detail, with the issue of violation of primary obligations and their reparation mechanisms. Contrary to duty structures were represented by introducing a new non-classical connective $\otimes$ [104, 107]. The interpretation of the formula $OA \otimes OB$ is “Obligation B is the reparation of the violation of obligation B”. This connective allows the combination of primary and reparatory obligation in a single regulation and satisfies important properties such as associativity, duplication and contraction on the right that enable reasoning with CTDs. Note that, according to Governatori, the Courteous Logic Programs of the previous presented approach is a notational variable of Defeasible Logic and thus the integration of properties of both approaches is possible. To sum up, this approach covers requirements R3, R5, R6, R7 and R9. No temporal dimension is given via the integration of some temporal logic, but an extension to this direction is feasible as shown in [108, 106, 109].

In [191, 189, 190] Paschke *et al.* presented the ContractLog system, an Extended Logic Program with negation-as-finite-failure and explicit negation. This approach deals with execution and monitoring of Service Level Agreements. SLAs are represented via reactive Event-Condition-Action rules that are enhanced with EC predicates and other special predicates for deontic notions. This work, also, uses Nute’s Defeasible Logic in combination with integrity constrains that express a condition which must always hold. Specifically, ContractLog supports four basic types of integrity constraints: a) Not-constraints that express that none of the stated conclusions should be drawn, b) Xor-constraints that express that the stated conclusions are mutual exclusive, c) Or-constraints that express that at least one of the stated conclusions should be drawn and d) And-constraints that express that all of the stated conclusion should be drawn. Moreover, other logic formalisms such as Description Logic are used to integrate ContractLog to Semantic Web. On the whole, this approach presents ideas that are similar to both previous presented approaches, e.g. ideas for dealing with conflict resolution and nonmonotonic reasoning, but differs on implementation level and seems to address requirements R2, R3, R5, R6, R7, R8 and R9.

4.4.1.4 Commitments

In this section we have gathered and discuss approaches that see e-contracts from a commitment-based perspective [257, 258, 259, 260, 261, 49, 64, 241, 211, 212, 153, 244]. Although the perspective is similar, they vary in what commitments denote.

In [257, 258] monitoring requirements for e-marketplaces and a system architecture are presented. Specifically, in [258], Xu proposed an approach for contract modelling that is based in Temporal Logic. This work aims to facilitate pro-active monitoring and violation prevention. This is accomplished by proposing workflow constrains and guards of workflow constrains that describe different complex relationships among actions and make possible to take the initiative to anticipate and avoid contract violations. Moreover a guard and a pro-active detection algorithm are presented to dynamically monitor business processes. Supplementary to previous papers, in [259] the notion of commitments is added to the formal representation of the electronic contract. A commitment is considered not as a distinct obligation but as a guarantee by one party towards other parties that some action sequence shall be executed completely. This fact is the main difference with the next three approaches. Next to the notion of commitment, the commitment graph is presented that is an overview of commitments between agents. So the commitment graph is a graphical encoding of contract clauses. This graph in cooperation with the two algorithms may point out which partner is responsible for which violations as shown in detail in [260]. On the whole, this approach deals with requirements R2, R3 (via commitments, not via classical deontic notions) and R6.

Yolum and Singh presented in [261] an approach for specifying and executing protocols that regulate multi agent interactions. Such protocols define a set of social commitments (or else commitments) that are assigned to agents. Conceptually, commitments capture obligations arising for an agent towards another agent to bring about a certain property. The business transaction is viewed as a finite state machine where operations (actions) on commitments and business rules are being represented in the circumscriptive version of the Event Calculus language as explained in [229]. Two basic commitment types are considered [230]: (i) Base-level commitments meaning that an agent is committed towards another agent to bring about condition, and (ii) Conditional commitments meaning that if a condition is satisfied then an agent will be committed towards another agent to bring about another condition. Six operations on commitments are used here [230]. Possible transitions in the business protocol can be specified in terms of the Event Calculus language. Once again this approach addresses requirements R2, R3 and R5 and moreover it addresses nonmonotonic reasoning through the circumscriptive version of the Event Calculus. In [49] Chopra and Singh introduce a method to contextualize commitment protocols by modifying them via different transformations. Contrary to previous work [261], protocols are now represented in C+ [102] and illustrated as transition systems. The nonmonotonic and causal character of the C+ action language supports elaboration tolerance. This means that, protocol transformations (accessions, removals or both) are possible simply by adding axioms to an existing protocol specification. Transformers are also protocol specifications and as a result in order to apply a transformation we simply append it to the target protocol specification. Furthermore, in [64], Desai *et al.* propose a more general way to represent and reason about commitments by addressing issues such as (i) complex and nested commitment conditions and (ii) concurrent commitment operations. This is possible, firstly, by specifying business processes as choreographies [82] that can support more complex interaction patterns, and secondly by considering choreographies as commitment protocols. On the whole, the work in [49, 64] deals with requirements R2, R5 and R7 and can been seen as an extension or a supplement to the work presented in [261]. Finally, in [241] an approach to formalize contracts and Virtual Organizations (VO) based on commitments is presented. This work focuses on the interrelation of e-contracts, as a way to model agents’ interactions, and VO formed among contracting agents, where commitments, policies and goals form agents’ relationships. Contracts are considered as static entities that capture relationships among two or more agents, while, VOs are considered as dynamic entities whose membership and structure might evolve, and within which commitments and contracts are manipulated via operations on contracts and commitments. This view of VOs as contexts of agents and contracts facilitates addressing requirements R5 and R7.

In [211] and [212] Rouached *et al.* present (i) a layered contract model, (ii) an approach for regulating Web Services to support cross-organizational collaborations, and (iii) how the integration of contact management services into the overall business process may be facilitated. This work, also, uses the Event Calculus language as presented in [149] to specify the contract state at particular time points. A point that is worth mentioning is that, special terms, expressing temporal relations, are used to express the relation between the occurrences of different events (composite events). As in [230, 261] this work accepts three types of commitments. The third type is the Persistent commitment expressing that an agent is committed towards another agent that some condition holds on all future time points. Here, deontic clauses, such as obligation, permission and prohibition, are defined in terms of operations on commitments using both commitments and EC axioms [229]. With respect to the specified requirements, this approach deals with issues R2 and R3.

In [153] another approach that considers contracts as protocols that regulate business agreements by specifying a set of commitments is proposed by Letia and Groza. Contracts are represented by Defeasible Commitments Machines (DCM), which is a theory in the Normative Defeasible Logic (NDL) presented in [108]. The theory consists of two parts. The first part captures the representation of standard commitments and the possible operations on them in terms of (NDL). The second part includes all contract dependent rules. As in previous commitment-based approaches, this work accepts two types of commitments (Base-level and Conditional commitments). Temporalized Defeasible Logic in combination with time constraints for commitments (deadlines for fulfilment) facilitates the entailment of conclusions about commitment states over time. In this way, besides the gain of reasoning temporally, agents are also able to reason with incomplete knowledge. To conclude, this approach addresses requirements R2, R3 but there is no mention about permission or prohibition, R7 but there are no particular conflict patterns specified, and R9 via Defeasible Logic. In [244], Vartic and Letia intend to verify all stages of a commercial interaction via presenting an agent model based on rules. Agents beside contracts and commitments also possess a set of pre-contacts and pre-commitments that illustrate the interaction during the negotiation stage. Beliefs, goals, obligations, permissions, rights, actions and violations are expressed in a non-monotonic logic, i.e. defeasible logic [16], where only defeasible rules and no strict rules are considered. Furthermore, as in [104, 105, 191, 189] static priorities for rules with contradictory conclusions are used for conflict resolution. This approach seems to addresses requirements R2, R3, R5, R7, R8 and R9 but very little technical detail is available.

4.4.1.5 Linguistic Aspects of e-Contracts

In this section we discuss the work presented in [238, 239]. The particular feature of the work of Tan and Thoen is the fact that it specifies the need for directed deontic notions. It deals with e-contracts from a linguistic perspective and, therefore, this approach mainly concerns with issues R1[10] and R3. [238] addresses some unanswered questions of their previous work where a formal model, called Deontic Deep Structure Model, was presented. According to [238] (i) the ambiguities that derive from the underlying logic for directed obligation [117, 220], as adopted in their previous work, and (ii) its shortcoming to express directed permissions, raised the need for improvement. An alternative definition for directed obligations is presented and a definition for directed permission is proposed. These definitions are based on a conditional operator interpreted as “count as” and an attempt operator, as presented in [136, 221], respectively. Moreover, a different interrelation between directed obligation and directed permission form the one that holds in SDL for obligation and permission is proposed. In their later work, [239], an approach to deal with

---

[10] Note that without any distinction, all gathered approaches, use specific terms in order to deal with e-contacts. Those terms are domain-specific and facilitate dealing with specific open problems. This is the main reason we do not refer in detail the way each approach address the first issue of interest (i.e. ontology). For a more detailed analysis, someone has to study other research approaches which address e-contracting from the perspective of ontologies. This is out of the scope of this thesis.

requirement R2 is presented. Towards this direction a contract representation in the Formal Language for Business Communication (FLBC) [142] was proposed.

4.4.1.6 Hypothetical Reasoning

Alberti *et al.* in [7] adopt the SCIFF abductive logic language to specify business contracts. SCIFF logic language is a mixture of the Abductive Logic Programming (ALP) [4] and the Constraint Logic Programming (CLP) [130]. The primary entities of the language are *events*, which are used to represent contractual notions such as actions or timeouts, and *expectations* that describe the desired behaviour in terms of events. Deontic operators such as obligations, permission and prohibition are related to abductive expectations via a proposed mapping [8]. Their primary goal is to address the problem of runtime verification of contract policies. However, an extension of SCIFF, named g-SCIFF, is defined in order to reason with contract specifications at design time, too. A contract in the SCIFF logic language is described by a knowledge base that captures domain-specific knowledge and by a set of integrity constrains that describe contract clauses. This approach captures the notions of violation and recovery and as stated by the authors conflicts and contradictions are possible to detect at run-time by the proposed notions of E-conistency and ⌐-consistency, but no specific normative conflict patterns are discussed.

ALP extends normal Logic Programming by allowing some predicates to be declared as abducible predicates. Thus, reasoning is based on employing hypotheses on these abducible predicates as possible solutions of problems to be solved. Problems can be either observations that need to be explained or goals to be achieved. The work presented in [6] follows the latter direction on verifying that a requested web service, provided specific input, will lead to a desired state that satisfies a requested goal. Although this approach does not meet directly our requirement R10 on autonomous and adaptive reasoning, we see that it can be used towards this scope.

Finally, a representation of contract rules in RuleML is proposed in order to integrate SCIFF to Semantic Web, and an inference engine called SCIFF Reasoning Engine (SRE) is given.

This approach addresses directly requirements R2, R3 and R5. Also, it seems to addresses requirement R7, but very little technical detail is available, and we believe that it partially addresses requirement R10. Finally, we see that it is possible to address requirement 9 via the relation of ALP with (general) Logic Programs (under stable model semantics [83]) or with extended Logic Programs (under answer sets [84]).

4.4.2 E-business

In this section we have gathered approaches that see contracts from the enterprise perspective. Specifically, in [58, 59, 167, 162] contracts are used to model and manage enterprise business processes, also known as workflows. As can be observed, the main issue those approaches address is temporal reasoning within business processes, while some of them adopt and represent deontic modalities.

In [58], Davulcu *et al.* propose the Concurrent Transaction Logic (CTR) [25] as a language for specifying, analysing and scheduling of workflows. CTR is a conservative extension of the classical predicate logic and as argued, CTR is capable for (i) representing control flow graphs with transition conditions, (ii) representing triggers, i.e. event-condition-action rules, and is (iii) reasoning temporally. The main idea of this approach is a transformation procedure, called Apply, which accepts a workflow specification, consisting of control flow graphs, triggers and temporal constraints, and constructs an equivalent specification in CTR. In [59] an extension, called CTR-S, is presented. CTR-S extends CTR with certain concepts borrowed from the Game Theory. The problem this approach deals with is adversarial situations that arise in service contracting. A typical case is where contractual parties such as buyers and sellers have conflicting goals. For example, the buyer needs to be assured that goods will either be delivered or money will be returned, while the seller needs to be assured in case of contract break the down-payment can be kept.

Marjanovic and Milosevic, in [167], describe some ideas for e-contract modelling. Formal modelling includes (i) modelling of deontic constraints and verification of deontic consistency, (ii) modelling of temporal constraints and verification of temporal consistency. They use the Reference Model of Open Distributed Processing (RM-ODP) [1] that introduces concepts and terminology to produce an enterprise specification. The basic concepts are: (i) the community, i.e., group of people/agents and resources. Precise behaviour is possible in terms of roles; (ii) the contract that defines obligations, permission and prohibitions. Temporal and deontic constraints are combined to verify temporal and deontic consistency. In this direction, visualization and verification of deontic constraints and their consistency is possible via role windows, while verification of deontic consistency is done through time maps.

The Simple Obligation and Right Model (SORM) is presented by Ludwig and Stolze in [162]. SORM provides an abstract and domain independent model for contractual content representation and management of promises denoted in e-contracts. The cornerstone of this approach is the notion of promise. Promises are the matter of subject in the electronic contract. Specifically, the party that promises enters an obligation, while the party that receives the promise holds a right. As mentioned in [162], although a Deontic Logic contract representation is suitable for reasoning about promises and consistency checking, it does not tell us how and when to check entailments for a request or when to check promises. Those issues are addressed in this paper. The main objective of the SORM is to provide a model that supports the monitoring of compliance and fulfilment of the contractual obligations. Towards this direction, contractual obligations and respectively contractual rights are distinguished in (i) state obligation and right, that are obligation and right of parties to maintain a particular state, (ii) obligation to perform a certain action and right to have an action performed, and (iii) option obligation and right to act, that are obligation of a party to tolerate an action performed by another party that has the corresponding right. A suitable representation of those obligations and right types is proposed in the SORM framework. Finally, certain operations performed on the set of active obligations and rights are discussed in order to capture the dynamics of the domain. This approach should be seen with respect to previous works, such as [110, 126], where the CrossFlow architecture is presented. CrossFlow is a contract-based framework that supports the dynamic establishment and enactment of a business relationship between two organizations.

Cardoso and Oliveira, in [33, 34, 35], describe how to represent and use norms in order to formalize cooperation agreements and operational contracts and to ensure contract monitoring and enforcement. The whole approach takes place in an electronic institution and specifically refers to the B2B field, mainly regarding the formations and handling of Virtual Organizations. In such an environment they distinguish three types of norms. Institutional norms regulate the behaviour of agents in the institution; constitutional norms describe the foundation of agents' Virtual Organization, which thereby commit to a certain agreement; and finally, operational norms specify contracts by indicating actions to be performed by contractual agents. Note that institutional norms pre-exist, while constitutional norms are created when agents reach an agreement and operational norms come into existence only when executable contracts are signed. An important notion in this work is the "institutional reality" [227]. Brute fact and institutional facts are considered along with constitutive rules that define "count-as" relations in order to distinguish between what is said and what is taken for granted, always in a specific context. Contract Law "default rules" [53] are also used allowing contracts, firstly, to be underspecified by defining default clauses, and secondly, to address CTD structures by defining default procedures [56]. Moreover, timestamps, directed obligations and fulfilment and violation detection rules are considered towards a complete contract norm representation.

4.4.3 Virtual Communities

Here we discuss approaches that see contracts as a way to regulate agent societies [63, 70, 31, 69, 23, 256, 5, 251]. Social contracts are considered as a set of norms, rules, commitments or conventions that coordinate and manage the society behaviour. Generally, social contracts are dynamically determined and stipulated by autonomous agents according to their own internal aims and architecture. It is out of our scope to study the problems and specifications of agent societies, thus we examine only what those approaches consider as contracts and the way they use them.

Dellarocas, in [63], presented a system, called Contractual Agent Societies (CAS), where agents may configure themselves and manage their activities through social contracts. Here contracts include beliefs, values, objectives, protocols and policies. Specifically, a social contract is a social commitment, which is agreed and established among agents, and it forms a particular social relationship and, more importantly, it regulates agents' behaviour [37, 133, 230]. An important part of this approach is the social control system, which is responsible for avoiding, detecting and resolving deviations from ideal behaviour via incentives (positive or negative sanctions) and sentinels (commitment monitors).

In [69, 70], Dignum *et al.*, presented a framework for agent societies, called OperA (Organizations per Agents). It consists of three different models. The interrelations between models are described by means of contracts. Here, two types of contacts are described: (i) social contracts that specify commitments between an agent and the society, and (ii) interaction contracts that specify agreements between individual agents. Note that, in this approach, the notion of the social contract differs from the one presented in Dellarocas [63] where both social and interaction contracts are merged into the social contract notion. Both types of contracts are represented through the Logic for Contract Representation (LCR) language [71], that is based on the Temporal and Deontic Logic (BTLcont) [68] and the branching-time temporal logic (CTL*) [76]. Based on this logic, formulae are represented as branching structures where nodes represent states and arcs represent events. The logic is extended with special operators to address issues such as (i) what is the agent's view on the consequents of actions, (ii) deontic modalities, e.g. obligations, and their violations, (iii) conditional obligations with deadlines, and (iv) CTD imperatives [69].

In the same spirit, Boella and van der Torre [23], address the problem of regulating societies of agents and agents via contracts. Here, contracts are modelled as legal institutions [214]. Boella and van der Torre present three reasons to argue that although most normative systems identify norms with obligations, permissions and prohibitions, this approach is not efficient for complex normative systems. Thus, they formalize obligations in terms of desires and goals, and constitutive rules as beliefs. Constitutive rules create obligations when a contract is stipulated or when some relevant event happens. This notion is close to the conditional obligations as presented in [70]. In an earlier work, Broersen *et al.* are interested in conflicts arising between an agent's beliefs, obligations, intentions and desires [31]. In this approach they use normal default rules [208] to detect conflicts, and priorities that stand among mental states to accomplish conflict resolution. However, they do not address conflicts in a temporal setting.

Wooldridge and van der Hoek, in [256], investigate the relationship between Alternating-time Temporal Logic (ATL) [10] and Deontic Logic that is the link among ability and obligations. Towards this direction, they introduce a variation of ATL called Normative ATL* (NATL*). In this logic, powers and coalitions of agents are seen through the perspective of a normative system, which is a set of rules that constrain the actions of the agents in the system in certain states. They introduce indexed modal operators for permission and obligation and, more importantly, they show how these operators shall be interpreted in terms of normative ability. NATL* is used to formalize the model of the social contract, i.e. the multi-agent system and the social law. Later, Ågotnes *et al.*, in [5], present a descendent of NALT* called Normative Temporal Logic (NTL). NTL is simpler than NALT* and a generalization of the temporal logic called Computational Tree Logic (CTL) [76]. In the same spirit, the path quantifiers – quantifier A “on all paths” and quantifier E “on some path”- are replaced by contextual deontic operators for permission and obligation. Thus, in, both, NALT* and NTL, the deontic operators are contextualized and have a temporal dimension. To conclude, the main issue this work focuses is the link of requirements 2 and 3. Furthermore, in their future work section, they argue about the need to examine under the scope of NATL* the CTDs structures. This remark is based on Prakken’s and Sergot’s [200] argument that many of the CTDs paradoxes can be solved within a temporal perspective. Finally, Walther *et al.*, in [251] introduce Alternating-time Temporal Logic with Explicit Strategies (ATLES) as a variant of Alternating-time Temporal Logic (ATL) [10] and as an extension of Counterfactual ATL (CATL) [242] and Action Logic [26]. In this work, the cooperation modalities of ATL are being extended with a commitment function, which ensures that agents cooperate according to a specific strategy.

4.5 Summary

In this chapter, a survey and classification of logic-based approaches that have emerged during the last decade for contract representation and performance monitoring, was attempted. Through this attempt, besides the critical review, we have also derived and recorded requirements that a tool for e-contacting should attend.

In the Appendix A we provide a summary of surveyed approaches that have emerged during the last decade, are related to contract performance monitoring and are based on logic. Each approach is summarized with respect to (i) its goal and main aspects of interest, (ii) the recorded requirements for efficient reasoning with electronic contracts and (iii) its integration to Semantic Web and tool presentation. Furthermore, Appendix A, contains a review and summary of the research presented in this thesis as it is attempted to place this thesis in the overall context of the e-contracting research approaches.

Research discussed in this chapter has been published or is under review in [96, 88].

5 Reasoning with Incomplete Knowledge

5.1 Introduction

In this chapter, we identify and discuss explicit research questions that arise in open norm-governed environments, where agents seek to establish missing information, and present the *Open Default Assumption*. The notion of the *Open Default Assumption* refers, not only, to the ability of agents to identify and employ assumptions dynamically, on the basis of their current knowledge, but also to their ability to manage their inference on the basis of these assumptions and any available knowledge at previous or future time points. It is inspired by the syntax and semantics of DfL and uses some of its several variations along with *Dynamic Default Logic*, i.e. a variation of DfL presented here, towards dynamic hypothetical non-monotonic reasoning. In the next chapters we show how this proposal applies in open norm-governed multi-agent systems and facilitate agents to reason autonomously either hypothetically or non-monotonically.

The rest of the chapter is organized as follows: section 5.2 examines the reasoning problem in open norm-governed multi-agent environments and identifies research questions and situations that preoccupy the following chapters of this thesis; section 5.3 presents the reasons that determined our decision to adopt DfL and our viewpoint in the usage of default reasoning in open environments; and, finally, section 5.4 provides a summary.

5.2 The Reasoning Problem

Consider an OS populated by software agents, whose behaviour is regulated by norms in which some temporal logic is employed. For the sake of simplicity and without loss of generality, in this chapter, we need not relate the discussion that ensues to a specific temporal language, since any temporal logic may be used[11]. We can take an abstract view of the norms and regard them as sentences of sequent calculus [132], i.e. as sentences of the form:

$$Y \leftarrow X_1 \wedge X_2 \wedge \ldots \wedge X_k \qquad (1)$$

where Y and X_i $(1 \leq i \leq k)$ are positive or negative literals (any variables are assumed universally quantified) representing the rule *conclusion* and *conditions*, respectively.

Obviously agents in any system (not necessarily open) do not possess information about the future. In order for an agent to meet its design goals, though, and plan its course of action at any given time, it needs to employ assumptions about the future. In OS, even the historical information available to an agent when it poses its query may be incomplete, for various reasons: Information may be lost, or distorted by noise, and in a truly open system, where agents join or leave the system at different times, information delivery from agent to agent may simply be delayed. In order to reason in the presence of incomplete historical knowledge, agents must be able to fill in information gaps, by employing assumptions about the past and the present.

We see that, the reasoning problem faced by an agent in this context, when it does not possess complete knowledge about the past, present or the future, involves the following issues of interest:

1H. Assumptions Identification and Usage: How can agents make assumptions, i.e.:
  a. When assumptions are needed in order to continue with inference?
  b. What assumptions are applicable to fill in information gaps?
  c. How assumptions should be employed in the inference process?

2H. Assumptions and the World: What is the relation between the assumptions and the current or future world, i.e.:

---

[11] Issues that concern reasoning with time, action and their effects are discussed in chapter 10, where we adopt a contract representation in EC [149].

  a. How do assumptions, employed at some time point, are allied with currently available knowledge?
  b. How do assumptions, employed at some time point, affect subsequent inferences, either supporting some or frustrating others?

3H. <u>Assumptions and New Information</u>: What happens when new information becomes available at some time point, i.e.:
  a. How does new information affect previously employed assumptions and drawn conclusions on their basis, either confirming some or disproving others?
  b. How does new information affect subsequent inferences, either enabling some or disabling others?

Essentially, to answer question 1H the agent in an open environment, where knowledge is dynamically become known or unknown, seeks to establish some way that identifies possible assumptions and indicates the appropriate ones when needed. In order to answer question 2H the agent needs to employ some way that commits its reasoning to specific assumptions. Finally, in order to answer question 3H the agent needs to reason non-monotonically.

In order to support reasoning with incomplete knowledge an agent could use some of the general approaches such as the CWA or the OWA, as already discussed in the subsection 2.2.3 Assumptions made under the CWA concern the *falsity* of certain missing formulae, rather than their *truth*. Under the OWA an information gap is assumed to be unknown by default. In this work we are concerned with situations that an agent may find itself in many realistic computer applications (e.g. distributed problem solving, task/resource allocation scenaria, joint planning, autonomic computing, risk management, e-contracting, e-auctions, services negotiation and composition, amongst others) where its indecisive is undesirable.

Specifically, we are interested in situations where the agent needs to make *specific* assumptions about the *truth*, rather than the falsity, of certain formulae. We claim that such reasoning may be useful in two cases:

- *Best-guess reasoning*: An agent cannot know the future, yet it may need to plan its activities on the basis of hypotheses that concern the future, i.e., on the assumption that certain events or other agents' actions will occur, or that certain causal relations will be effected in the environment, or that it will bear a certain normative status (obligations, permissions, prohibitions, powers) towards other agents.

- *No-risk reasoning*: An agent may not know everything about the past and present, i.e., the history of its environment, other agents and itself so far, yet it may need to plan its activities on the basis of hypotheses that concern the past and present, i.e., on the assumption that certain events or other agents' actions have occurred, or that certain normative relations have obtained between itself and other agents, in order to protect itself from an undesirable situation in the future.

To illustrate these cases, consider a 3-party business transaction that takes place in an electronic marketplace populated by software agents. A buyer agent (BA) communicates, at time point T (T0<T<T1), with a seller agent (SA) and establishes an agreement with it for purchasing a certain product. Consequently, SA communicates with a carrier agent (CA) and establishes another agreement with it for the timely and safe delivery of goods to BA. A reasonable query that the buyer might have is "When will I, potentially, have to pay for this order, assuming all goes well and I receive the goods in due time, so that I plan to have adequate available funds?" To derive an answer the buyer needs to reason on the basis of hypotheses, i.e. best-guess reasoning. Moreover, consider the case where, at time point T2, the buyer agent does not yet know that the carrier agent has performed delivery, still it needs to plan its business activity so that it may be able to fulfil an obligation to pay the seller agent in due time, should it later be informed that the carrier agent delivered at T' (T1<T'<T2). This situation corresponds to no-risk reasoning, i.e., an agent should be able to derive a conclusion even though this is based on assumptions, because alternatively it might find itself in an undesirable situation. Therefore, it is clear that an agent should be able to establish potential conclusions on the basis of hypotheses.

There are, of course, various other approaches to dynamic assumption-based reasoning, which we discuss, in relation to our work, in chapter 8. At this point though, note that these approaches rely either on the existence of a pre-specified space of assumptions or on pre-specified criteria for the identification of assumptions. In the first case, assumption identification is not really dynamic, rather assumption usage, i.e. the management of the pre-specified assumption space, may be dynamic. In the second case assumption identification is dynamic, only in the sense that the appropriate assumption is chosen at run-time, but since this choice is made on pre-specified criteria, it is in a sense static. What distinguishes, therefore, our work from these approaches, is that we propose a way in which both identification *and* usage of appropriate candidate assumptions are done dynamically.

Moreover, since, under CWA or OWA, any assumptions employed at some point of the inference process are not retained for future reference, there is no way to relate them to future inferences. Hence with CWA or OWA we cannot address 2H satisfactorily. When new information becomes available, possibly refuting some of the assumptions that were employed at earlier points in the inference process, there is no way to retract previously drawn conclusions, that is CWA or OWA does not address 3H satisfactorily. Of course, one may argue that such questions can be addressed, in a domain-specific manner, via the use of special purpose predicates (e.g. by recording assumptions used during the inference of each specific conclusion). However, we argue that by resorting to Open Default Assumption we obtain a more general-purpose solution to the problem of dynamic assumption identification, which is also compatible with our common intuitions.

The *Open Default Assumption* (ODA) is the presumption that the truth-value of a statement that is not currently known may considered to be *true* if this does not cause an inconsistent view of the world. With respect to other presumptions (CWA and OWA) the proposed one can be seen as an opposite assumption to the CWA, i.e. we accept the truth rather than the falsity of statements, and as a particularization of the OWA, i.e., we focalize on the truth rather than the unknown state. Under the ODA, an agent reformulates all initial norms that regulate an OMAS into DfL rule schemata [208].

5.3 Why Default Logic?

DfL is arguably the most notable formulation for default reasoning (cf. [14, 154]) and addresses general issues, such as negation by default, the frame problem and causal reasoning, satisfactorily [154]. Also, it is suitable for prototypical, no-risk, and best-guess reasoning, all of which interest us [14].

In this thesis, we were inspired by the DfL due to the following three main reasons:

- The syntax of DfL offers a intuitive way to relate knowledge (i.e. information about which an agent is certain) with hypotheses (i.e. information that the agent may employ tentatively) during the inference process; the schema of DfL rules comprises three distinct parts, namely prerequisites, justifications and consequents, that is it helps in addressing issue 1H.
- The semantics of DfL and its variations offers the ability to preserve the relation between an assumption and inferences drawn on its basis, as well as the relation of new information that becomes known, possibly necessitating the revision of past assumptions and conclusions drawn on their basis (in the sense of argumentation [198]) and to maintain consistency and rationality, that is it helps in addressing issues 2H and 3H.
- We can implement the inference mechanism of DfL without resorting to theorem proving, but by maintaining syntactically consistent sets of formulae, whose conditions part (prerequisites and justifications) is interpreted conjunctively and the conclusions part (consequent) is interpreted disjunctively, as in sequent calculus.

5.3.1 Default Logic for Common-sense Reasoning in the Context of Open Environments

Consider the following DfT (W, D) with $W=\{ P1, P2, P3 \}$ and D contains the following defaults (Example 3, Section 3.3):

$D=\{ D1 \equiv P1{:}J1/C1,\ D2 \equiv P2{:}J2/C2 \}$

where $C1$ and $C2$ are formulae that render the knowledge base inconsistent when both of them hold simultaneously. In this case there are various computations (possible extensions) for In and Out sets: $In(1)=\{P1, P2, P3, C1\}$, $Out(1)=\{\neg J1\}$ for $\Pi(1)=\{D1\}$; $In(1)=\{P1, P2, P3, C2\}$, $Out(1)=\{\neg J2\}$ for $\Pi(1)=\{D2\}$; and $In(2)=\{P1, P2, P3, C1, C2\}$, $Out(2)=\{ \neg J1, \neg J2\}$, for $\Pi(2)=\{D1, D2\}$[12].

This example sets some research questions when using DfL for CSR. A first question that arises is “Which of the above computations may be considered to be

---

[12] We may also consider the process $\Pi(2)=\{D2, D1\}$.

successful and closed?". A second question that arises is "In order to facilitate CSR, which of the above computations may be considered as a theory extension?".

According to Antoniou, the processes $\Pi(1)=\{D1\}$ and $\Pi(1)=\{D2\}$ are not extensions of the DfT, due to the fact that these processes are not closed processes. On the contrary, the processes $\Pi(2)=\{D1, D2\}$ and $\Pi(2)=\{D2, D1\}$, are considered to be extensions (successful and closed processes) of the DfT. But in this case, an inconsistency arises in an agent's world.

We see that the answers to the above questions are related to the type of the agent one needs to adopt and use in its framework. For instance, a *bureaucrat*[13] agent, that computes extensions as proposed by Reiter or Antoniou, accepts as theory extensions only the processes $\Pi(2)=\{D1, D2\}$ and $\Pi(2)=\{D2, D1\}$, despite the inconsistency. On the contrary, a *non fault-tolerant* agent would reject both processes ($\Pi(2)=\{D1, D2\}$ and $\Pi(2)=\{D2, D1\}$) due to the semantic inconsistency that arises.

In this thesis, in order to facilitate CSR, we adopt agents of none of the above two kinds. We see that agents should be *open-minded*, i.e. to be: (i) *rational*, in the sense that, to be able to avoid inconsistencies, and (ii) *autonomous*, in the sense that, to be able to *compute* and *examine* possible worlds on the basis of their currently available knowledge and direct their reasoning accordingly. Moreover, our agent aims to reasons with common-sense in a non-sense world, i.e. a world that is incomplete and possibly inconsistent. In such environments, first, we should not blame the agent for the lack of extensions due to the non-sense of the world. In fact a DfT may not have any extensions at all, but we see this not as a shortcoming. Second, although the possibility of the non-existent of extensions, our agents attempts to search its future worlds or tries to explain it current state on the basis of hypothetical previous worlds. In an open world where the environment and whatever is associated with it are matters of continuous change, we should not require or expect from the agent to either accept worlds that belong to the distance future or reject them, in advance. It seems rational and realistic to us, to require from an agent to reason in a step-wise manner and on the basis of its current knowledge and some plausible and rational assumptions. Such an agent would compute all possible worlds, examine the state that these worlds reveal (consistent/inconsistent, rational/irrational, eligible/non-eligible,

---

[13] A bureaucrat agent is an agent that lacks of rationality and autonomy.

desirable/non-desirable etc) and finally decide to either to follow or not a specific course of action. In other words, we see that CSR in OS calls for an iterative reasoning process where the agent gains knowledge about its current state in the available world, commit itself to this world, produce entailments either on factual or on hypothetical basis and re-examine its world for any changes due to exogenous or endogenous factors. A schematic representation of this process is shown in Figure 5.1. The overall reasoning process is represented like an Archimedean Spiral where the agent performs:

- *Initialization/Position/Reposition*: The agent gets its initial world; position itself in it by gaining self-knowledge. Whenever changes are being detected the agent is able to reposition itself in the world.
- *Inference*: The agent derives conclusions either on the basis of factual knowledge or hypotheses.
- *Update*: The agent checks if the world has changed either due to endogenous or due to exogenous reasons.

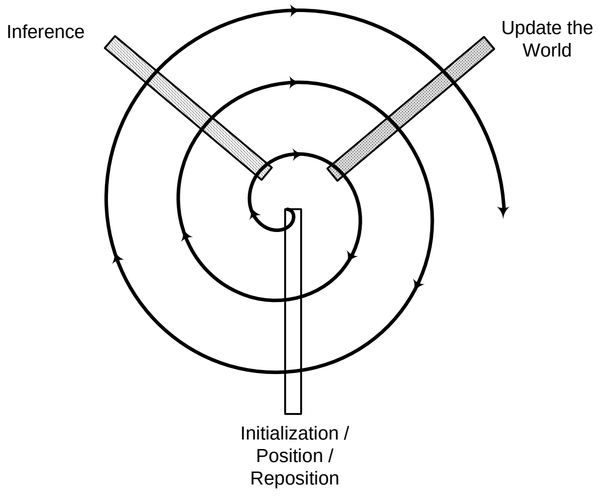


Figure 5.1 Agent reasoning in open environments

Specifically, with respect to the above example, our agent may accept the processes $\Pi(1)=\{D1\}$ and $\Pi(1)=\{D2\}$ as theory extensions, on the criteria that: (i) the next step in inference, i.e. applying defaults $D2$ or $D1$ respectively, if followed, would turn itself into an irrational agent, and (ii) to ground itself in a protracted period of idleness is not the reason of its development (Figure 5.2). Thus, the agent may place itself in either one of the two states ($s1$ or $s2$) and continue with inference from this state on a new basis, i.e. a new current world that probably contains different factual and prescriptive knowledge.

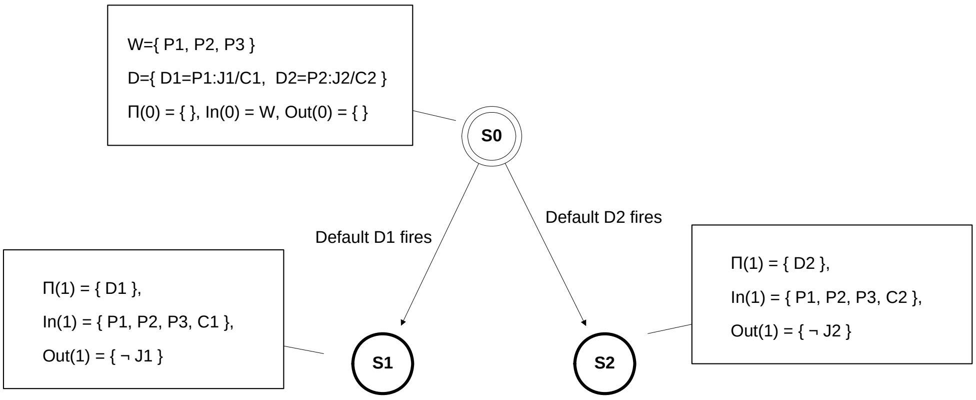


Figure 5.2 Extensions in an open environment

5.3.2 Dynamic Default Logic

Dealing with incomplete information, calls for *assumption*s employment in the inference process in order to fill in information gaps. In this thesis we propose and use the *Dynamic Default Logic* (DDfL). DDfL accepts the ODA, hence, in DDfL it is possible that the initial default rules to be reformulated appropriately when needed. As a result, inferencing is possible on a totally hypothetical basis via the dynamic identification and employment of appropriate candidate assumptions, i.e. *extended* versions of extensions are possible to be computed.

Specifically, in DDfL starting from a default rule of the form $X_1 \wedge X_2 \wedge \ldots \wedge X_k$: true / Y, (i.e., a justification-free default rule where $x_1 \ldots x_k$ are propositional terms) 2^k-1 new possible defaults may be derived by augmenting its justifications (the J set) with prerequisites (members of the P set) whose truth-value can not be established on the basis of current knowledge. That is, for each initial default rule that has k prerequisites in the set P correspond 2^k-1 new possible reformulated defaults.

To illustrate this idea, schematically, consider the following initial defaults:

$D1 = X_1 \wedge X_2 \wedge X_3 : true / Y_1$, and

$D2 = X_4 \wedge X_5 : X_6 / Y_2$

The corresponding sets of alternative defaults for each one of initial defaults are:

D1: $\{ D1_1 = X_1 \wedge X_2 \wedge X_3 : true / Y_1,\ D1_2 = X_1 \wedge X_2 : X_3 / Y_1,\ D1_3 = X_1 \wedge X_3 : X_2 / Y_1,\ D1_4 = X_2 \wedge X_3 : X_1 / Y_1,$
$D1_5 = X_1 : X_2, X_3 / Y_1,\ D1_6 = X_2 : X_1, X_3 / Y_1,\ D1_7 = X_3 : X_1, X_2 / Y_1,\ D1_8 = true : X_1, X_2, X_3 / Y_1 \}$

D2: $\{ D2_1 = X_4 \wedge X_5 : X_6 / Y_2,\ D2_2 = X_4 : X_6, X_5 / Y_2,\ D2_2 = X_5 : X_6, X_4 / Y_2,\ D2_3 = true : X_6, X_4, X_5 / Y_2 \}$

$D_{lnumber}$ denotes an identification number within the defaults and it is used to facilitate reference.

Hence, for DDfL a *Dynamic Default Theory* is a pair of the form (W, D), where W is a set of logic formulae that represent currently available knowledge, as in original Reiter's DfT, but D is a set that contain sets of defaults, each containing the possible reformulations of the initial defaults. Note that, during reasoning for each initial default rule only one of many candidate reformulated defaults may be employed.

Reasoning starts with the initial form of the defaults, by applying as many as possible given the initial current knowledge. Each time a default applies its conclusions are included in the current extension that is being computed. Until now, the inference procedure is identical to inference with classical DfL.

When there are no more defaults that can be applied, this signals that further worlds can, only, be computed on the basis of *new* assumptions. Thus inference continues by examining the alternative default formulations that exist in the set of alternative defaults for each initial default that have not fired, already. The algorithm is shown as a flowchart in Figure 5.3 (Algorithm 1). At this point we do not provide more details of how the alternative defaults are being computed or the way these are applied in the theory. Such issues are discussed in the next chapters.

To illustrate the reasoning process, consider the previous example with the two norms $D1$, $D2$ and the corresponding sets. Here are some possible scenaria, with different initial knowledge available each time, in the beginning of the reasoning process:

- if $W=\{ X_1, X_2, X_3\}$ then according to classical DfL the only extension that is computed is $In(1)= W \cup \{Y_1\}$, $Out(1)=\varnothing$ by applying defaults $D1_1$. In contrast, allowing hypothetical reasoning with DDfL then we may initially compute the previous extension $In(1)$ on a factual basis and the extension $In(2) = In(1) \cup \{ Y_2\}$, $Out(2)= Out(1) \cup \{\neg X_6, \neg X_4, \neg X_5\}$ by applying defaults $D1_1$ and $D2_3$ respectively on a hypothetical basis.

- if $W=\{X_1, X_3, X_4, X_5\}$ then according to classical DfL the only extension that is computed is $In(1)= W \cup \{Y_2\}$, $Out(1)= \{\neg X_6\}$ by applying defaults $D2_1$. In contrast, under DDfL also the extension $In(2)= In(1) \cup \{Y_1\}$, $Out(2)= Out(1) \cup \{\neg X_2\}$ is computed by making an additional assumption that X_2 holds and by applying defaults $D2_1$ and $D1_3$ respectively.

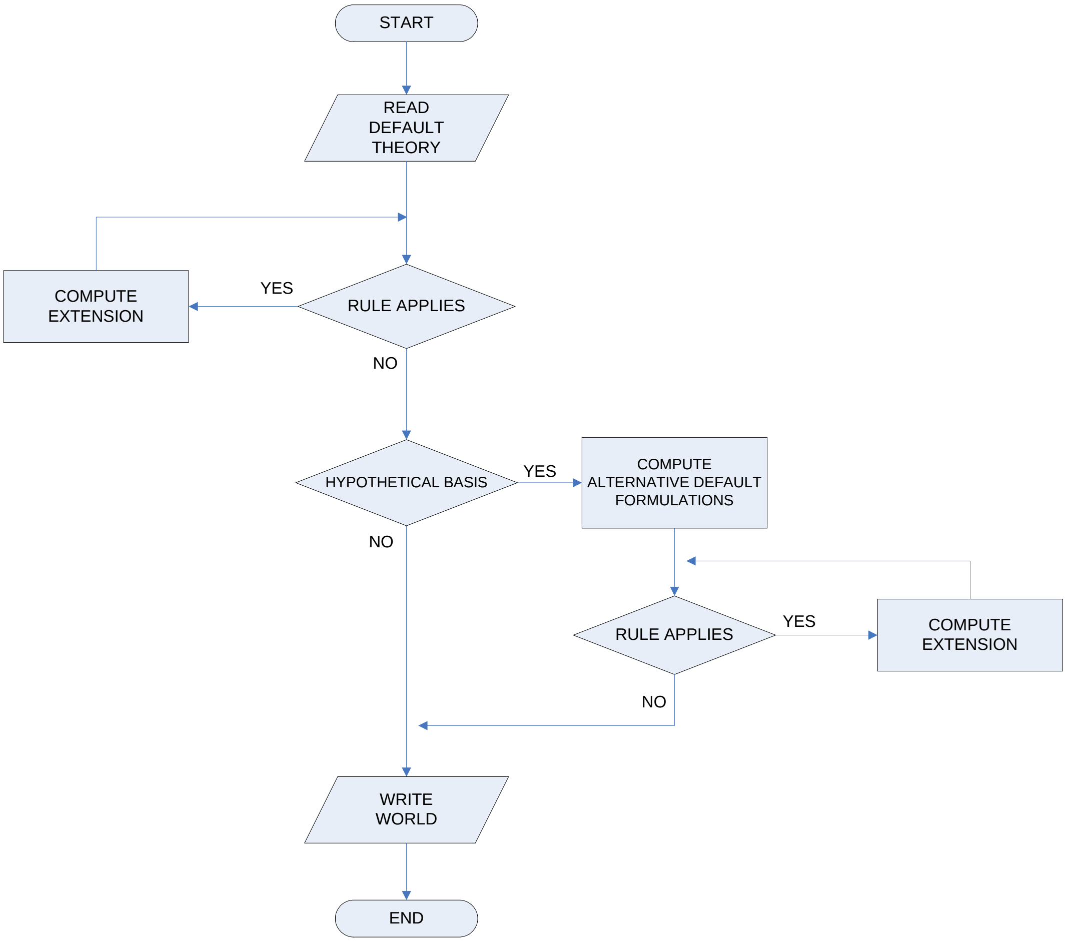


Figure 5.3 Algorithm 1: Default reasoning in open environments

For FOL language we need to identify a substitution instance for each prerequisite chosen as an appropriate assumption before employing it in the inference procedure. To this end, we may adopt *Herbrand semantics* for the FOL language [137, 138]. The *Herbrand universe* of a FOL language is the set of all ground terms. The *Herbrand base* of a FOL language is the set of all ground atoms formed using elements of the Herbrand universe as arguments. Thus, we may identify possible grounded assumptions by searching among grounded formulae computed on the basis of the Herbrand semantics.

Consider the following theory consisting of default schemata of the form:

D1 = Has(x,Hair) ∧ Produce(x,Milk) : true / Specie(x,Mammal),

D2 = Specie(x,Mammal) ∧ Color(x,Brown-Orange) : Has(x,Stripes) / Is(x,Tigger)

and initial knowledge:

W= { Has(Animal,Hair) }

On the basis of this knowledge and without resorting to any assumptions inference is not possible with classical DfL. In contrast, under the DDfL we are able to compute a possible world:

- In'(2)= W ∪ { Has(Animal,Hair), Specie(Animal,Mammal), Is(Animal,Tigger) } on the basis of the assumptions that Animal produces milk, its color is brown-orange and has stripes, i.e., Out'(2)={ ¬Produce(Animal,Milk), ¬Color(Animal,Brown-Orange), ¬Has(Animal,Stripes) }

Naturally, whenever we compute ground instances for potential assumptions, under the Herbrand semantics, we are forced to ground inference to whatever is recorded in the current logic language we use. In other words we accept the *Domain Closure Assumption* [137, 138]. A question that arises is whether under this assumption we are truly capable to deal with OS? For instance, consider an OMAS where agents join and leave the environment in an ad hoc manner. An agent, in order to employ assumptions about other agents, first needs to know that other agents exist. Thus, we see that, in order to facilitate hypothetical reasoning we may accept and consider the Domain Closure Assumption as putting the world in quarantine temporarily. In this case we accept a partially open environment, even temporarily. Of course, an agent may also be allowed to make assumptions about the existence of other agents by naming them explicitly and by augmenting its logic language. In this case, we allow inference in a truly open system.

During inference, we need to remember which default formulation we chose for each of the norms that we reason with. When new information becomes available, either merely augmenting the knowledge base or updating some part of it, we need to update the choice of default formulations.

Although, initially, it seems that this mechanism gives a blow up in the search space of possible defaults to apply, we see that this space is manageable. Indeed, in the next sections we propose and show various techniques to manage this space.

Moreover, note that, with DDfL we extend the extensions of DfL, e.g. in the case a theory has no extensions (besides its initial world) under DDfL extensions are possible on a hypothetical basis. As a result, in DDfL the problem of the existence of extensions is downsized only on the requirement of a process Π to be a successful one and that depends only on whether $In(i) \cap Out(i) = \emptyset$ or not. The requirement of the closed process has no meaning due to the ability of DDfL to apply more rules on a hypothetical basis and under consistency maintenance.

Finally, note that the process-oriented technique to compute extensions as presented in [14], along with the technique to employ dynamically additional justifications in the reasoning process as presented here, afford an agent the ability to reason:

- with incomplete knowledge in open environments, by deriving conclusions that are based not only on the justifications of the initial default rules, as occurs in classical DfL, but also on the basis of additional consistent assumptions engaged from the rule prerequisites. This is possible by reformulating, dynamically, the theory initial default rules. Towards this scope, in the next sections, we address assumption-based reasoning in open systems where the norms that govern the environment are sentences of the form (1). This fact, first, enable an agent to construct autonomously its norms/ defaults of inference from initial temporal norm representations, and second, to avoid any misinterpretations on the semantics of the alternative defaults that correspond to an initial norm.
- hypothetically and non-monotonically in an autonomous and argumentation-like manner by maintaining syntactically consistent sets of formulae. Towards this scope, in the next sections, we show how various symbolic and schematic representations of an agent knowledge/hypothesis space facilitate the management of both hypothetical and non-monotonic reasoning.

5.4 Summary

In this chapter we examined the reasoning problem in open norm-governed multi-agent environments. Towards this scope, we identified three research questions (1H-

3H) that concerns the identification and usage of assumptions and the manner these assumptions are related to the currently available knowledge and any possible changes of the world. Moreover, we claimed that in some realistic situations where best-guess or non-risk reasoning is needed, it is essential for an agent to be able to make specific consistent assumptions about the truth, rather than the falsity, of certain formulae. We called this behaviour as the Open Default Assumption. Then we discussed the reasons that determined our decision to adopt DfL along with our differentiations in the usage of default reasoning in open environments; and, finally, we presented Dynamic Default Logic which accepts the Open Default Assumption, and hence, it is possible for an agent to reformulate, appropriately, its world representation when needed.

In the following chapters we show how these ideas facilitate both hypothetical and non-monotonic reasoning.

Research discussed in this chapter has been published or is under review in [90, 97, 98, 86].

6 Assumption-based Reasoning in Open Normative Environments

6.1 Introduction

In this chapter we address dynamic assumption-based reasoning in open agent systems, where, unavoidably, agents have incomplete knowledge about their environment and about other agents. The interactions among agents in such systems are typically subject to norms, which stipulate what each agent is obliged, permitted, prohibited, empowered etc. to do, while it participates in the system. In such environments agents need to resort to assumptions, in order to establish what actions are appropriate to perform, and they need to do so dynamically, since the environment, the agents that exist in it, the information that is exchanged between them, and the normative relations between them change over time.

The rest of the chapter is organized as follows: section 6.2 provides a preliminary discussion, presents, in brief, a first approach on assumption-based reasoning and discusses its limitation with respect to a computational implementation; section 6.3 proposes a second and alternative way to reason hypothetically, which is appropriate for the implementation of a computational tool; section 6.4 presents a prototype as proof of concept for this technique and for experimentation; section 6.5 presents an example that illustrates this technique, and, finally, section 6.6 provides a summary.

6.2 Preliminaries

Initially, we represent norms as sentences of the form (1). This representation employs the predicates of the temporal logic augmented with some special predicates to denote normative relations (obligation, prohibition, permission, power). We view normative relations as properties that are initiated or terminated by the occurrence of agents' actions or events. The norms that can be expressed in such a representation take the form, for example "agent Agent2 is obliged/permitted/prohibited towards agent Agent1 to perform action Action2 by time Time2, if agent Agent1 performs action Action1, at time Time1".

The initial representation of an e-contract may be characterized as a triple (H, R, A). H corresponds to historical information and is a possibly empty or incomplete set of domain-dependent definitions for currently available information, i.e. H is a set of propositional or predicate formulae representing events that have occurred and facts that holds. R corresponds to domain-dependent causal information and is a possibly empty or incomplete set of sentences of the form (1). A is a non-empty set of sentences of the form (1) expressing the domain–independent knowledge.

In [90], we proposed the construction of a DfT, by mapping a sentence of the form (1) to *any one* of the following defaults:

$X_1 \wedge X_2 \wedge \ldots \wedge X_k : true / Y$ (justification-free default rule)

$X_1 \wedge X_2 \wedge \ldots \wedge X_k : Y / Y$ (normal default rule)

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-1} : X_k / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_k : X_{k-1} / Y$

...

$X_2 \wedge \ldots \wedge X_k : X_1 / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-2} : X_{k-1}, X_k / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-1} : X_{k-2}, X_k / Y$

…

$X_2 \wedge \ldots \wedge X_{k-1} : X_1, X_k / Y$

…

$true : X_1, X_2, \ldots, X_{k-2}, X_{k-1}, X_k / Y$ (prerequisite-free default rule)

That is, each sentence of the initial norm representation, which involves k conditions, corresponds to any one of 2^k+1 defaults. The question that arises for the agent constructing the DfT is, which one of these 2^k+1 defaults should be chosen and employed in the inference procedure. This is tantamount to seeking to establish what assumptions are appropriate in order to fill in information gaps.

A first answer to this question and at the same time a proposal for the formal characterization of the DfT construction, relative to the currently available knowledge $H \cup R$, were presented in [95]. An e-contract can be represented as a DfT $\equiv (W, D)$ by translating/reformulating its initial representation. The construction of W and D sets is carried out as follows:

- The currently available knowledge W is constructed from the domain-specific part of the initial contract representation. Specifically, the W part of the DfT, is a copy of H, the possibly empty or incomplete historical information of the initial contract representation which contains all currently available knowledge about what holds and what happened.
- The set of defaults D of the DfT is constructed from the domain-independent definitions of the initial representation and domain-dependent definitions for causal relations. Specifically, D is constructed from sets A and R which contain sentences of the form (1) as follows: The conclusion of each such sentence is mapped to the consequent part of each default, while its conditions may be mapped to the prerequisite or the justification part of each default, depending on what information is defined in the initial knowledge base $H \cup R$: conditions that can be derived from $H \cup R$ are mapped to the prerequisite, while conditions that cannot be derived from $H \cup R$ are candidates for assumptions, and are mapped to the justifications. That is, each initial axiom of the form (1) does not correspond uniquely to a default. Although this may seem unsettling, it affords an agent flexibility in the construction of the DfT, as it can identify the set of candidate assumptions for its reasoning, dynamically, depending on the knowledge it possesses.

As a result, an e-contract (and normative systems in general) may be characterized formally as the pair (W, D), where $W=H$ and D contains, for each definition $(Y \leftarrow X_1 \wedge \ldots \wedge X_k) \in A \cup R$, (possibly semi-grounded) defaults of the form $P_1 \wedge \ldots \wedge P_m : J_1, J_2, \ldots J_n \,/\, C$, such that $m+n=k$ and $P_i = SUBST(\theta, X_j)$ if $H \cup R \vdash SUBST(\theta, X_j)$, $Ji = SUBST(\theta, X_j)$ if $H \cup R \nvdash SUBST(\theta, X_j)$, and finally $C = SUBST(\theta, Y)$.

This formal characterization is not amenable to computational implementation, since the agent that constructs the DfT must attempt to prove literals from its knowledge base, in order to decide whether to use them in the prerequisite or the justification part of each default. In other words, the agent needs to attempt to prove literals (and fail in doing so) in order to determine which of these are candidate assumptions.

In order to overcome this limitation we describe, in the next section, an alternative procedure by which an agent may determine assumptions, and consequently construct the DfT, dynamically. This technique does not require the agent to prove literals from its current knowledge base, and therefore, it is suitable for implementation.

6.3 Theory Construction and Inference

6.3.1 Rule mapping

We may think of the 2^k possible defaults for a single norm of the form (1) as representations of the possible mental states in which the agent may find itself[14]. Each such state is characterized by what is known and what is not known to the agent, i.e. it represents what we may call the single-norm *knowledge/hypothesis (KH) status* of the agent. These possible states are organized in a multi-level hierarchy, which we depict as a triangle, such as the one shown in Figure 6.1. The top of the triangle denotes the direction in which the agent’s mental state evolves over time. Each level of the *KH structure* contains one or more of the 2^k defaults, depending on the number of assumptions that these defaults employ. Level 0 contains the single assumption-free default, level 1 contains the k one-assumption defaults, and so on, until the top level which contains the single, knowledge-free default. That is, for a an agent which possesses an initial rule of the form (1), moving upwards in a stepwise manner until it reaches the top level of the single-norm KH structure, is tantamount to identifying possible assumptions among the conditions that are included in the initial rule. Defaults contained in the same level have the same number of assumptions; the defaults of any given level contain one more assumption than the defaults of the immediately lower level, and one fewer assumption than the defaults of the immediately higher level. Let $|L|$ denote the total number of defaults contained at

---

[14] For the moment we omit the normal default rule. We discuss normal defaults separately in subsection 6.3.3.

level L, where $0 \leq L \leq k$, and k is the total number of conditions in an initial rule of the form (1). Then, it is easy to verify that the following properties hold:

- $|L| = 1$ if $L = 0$
- $|L| = (k - L + 1) * |L-1| / L$ if $L \neq 0$

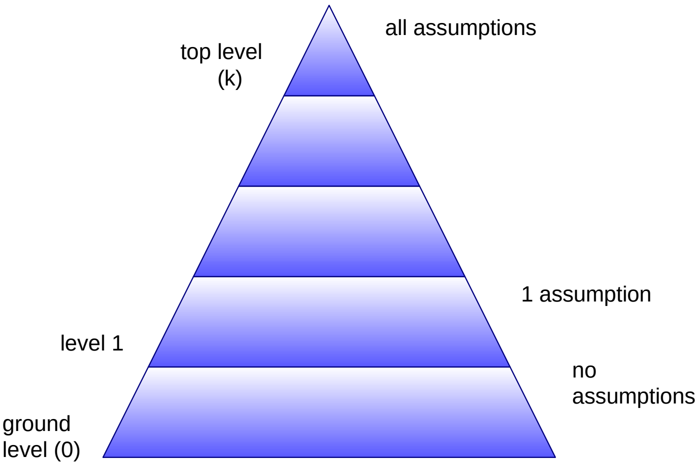


Figure 6.1 Single-norm KH structure of an agent's mental states

To illustrate this idea consider the following rule, which involves four conditions ($k=4$):

$Y \leftarrow X_1 \wedge X_2 \wedge X_3 \wedge X_4$

The corresponding 5-level triangle is[15]:

Level 0: $\{ X_1,X_2,X_3,X_4 : true / Y \}$

Level 1: $\{ X_1,X_2,X_3 : X_4 / Y,$
$X_1,X_2,X_4 : X_3 / Y,$
$X_1,X_3,X_4 : X_2 / Y,$
$X_2,X_3,X_4 : X_1 / Y \}$

Level 2: $\{ X_1,X_2 : X_4, X_3 / Y,$
$X_1,X_3 : X_4, X_2 / Y,$
$X_2,X_3 : X_4, X_1 / Y,$
$X_1,X_4 : X_3, X_2 / Y,$
$X_2,X_4 : X_3, X_1 / Y,$
$X_3,X_4 : X_2, X_1 / Y \}$

Level 3: $\{ X_1 : X_4, X_3, X_2 / Y,$
$X_2 : X_4, X_3, X_1 / Y,$

---

[15] Note that, in sequent calculus, comma separated prerequisites are interpreted conjunctively.

$X_3 : X_4, X_2, X_1 / Y$,

$X_4 : X_3, X_2, X_1 / Y$ }

Level 4: { true : $X_4,X_3,X_2,X_1 / Y$ }

Table 6.1 Algorithm 2: Computation of the single-norm KH structure

Algorithm 2

```
PROCEDURE DefaultsForANorm(R)

VARIABLES
    INTEGER: i, j, k, d
    LIST: R   // list representing the initial rule R, where the first element is the conclusion
                  and the rest elements are the conditions
        PS   // list of lists representing the power set
        TempL   // list of integers representing a binary number

START

    k=Size(R)-1;   // Size(LIST R) returns the number of elements in a list R,
                        i.e. k is the number of conditions in a rule

    PS ← PowerSet(k) // PowerSet(INTEGER k) returns the integers from 0 up to $2^k$-1 as a list of $2^k$
                          elements in binary format, e.g. returns 0 as [0,0,0,0,…,0], 1 as [0,0,0,0,…,1] etc
    d=2^k   // representing the number of elements contained in PS

    FOR i=0 TO i=d-1 STEP 1 DO {

        Element GetMember(R,0) is the Consequent // GetMember(LIST R,INTEGER x) returns the
                                                     element of a list R at the specified position x
        TempL ← PS[i]
        FOR j=0 TO j=k-1 STEP 1 DO {

            IF ( TempL[j] = = 0) THEN {
                Element GetMember(R, j+1) is a Prerequisite
            } ELSE IF ( TempL[j] = = 1) THEN {
                Element GetMember(R, j+1) is a  Justification
            }   // end of IF

        }   // end of FOR

    }   //end of FOR

END_ PROCEDURE
```

Of course, contracts (and normative systems in general) include multiple norms, for each of which a structure, such as the one described above may be constructed. The construction is done according to Algorithm 2 (Table 6.1). This algorithm needs to identify all possible combinations of knowledge and assumptions for the construction of the KH structure for a single norm. This is tantamount to computing the power set $\mathcal{P}(S)$ of a given set S. For instance, given a set S = {a, b, c, d} of n=4 elements, the corresponding power set is the set $\mathcal{P}(S)$ that contains 2^n elements as follows:

$\mathcal{P}(S)$ = { {}, {a}, {b}, {c}, {d}, {a,b}, {a,c}, {a,d}, {b,c}, {b,d}, {c,d}, {a,b,c}, {a,b,d}, {a,c,d}, {b,c,d}, {a,b,c,d} }

Note that the elements of $\mathscr{P}(S)$ may be considered as sorted lists of length n, i.e. of n positions, where 0 denotes the absence of the corresponding element of the set S at this position and 1 denotes the presence of the corresponding element of the set S at this position, i.e.:

$\mathscr{P}(S)$ = { {0,0,0,0}, {1,0,0,0}, {0,1,0,0}, {0,0,1,0}, {0,0,0,1}, {1,1,0,0}, {1,0,1,0}, {1,0,0,1}, {0,1,1,0}, {0,1,0,1}, {0,0,1,1}, {1,1,1,0}, {1,1,0,1}, {1,0,1,1}, {0,1,1,1}, {1,1,1,1} }

Also, note that, the elements of $\mathscr{P}(S)$ may be seen as binary integers of length n, i.e.:

$\mathscr{P}(S)$ = { {0000},{1000},{0100},{0010},{0001},{1100},{1010},{1001},{0110},{0101},{0011},{1110},{1101},{1011},{0111},{1111} }

or as their decimal equivalents:

$\mathscr{P}(S)$ = { 0, 8, 4, 2, 1, 12, 10, 9, 6, 5, 3, 14, 13, 11, 7, 15 }

This analysis shows that when an agent attempts to compute all the 2^k different arrangements of the k conditions of some initial norm, in order to decide whether to use them as knowledge or assumptions, it needs to compute the binary format of decimal integers from 0 up to 2^k-1 (Algorithm 2, line 15). Next, it needs to match the 0 indicator to denote knowledge and the 1 indicator to denote assumptions. It can then use this in order to populate the prerequisites (P) and justifications (J) parts of a default, as shown in Table 6.1 (Algorithm 2, lines 19 – 33).

6.3.2 Inference on the basis of Assumptions

Normative systems contain multiple norms, for each of which an agent constructs a KH structure. All the resulting single-norm KH structures are composed into a single polygon-like structure (Figure 6.2), which contains as many levels as the tallest of the constituent single-norm KH structures. Given an initial set of norms, the number of levels of the multi-norm KH structure is equal to the maximum k_i, where $1 \leq i \leq r$ and r is the number of the initial norms of the form (1). To be precise, we should note that the multi-norm KH structure does not have a single top, since each constituent single-norm KH structure may have its own top level. We are interested in the highest top level, since this denotes the point of termination of an agent's inference process, when an agent moves upwards in the multi-norm KH structure and its mental state evolves over time.

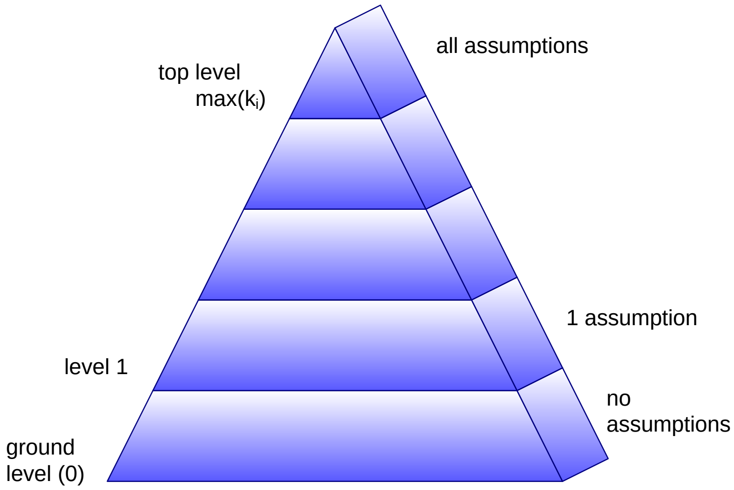


Figure 6.2 Multi-norm KH structure of an agent's mental states

Therefore, the full DfT that is constructed by an agent is a pair of the form (W, D), where W contains all of the available (if any) historical information and D is the multi-norm KH structure. Level 0 contains the r assumption-free defaults, level 1 contains the $\sum_{i=1}^{r} k_i$ one-assumption defaults, and so on, until the top $\max(k_i)$ level, which contains some of the knowledge-free defaults.

Note that, although the corresponding rule mapping is *one-to-many*, only one default for each initial norm may finally be employed for inference. Specifically, the inference process starts from the ground level, by applying as many defaults as possible given the agent's current knowledge. Each time a default applies its consequent is included in the extension that is being computed currently. When there are no further defaults that can be applied in a level, this signals to the agent that assumptions are needed in order to proceed, and inference continues by examining defaults that lie in the next level upwards. Note that the case where reasoning is possible using only rules from the ground level is identical to inference in classical logic, but here we are also able to preserve consistency of entailment, by employing DfL or some of its variations such as Constrained Default Logic [225]. The inference is done according to Algorithm 3 (Table 6.2) and Algorithm 4 (Table 6.3). Algorithm 3, first, calls for the construction of all single-norm KH structures, each of which is done as per Algorithm 2, and, second, calls for the computation of extensions, i.e. Algorithm 4, in a stepwise manner. Given the knowledge/hypothesis status of the agent at any given point Algorithm 4 computes extensions by maintaining syntactically consistent sets of sentences.

Table 6.2 Algorithm 3: Inference procedure

Algorithm 3

```
PROCEDURE Hypotheser(W, RS)
VARIABLES
        LIST: W   // initial knowledge set
              RS   // list of lists representing the initial rule set
              R   // list representing the initial rule R, where the first element is the conclusion and
                     the rest elements are the conditions
              In, Out, $\Pi$   // lists representing the current In, Out and $\Pi$ sets
              DS   // list that contains defaults contained in a level
              $\Delta$   // list that contains indicators for rules that have not fired yet
        INTEGER: L //representing the current level

START

    // Compute all possible mappings for initial rules
    FOR EACH R MEMBER OF RS DO {
            DefaultsForANorm(R) // Procedure that computes all possible mappings
                                   for the initial rule R
    }   // end of FOR

    // Initialize the world
    In  $\leftarrow$ W
    Out  $\leftarrow \varnothing$
    $\Pi \leftarrow \varnothing$
    $\Delta \leftarrow$ indicators for all rules contained in RS
    L  $\leftarrow$ 0

    // Inference process
    WHILE ( $\Delta \neq \varnothing$  and No Inconsistencies Exist) DO {
          DS $\leftarrow$ GetDefautls(L, $\Delta$)   // GetDefault(INTEGER  L, LIST $\Delta$) queries the hierarchical
                                          structures and returns  the defaults that are contained
                                          in levels L  for rules that are members of the list $\Delta$
          Reasoner(In, Out, $\Pi$, DS, $\Delta$)   // procedure that computes extensions
          L  $\leftarrow$ L + 1 // Move to next level
    }   // end of WHILE

END_ PROCEDURE
```

This analysis indicates that during the reasoning process an agent infers all possible conclusions on the basis of its current knowledge. When no further inference is possible, the agent is able to reassess its mental state and establish its knowledge/hypothesis status, in order to continue. That is, the agent first attempts to draw conclusions using only assumption-free defaults, then by employing one assumption per default, then by employing two assumptions per default, and so on, until no further defaults apply. In other words, a general priority criterion among defaults is being established: This is the number of assumptions employed via the use of a default rule. Thus, such inference in a step-wise manner ensures that the agent employs the fewest possible hypotheses, always. This can be proved formally by mathematical induction. Here is a sketch of such a proof:

Table 6.3 Algorithm 4: Computation of extensions

Algorithm 4

```
PROCEDURE Reasoner(In, Out, Π, DS, Δ)
VARIABLES
        LIST: In, Out, Π   // lists representing the current In, Out and Π sets
                DS   // list that contains defaults contained in a level
                Δ   // list that contains indicators for rules that have not fired yet
                D   // list representing a default rule, where the first element is the conclusion and
                        the rest elements are the prerequisites and justifications
                PS, JS, CS   // lists that contains the default prerequisites, justifications and consequents
                P, J, C   // lists that represent a prerequisite, justification and consequent

START

          FOR EACH D MEMBER OF DS DO {

                    PS ← GetPrerequisites(D)   // GetPrerequisites(LIST D) returns a list that contains
                                                    the rule prerequisites
                    JS ← GetJustificatios(D)   // GetJustifications(LIST D) returns a list that contains
                                                    the rule justifications

                    // check the prerequisites
                    FOR EACH P MEMBER OF PS DO {
                              Condition 1: Check whether all members of PS are contained in the In list
                    }   // end of FOR

                    // check the justifications
                    FOR EACH J MEMBER OF JS DO {
                              Condition2: Check whether all members of JS are not contained in the Out list
                    }   // end of FOR

                    // the default rule fires
                    IF ( both of the above conditions hold ) THEN {
                              add GetConsequent(D) in the In list as a new member
                              // GetConsequent(LIST D )returns a list that contains the rule consequent
                              add the negation of all member of JS in the Out list as new members
                              add D in the Π list as a new member
                              remove the indicator of D form the Δ list
                    }   // end of IF

          }   // end of FOR

          RETURN  In, Out, Π, DS, Δ

END_ PROCEDURE
```

Let $J_i(L)$ be the total number of assumptions employed, when inference uses defaults from level L for initial norm i ($1 \leq i \leq r$) and $\hat{J}(n)$ be the total number of assumptions employed when inference uses defaults from level L, for all of the initial norms r, i.e.:

$$\hat{J}(L) = \sum_{i=1}^{r} J_i(L)$$

For $L=0$, i.e. when inference is made on the ground level, $\hat{J}(L)=0$. Assuming that for $L=m$, each rule at this level employs m assumptions, then $\hat{J}(m)=r*m$ satisfies the property (this is the worst case, i.e. all r defaults apply). Then for $L=m+1$ we must show that $\hat{J}(m+1) = \hat{J}(m) + \Delta$, where Δ is the number of defaults that apply and $1 \leq \Delta \leq r$. $\hat{J}(m)$ is the fewest possible assumptions given our induction hypothesis, and Δ is the fewest possible assumptions (one assumption for each rule) for the next step of the inference process, that is step $m+1$. In the worst case where all defaults apply $\Delta = r$.■

To illustrate the inference procedure, consider this next example: let us assume that a normative system comprises two rules of the form:

$R1 \equiv Y_1 \leftarrow X_1 \wedge X_2$, and $R2 \equiv Y_2 \leftarrow X_3 \wedge X_4 \wedge X_5$

Thus, the corresponding single-norm and multi-norm KH structures are as follows ($D_{level,number}$ denotes the level of the default and its identification number within its level, and it is used to facilitate reference):

Single-norm KH structure for $R1$:
Level 0: $\{ D1_{0,1} \equiv X_1,X_2 : true / Y_1 \}$

Level 1: $\{ D1_{1,1} \equiv X_1 : X_2 / Y_1,$
$D1_{1,2} \equiv X_2 : X_1 / Y_1 \}$

Level 2: $\{ D1_{2,1} \equiv true : X_2,X_1 / Y_1 \}$

Single-norm KH structure for $R2$:
Level 0: $\{ D2_{0,1} \equiv X_3,X_4,X_5 : true / Y_2 \}$

Level 1: $\{ D2_{1,1} \equiv X_3,X_4 : X_5 / Y_2,$
$D2_{1,2} \equiv X_3,X_5 : X_4 / Y_2,$
$D2_{1,3} \equiv X_4,X_5 : X_3 / Y_2 \}$

Level 2: $\{ D2_{2,1} \equiv X_3 : X_5, X_4 / Y_2,$
$D2_{2,2} \equiv X_4 : X_5, X_3 / Y_2,$
$D2_{2,3} \equiv X_5 : X_4, X_3 / Y_2 \}$

Level 3: $\{ D2_{3,1} \equiv true : X_5, X_4, X_3 / Y_2 \}$

Multi-norm KH structure for $R1$ and $R2$:
Level 0: $\{ D1_{0,1} \equiv X_1,X_2 : true / Y_1, \quad D2_{0,1} \equiv X_3,X_4,X_5 : true / Y_2 \}$

Level 1: $\{ D1_{1,1} \equiv X_1 : X_2 / Y_1, \quad D2_{1,1} \equiv X_3,X_4 : X_5 / Y_2,$
$D1_{1,2} \equiv X_2 : X_1 / Y_1, \quad D2_{1,2} \equiv X_3,X_5 : X_4 / Y_2,$

$D2_{1,3} \equiv X_4{,}X_5 : X_3 \,/\, Y_2$ }

Level 2: { $D1_{2,1} \equiv \text{true} : X_2{,}X_1 \,/\, Y_1$, $D2_{2,1} \equiv X_3 : X_5, X_4 \,/\, Y_2$,

$D2_{2,2} \equiv X_4 : X_5, X_3 \,/\, Y_2$,

$D2_{2,3} \equiv X_5 : X_4, X_3 \,/\, Y_2$ }

Level 3: { $D2_{3,1} \equiv \text{true} : X_5, X_4, X_3 \,/\, Y_2$ }

Here are some possible scenaria, with different initial knowledge available each time, in the beginning of the reasoning process:

- if $W=\{X_1, X_2\}$ then extension $In(2)= W \cup \{Y_1, Y_2\}$ is computed by making the assumption that X_5, X_4 and X_3 hold ($Out(2)=\{\neg X_5, \neg X_4, \neg X_3\}$) and by applying defaults $D1_{0,1}$ and $D2_{3,1}$ respectively, i.e. $\Pi(2)=\{ D1_{0,1}, D2_{3,1}\}$. Note that, the default $D1_{0,1}$ takes priority over the default $D2_{3,1}$, due to the fact that the first one does not employ any assumptions while the second one employs three assumptions in the inference process.
- if $W=\{X_1, X_2, X_3\}$ then extension $In(2)= W \cup \{Y_1, Y_2\}$ is computed by making the assumption that X_5 and X_4 hold ($Out(2)=\{\neg X_5, \neg X_4\}$) and by applying defaults $D1_{0,1}$ and $D2_{2,1}$ respectively, i.e. $\Pi(2)=\{ D1_{0,1}, D2_{2,1}\}$. Also, note that, the default $D1_{0,1}$ takes priority over the default $D2_{2,1}$.
- if $W=\{X_1, X_3, X_4, X_5\}$ then extension $In(2)= W \cup \{Y_2, Y_1\}$ is computed by making the assumption that only X_2 holds ($Out(2)=\{\neg X_2\}$) and by applying defaults $D2_{0,1}$ and $D1_{1,1}$ respectively, i.e. $\Pi(2)=\{ D2_{0,1}, D1_{1,1}\}$. The default $D2_{0,1}$ takes priority over the default $D1_{1,1}$, due to the fact that the first one does not employ any assumptions while the second one employs an assumption in the inference process.
- if $W=\{X_1, X_3, X_4\}$ then extension $In(2)= W \cup \{Y_1, Y_2\}$ is computed by making the assumptions that X_2 and X_5 hold ($Out(2)=\{\neg X_2, \neg X_5\}$) and by applying defaults $D1_{1,1}$ and $D2_{1,1}$ respectively, i.e. $\Pi(2)=\{ D1_{1,1}, D2_{1,1}\}$. Now, note that, defaults $D1_{1,1}$ and $D2_{1,1}$, employ the same number of assumptions in the inferences process. Due to this fact and according to the priority criterion on the basis of the total number of assumptions employed by a rule, none of the rules takes priority over the other. Thus, both process $\Pi(2)=\{ D1_{1,1}, D2_{1,1}\}$ or $\Pi(2)=\{ D2_{1,1}, D1_{1,1}\}$ are feasible. It just happens in this case that, processes have identical final impacts to the environment, i.e. $In(2)= W \cup \{Y_1, Y_2\}$ and $Out(2)=\{\neg X_2, \neg X_5\}$ or $In(2)= W \cup \{Y_2, Y_1\}$ and $Out(2)=\{\neg X_5, \neg X_2\}$.

This last example indicates the need for additional priority criteria. For instance, we may use as a criterion the size of factual knowledge a rule employs, i.e. the number of prerequisites. In this case the default $D2_{1,1}$ takes priority over the default $D1_{1,1}$, due to the fact that the first one fires on a larger factual basis in contrast to the second one, although both of them employ the same number of assumptions in the inference process.

Note that although a level may contain two or more defaults that correspond to the same initial contract rule (e.g. $D2_{1,1}$ or $D2_{1,2}$ or $D2_{1,3}$) there is no need for some kind of prioritization among those defaults. If two or more defaults of the same level, which are derived from the same initial rule (i.e. they belong to the same level within the same single-norm KH structure), were to apply simultaneously, then the more general default contained in the immediately lower level should have applied.

We should note that it is important to consider the issue of consistency between assumptions employed during the reasoning process and new inferences derived as a result of the reasoning process. One of the reasons for which we revised our initial proposal for the construction of the DfT, which was described in section 6.2, is precisely because an agent would require a revision mechanism in order to reconstruct the default rules as new information becomes available, and the agent is able to prove literals from its updated knowledge, and hence treat them as prerequisites rather than justifications. The alternative way for the construction of the DfT does not require any revision of the defaults. This is because inference involves one level at a time in a stepwise manner, and the agent moves upwards to the next level of the multi-norm KH structure only when it has exhausted inference at a given level. This ensures that the agent employs the fewest possible hypotheses.

6.3.3 Normal and Semi-Normal Default Theories

So far, we have omitted normal defaults from the discussion about the way in which an agent may construct its DfT. Normal defaults have the form $P{:}C/C$, i.e., their justification coincides with their consequent. Two questions seem to arise naturally:

- Should the agent include normal defaults in the KH structures that it constructs, and, if so,
- In which level of the KH structure should normal defaults be placed?

Normal default theories always possess an extension. Thus, it is practical to require the use of such defaults in order to ensure that the agent can compute at least one extension of its currently available knowledge, by adding to it new information, provided that consistency is preserved. That is, the normal default may be viewed as behaving similarly to the justification-free default, in that all its prerequisites should be satisfied by the current knowledge base; the only additional assumption made in the case of the normal default concerns the consistency of its consequent with the current knowledge base. For this reason, although the normal default contains a single assumption, and should therefore belong to level 1 of the KH structure, ‘operationally’ it belongs to level 0, since its assumption is not genuinely about something that holds in the world.

Hence, it seems to us that ‘operationally’ an agent may either omit normal defaults totally from the KH structures that it constructs, or it may include them in level 0, instead of the assumption-free default shown above, when it is important to ensure that the agent will compute at least one extension, while preserving consistency. In this case, $J_0 = \{Y\}$, and the defaults of the higher levels of the structures will be semi-normal, i.e. of the form $P{:}J \wedge C/C$, i.e., all of the justifications imply the consequent. In this case, the agent will verify the consistency of a future world before it actually proceeds with inference, i.e. it will be more cautious.

To illustrate this, recall the earlier example with the two rules of the form:

$R1 \equiv Y_1 \leftarrow X_1 \wedge X_2$, and $R2 \equiv Y_2 \leftarrow X_3 \wedge X_4 \wedge X_5$

where I is an additional set that contains pairs of formulae that render the knowledge base inconsistent when both members of the pairs hold simultaneously. The corresponding KH structures are as follows:

Single-norm KH structure for R1:

Level 0: $\{ D1_{0,1} \equiv X_1,X_2 : Y_1 / Y_1 \}$

Level 1: $\{ D1_{1,1} \equiv X_1 : Y_1, X_2 / Y_1,$

$D1_{1,2} \equiv X_2 : Y_1, X_1 / Y_1 \}$

Level 2: $\{ D1_{2,1} \equiv true : Y_1, X_2,X_1 / Y_1 \}$

Single-norm KH structure for R2:

Level 0: $\{ D2_{0,1} \equiv X_3,X_4,X_5 : Y_2 / Y_2 \}$

Level 1: $\{ D2_{1,1} \equiv X_3,X_4 : Y_2, X_5 / Y_2,$

$D2_{1,2} \equiv X_3,X_5 : Y_2, X_4 / Y_2,$

$D2_{1,3} \equiv X_4,X_5 : Y_2, X_3 / Y_2 \quad \}$

Level 2: $\{ D2_{2,1} \equiv X_3 : Y_2, X_5, X_4 / Y_2,$

$D2_{2,2} \equiv X_4 : Y_2, X_5, X_3 / Y_2,$

$D2_{2,3} \equiv X_5 : Y_2, X_4, X_3 / Y_2 \quad \}$

Level 3: $\{ D2_{3,1} \equiv true : Y_2, X_5, X_4, X_3 / Y_2 \}$

<u>Multi-norm KH structure for R1 and R2:</u>

Level 0: $\{ D1_{0,1} \equiv X_1,X_2 : Y_1 / Y_1, \quad D2_{0,1} \equiv X_3,X_4,X_5 : Y_2 / Y_2 \quad \}$

Level 1: $\{ D1_{1,1} \equiv X_1 : Y_1, X_2 / Y_1, \quad D2_{1,1} \equiv X_3,X_4 : Y_2, X_5 / Y_2,$

$D1_{1,2} \equiv X_2 : Y_1, X_1 / Y_1, \quad D2_{1,2} \equiv X_3,X_5 : Y_2, X_4 / Y_2,$

$D2_{1,3} \equiv X_4,X_5 : Y_2, X_3 / Y_2 \quad \}$

Level 2: $\{ D1_{2,1} \equiv true : Y_1, X_2,X_1 / Y_1, \quad D2_{2,1} \equiv X_3 : Y_2, X_5, X_4 / Y_2,$

$D2_{2,2} \equiv X_4 : Y_2, X_5, X_3 / Y_2,$

$D2_{2,3} \equiv X_5 : Y_2, X_4, X_3 / Y_2 \quad \}$

Level 3: $\{ \quad D2_{3,1} \equiv true : Y_2, X_5, X_4, X_3 / Y_2 \quad \}$

Here are some possible scenaria, for the cases discussed previously, but now with various sets of inconsistencies:

- if $W=\{X_1, X_2\}$ and $I=\{(Y_1, Y_2)\}$ then the extension computed previously $In(2)= W \cup \{Y_1, Y_2\}$ is not feasible due to the inconsistency that holds between Y_1 and Y_2. Instead, a new extension is computed as follows: $In(1)= W \cup \{Y_1\}$, $Out(1)=\{\neg Y_1\}$ by applying only the default $D1_{0,1}$, i.e. $\Pi(1)=\{ D1_{0,1}\}$.
- if $W=\{X_1, X_2, X_3\}$ and $I=\{(Y_1, Y_2)\}$ then the previously computed extension $In(2)= W \cup \{Y_1, Y_2\}$ is not feasible due to the inconsistency that holds between Y_1 and Y_2. Instead, a new extension is computed as follows: $In(1)= W \cup \{Y_1\}$, $Out(1)=\{\neg Y_1\}$ by applying only the default $D1_{0,1}$, i.e. $\Pi(1)=\{ D1_{0,1}\}$.
- if $W=\{X_1, X_3, X_4, X_5\}$ and $I=\{(Y_1, Y_2)\}$ then the previously computed extension $In(2)= W \cup \{Y_2, Y_1\}$ is not feasible due to the inconsistency that holds between Y_1 and Y_2. Instead, a new extension is computed as follows: $In(1)= W \cup \{Y_2\}$, $Out(1)=\{\neg Y_2\}$ by applying only the default $D2_{0,1}$, i.e. $\Pi(1)=\{ D2_{0,1}\}$.
- if $W=\{X_1, X_3, X_4\}$ and $I=\{(Y_1, Y_2)\}$ then the previously computed extension $In(2)= W \cup \{Y_1, Y_2\}$ is not feasible due to the inconsistency that holds between Y_1 and Y_2. Instead, two new extensions are computed as follows: $In(1)= W \cup \{Y_1\}$, $Out(1)=\{\neg Y_1,\neg X_2\}$ by

applying only the default $D1_{1,1}$, i.e. $\Pi(1)=\{D1_{1,1}\}$ and $In(1)= W \cup \{Y_2\}$, $Out(1)=\{\neg Y_2,\neg X_5\}$ by applying only the default $D2_{1,1}$, i.e. $\Pi(1)=\{D2_{1,1}\}$.

6.4 Prototype Implementation

We found it useful to implement a prototype as proof of concept for our technique and for experimentation. The prototype consists of three distinct components, shown in Figure 6.3:

- the Rule Constructor,
- the Rule Query, and
- the Inference Engine.

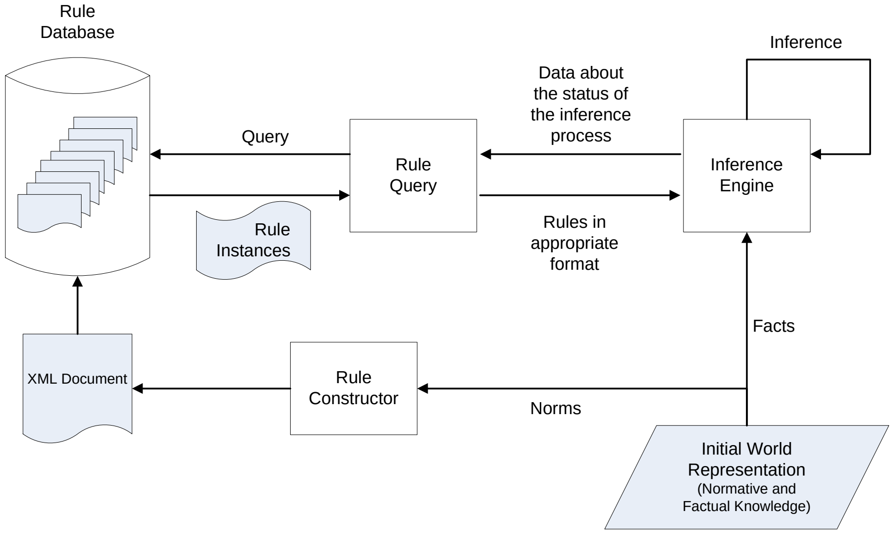


Figure 6.3 Prototype architecture

The Rule Constructor accepts a set of norms in the form of sequent calculus and constructs for each one the corresponding single-norm KH structure (Algorithm 2). It is implemented in Java. We choose to store these structures as XML documents for various reasons: First, because this offers the advantage of effortless storing and sharing of rules among the distinct components of the prototype. Moreover, it renders feasible the transport of rules to different platforms (e.g. different inference engines such as Prolog, Jess, Mandarax etc) or even to different applications (e.g. business applications, distributed systems applications, services composition applications etc).

To illustrate this idea consider the following rule, which involves three conditions ($k=3$):

$Y \leftarrow X_1 \wedge X_2 \wedge X_3$

Table 6.4 XML document that represents the single-norm KH structure

XML Document

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<defaults>
    <default level=0 number=1>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=prerequisite>X1</element>
          <element index=2 type=prerequisite>X2</element>
          <element index=3 type=prerequisite>X3</element>
    </default>
    <default level=1 number =1>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=justification>X1</element>
          <element index=2 type=prerequisite>X2</element>
          <element index=3 type=prerequisite>X3</element>
    </default>
    <default level=1 number =2>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=prerequisite>X1</element>
          <element index=2 type=justification>X2</element>
          <element index=3 type=prerequisite>X3</element>
    </default>
    <default level=1 number =3>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=prerequisite>X1</element>
          <element index=2 type=prerequisite>X2</element>
          <element index=3 type=justification>X3</element>
    </default>
    <default level=2 number =1>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=prerequisite>X1</element>
          <element index=2 type=justification>X2</element>
          <element index=3 type=justification>X3</element>
    </default>
    <default level=2 number =2>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=justification>X1</element>
          <element index=2 type=prerequisite>X2</element>
          <element index=3 type=justification>X3</element>
    </default>
    <default level=2 number =3>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=justification>X1</element>
          <element index=2 type=justification>X2</element>
          <element index=3 type=prerequisite>X3</element>
    </default>
    <default level=3 number =1>
          <element index=0 type=consequent>Y</element>
          <element index=1 type=justification>X1</element>
          <element index=2 type=justification>X2</element>
          <element index=3 type=justification>X3</element>
    </default>
</defaults>
```

The corresponding XML document is shown in Table 6.4 (the total number of levels is 4, the total number of default rules is 8, the number of defaults at level 0 (ground level) is 1, the number of defaults at level 1 is 3, the number of defaults at level 2 is 3, and finally, the number of defaults at level 3 (top level) is 1). All corresponding XML documents are stored as plain text files in the Rule Database whose role is identical to the role of the multi-norm KH structure.

The Inference Engine is implemented in Prolog (Algorithms 3 and 4). During reasoning, the Inference Engine communicates with the Rule Query in order to obtain new rules and continue inference on a hypothetical basis. For the implementation of the Rule Query we used technologies such as DOM, XSLT and XQuery in order to extract data from the XML documents that are stored in the Rule Database, i.e. procedure *GetDefaults* (Algorithm 3, line 28).

The prototype that we developed follows the specifications listed below:

- Norms are initially represented in propositional logic as sequent calculus sentences of the form (1), and the tool constructs propositional DfTs. This is clearly an aspect that we wish to review for the next version of the prototype.
- Extensions are computed in the manner presented in [14], i.e., by maintaining syntactically consistent sets of formulae whose conditions part (prerequisites and justifications) is interpreted conjunctively as in sequent calculus. The computation of extensions in Prolog is based on ideas presented in [12]. We extended these ideas in our implementation, in order to address more general cases, e.g. to support rule schemata with multiple prerequisites and justifications.
- The agent constructs the KH structures before it starts its inference process, and it stores the KH structures that it produces as XML documents.
- During the inference process, the agent processes the multi-norm KH structure that it produced earlier. It processes it starting from the ground level and moving upwards, towards the top of the structure. Within each level of the multi-norm KH structure, the agent encounters those defaults of single-norm KH structures that lie at the same level within the corresponding single-norm KH structures. That is, when the agent processes level κ of the multi-norm structure, all defaults that lie at level κ of the constituent single-norm KH structures are available to it. It examines defaults at this level, in the order it which it produced them, during the construction of the constituent single-

norm KH structures. Of course this is amenable to change, if we so wish, by defining other criteria for prioritization among defaults, as we noted earlier in subsections 6.3.1 and 6.3.2.

6.5 Example

Consider a 3-party business transaction that takes place in an electronic marketplace populated by software agents. A buyer agent (BA) communicates with a seller agent (SA) and establishes an agreement for purchasing a certain product. Consequently, the seller agent communicates with a carrier agent (CA) and establishes a separate agreement for the safe and timely delivery of goods to the buyer agent. An extract of the initial set of contract norms for the agreement between the buyer agent and the seller agents is as follows:

R={ R1 ≡ SAIsObligedToDeliverToBAWithinNext20days ← BAOrdersFromSA ∧ E-shopFunctionsWell,

R2 ≡ BAIsObligedToPayCAOnBehalfOfSA ← BAOrdersFromSA ∧ CADeliversToBA
∧ CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA }

Note that these norms have the same number of conditions as the norms considered in the abstract example presented previously. Thus, the corresponding KH structures are as follows:

<u>Single-norm KH structure for R1:</u>
Level 0: {
$D1_{0,1} \equiv$
BAOrdersFromSA, E-shopFunctionsWell
: true
/ SAIsObligedToDeliverToBAWithinNext20days }

Level 1: {
$D1_{1,1} \equiv$
BAOrdersFromSA
: E-shopFunctionsWell
/ SAIsObligedToDeliverToBAWithinNext20days,

$D1_{1,2} \equiv$
E-shopFunctionsWell
: BAOrdersFromSA
/ SAIsObligedToDeliverToBAWithinNext20days }

Level 2: {
$D1_{2,1} \equiv$
true
: E-shopFunctionsWell, BAOrdersFromSA
/ SAIsObligedToDeliverToBAWithinNext20days }

<u>Single-norm KH structure for R2:</u>
Level 0: {

$D2_{0,1} \equiv$
BAOrdersFromSA, CADeliversToBA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: true
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 1: {
$D2_{1,1} \equiv$
BAOrdersFromSA, CADeliversToBA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{1,2} \equiv$
BAOrdersFromSA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: CADeliversToBA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{1,3} \equiv$
CADeliversToBA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 2: {
$D2_{2,1} \equiv$
BAOrdersFromSA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, CADeliversToBA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{2,2} \equiv$
CADeliversToBA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{2,3} \equiv$
CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: CADeliversToBA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 3: {
$D2_{3,1} \equiv$
true
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, CADeliversToBA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Multi-norm KH structure for R1 and R2:

Level 0: {
$D1_{0,1} \equiv$
BAOrdersFromSA, E-shopFunctionsWell
: true
/ SAIsObligedToDeliverToBAWithinNext20days,

$D2_{0,1} \equiv$
BAOrdersFromSA, CADeliversToBA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: true
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 1: {
$D1_{1,1} \equiv$
BAOrdersFromSA
: E-shopFunctionsWell
/ SAIsObligedToDeliverToBAWithinNext20days,

$D1_{1,2} \equiv$
E-shopFunctionsWell
: BAOrdersFromSA
/ SAIsObligedToDeliverToBAWithinNext20days

$D2_{1,1} \equiv$
BAOrdersFromSA, CADeliversToBA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{1,2} \equiv$
BAOrdersFromSA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: CADeliversToBA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{1,3} \equiv$
CADeliversToBA, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 2: {
$D1_{2,1} \equiv$
true
: E-shopFunctionsWell, BAOrdersFromSA
/ SAIsObligedToDeliverToBAWithinNext20days,

$D2_{2,1} \equiv$
BAOrdersFromSA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, CADeliversToBA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{2,2} \equiv$
CADeliversToBA
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA,

$D2_{2,3} \equiv$
CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA
: CADeliversToBA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Level 3: {
$D2_{3,1} \equiv$
true
: CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA, CADeliversToBA, BAOrdersFromSA
/ BAIsObligedToPayCAOnBehalfOfSA }

Suppose that the current explicit knowledge that the buyer agent possesses is that it has ordered goods from the seller agent, that the e-shop functions properly, and that the carrier agent that will actually deliver the goods is legally empowered to accept payment on behalf of the seller agent, i.e., the buyer agent’s current knowledge is:

W={ BAOrdersFromSA, E-shopFunctionsWell, CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA }

On the basis of this knowledge alone, the buyer may only infer, that the seller is obliged to deliver products to it, within the next 20 days, i.e. the extension $In(1)= W \cup \{$

SAIsObligedToDeliverToBAWithinNext20days } is computed by making no assumptions (Out(1)={ }) and by applying default $D1_{0,1}$, i.e. Π(1)={ $D1_{0,1}$ }.

But, apart from establishing what it must expect from its counterparty, the buyer agent may wish to explore potential future scenaria. For instance, the buyer may need to perform *best-guess reasoning* and plan its future activities on the assumption that certain events/actions will occur, and that its partners' actions will be valid. Suppose that the buyer wants to infer the time by which it will have to pay for the goods, assuming that all goes well and it receives them in good time, because it wants to plan to have adequate funds available. To derive such an answer the buyer agent needs to identify and employ the assumption that delivery happens in due time (CADeliversToBA)[16], i.e. the extension In(2)= W ∪ { SAIsObligedToDeliverToBAWithinNext20days, BAIsObligedToPayCAOnBehalfOfSA } is computed by making the assumption that CADeliversToBA holds (Out(2)={ ¬CADeliversToBA }) and by applying defaults $D1_{0,1}$ and $D2_{1,2}$ (Π(2)={ $D1_{0,1}$, $D2_{1,2}$ }), respectively.

Now suppose that the buyer agent does not possess complete historical information, i.e. it does not know everything that may have happened so far. Let its current knowledge be such that it only knows that it ordered goods from the seller agent, that the e-shop functions well, and that the carrier agent delivered goods to it:

W={ BAOrdersFromSA, E-shopFunctionsWell, CADeliversToBA }

The buyer may need to perform *no-risk reasoning*, in order to derive a conclusion based on assumptions, because alternatively it might find itself in an undesirable situation. For instance, it may want to infer that it has an obligation to pay for the goods that it received, yet this inference is not possible, unless it assumes that the carrier agent is legally empowered to accept payment on behalf of the seller agent (CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA), i.e. the extension In(2)= W ∪ { SAIsObligedToDeliverToBAWithinNext20days, BAIsObligedToPayCAOnBehalfOfSA } is computed by making the assumption that CAIsEmpoweredToAcceptPaymentFromBAOnBehalfOfSA holds (Out(2)={ ¬CAIsEmpoweredToAcceptPaymenFromBAOnBehalfOfs }) and by applying defaults $D1_{0,1}$ and $D2_{1,1}$ (Π(2)={ $D1_{0,1}$, $D2_{1,1}$ }), respectively. In this scenario, the buyer agent does not possess knowledge about the carrier agent's legal power to accept payment on behalf of the seller agent. It may be the case that when such information was communicated to it by the seller

---

[16] In the full representation of the example, using some temporal logic, the temporal conditions involved in norms, are treated as all other conditions, when the agent constructs single-norm KH structures, i.e. the agent can make assumptions about them as well.

agent, it got lost or distorted, or it may be the case that the seller agent simply ‘forgot’ to communicate such information to it. If the buyer agent does not perform *no-risk reasoning*, it risks finding itself in a situation where it will have violated its obligation to pay for the goods that it received, inadvertently, and it will have to face the legal consequences, e.g. to pay extra charges.

6.6 Summary

The work presented in this chapter is motivated by the need for assumption-based reasoning in open normative multi-agent environments. The behaviour of agents in multi-agent environments is restricted by the norms that regulate the particular environment in which they participate. In the most general case, regardless of any particular application domain, some agreements govern the society of agents. Unavoidably in open environments agents have incomplete knowledge about their world, and about other agents, yet they must somehow plan their activities. The question we seek to address is whether it is possible for agents to identify appropriate assumptions dynamically. We argued that e-contracts could be represented as DfT and proposed a theoretical way in which such theories could be constructed automatically from initial representations. That proposal relied on determining what information could be proved from the agent’s knowledge base, in order to decide whether it would serve as an assumption or not. Later, we proposed an incremental technique that can be used for this construction which enables the dynamic and ad hoc identification of candidate assumptions without resorting to proof. We have developed a prototype implementation based on this idea, which translates initial propositional representations into propositional DfT. Norms are represented as sentences of the form (1) and extensions are computed in the manner discussed in chapter 5.

Research discussed in this chapter has been published or is under review in [92, 95, 97, 85, 89].

7 Autonomous Hypothetical Non-monotonic Reasoning

7.1 Introduction

In the previous chapter, we proposed two different approaches to assumption-based reasoning that enable agents to *reformulate* an initial set of norms by identifying and employing appropriate candidate assumptions dynamically. The first approach fall short due to the agent need to attempt to prove formulae (and fail in doing so) in order to decide which of these are candidate assumptions. This is obviously a strong limitation towards a computational implementation. Concerning the second approach, although the implementation is feasible, and indeed we presented a prototype tool, we see, now, that additional requirements are essential towards *rational autonomous* CSR and specifically, autonomous hypothetical non-monotonic reasoning.

In this chapter, furthermore, we address the issue of agent autonomy. We take the most recent perspective on autonomy that is relevant to an agent’s reasoning process. Our agent is expected to make inferences about which beliefs to adopt about its environment, other agents and norms in force, which goals to commit to, and which actions to perform, in the presence of incomplete or inconsistent information, and it is expected to be independent from external intervention in this reasoning process. We see that the degree to which an agent's reasoning is autonomous is affected by the degree to which it is able to choose its assumptions autonomously. We claim that an agent that answers the reasoning problem 1H - 3H, addresses also the autonomy problem:

1A. What is the appropriate behaviour, i.e. physical and/or mental actions, in order to be autonomous?

2A. How does my independence at some time point explain my current state and how does it affects my inferences in the future? and

3A. What happens when I need to adapt my independence (reduce or increase it) because of changes in the environment?

We re-introduce the incremental technique discussed in the previous chapter in a manner that enables agents to ‘develop for themselves the laws and strategies according to which they regulate their behaviour (in the spirit of [234]) and to ‘make their own inferences and reasoning and to rely on their own conclusions’ (in the spirit of [40]). It turns out that the KH structure is, in fact, a *lattice*. The lattice represents what we may call the *KH space* of the agent and each lattice node, i.e. a default rule, is characterized by what is known and what is not known to the agent, i.e. it represents what we called the *KH status* of the agent. At any particular time point the agent may position itself on it, given the explicit knowledge that it currently possesses, i.e. without resorting to proof. Once the agent has positioned itself on this lattice, it finds out what assumptions are related to the node it occupies and may employ them in its reasoning. As the agent’s knowledge changes over time, and consequently as its assumption needs change, the agent re-positions itself on the lattice by moving on it from node to node. The lattice structure that we use in order to represent *KH spaces*, first, suggests that an implementation is also feasible, relying only on set manipulation rather than proof, and, second, facilitates hypothetical nonmonotonic reasoning.

The rest of the chapter is organized as follows: sections 7.2 to 7.4 re-introduce the incremental technique discussed in the previous chapter, specifically, section 7.2 concerns with the dynamic identification of assumptions, section 7.3 concerns with theory construction, and section 7.4 concerns with the inference proccess; section 7.5 presents ways to manage the space of knowledge/hypothesis towards autonomous and rational reasoning; section 7.6 illustrates these proposals by discussing hypothetical nonmonotonic reasoning in various examples; and, finally, section 7.7 provides a summary.

7.2 Assumption Identification

Recall, that a norm of the form (1) may be mapped to *any one* of the following defaults:

$X_1 \wedge X_2 \wedge \ldots \wedge X_k$: true / Y (assumption-free default rule)

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-1} : X_k / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_k : X_{k-1} / Y$

...

$X_2 \wedge \ldots \wedge X_k : X_1 / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-2} : X_{k-1}, X_k / Y$

$X_1 \wedge X_2 \wedge \ldots \wedge X_{k-1} : X_{k-2}, X_k / Y$

…

$X_2 \wedge \ldots \wedge X_{k-1} : X_1, X_k / Y$

…

true : $X_1, X_2, \ldots, X_{k-2}, X_{k-1}, X_k / Y$ (knowledge-free default rule)

Each one of the 2^k possible states is characterized by what is known and what is not known to the agent, i.e. it represents what we called the single-norm KH status of the agent. These possible KH states may be organized in a multi-level hierarchy as proposed in the previous chapter. But, now, it turns out that this structure is, in fact, a lattice structure of height $k+1$, where the binary relation that causes them to be partially ordered is the number of assumptions employed. That is, the formulation of a norm at level 0 is the single assumption-free default; level 1 contains the k one-assumption defaults, and so on, until the top level which contains the single, knowledge-free default.

This structure, in constrast to what we proposed in the previous chapter, may be traversed either bottom-up or top-down causing the P and J sets to contract or expand accordingly. An agent trying to choose the appropriate formulation for a norm, given its current knowledge, traverses the structure upwards starting from level 0, and in this case, at each level l computes that $P_l = P_{l-1} - \{X_j \mid X_j \text{ is not known explicitly}\}$ and $J_l = J_{l-1} \cup \{X_j \mid X_j \text{ is not known explicitly}\}$, where $1 \le j \le k$ and $0 \le l \le k$. If $l=0$, then $P_0 = P$ and $J_0 = \emptyset$. An agent that receives new information, which necessitates the retraction of previously drawn conclusions, traverses the structure downwards, starting from some level m (this is the level of the norm formulation that it employed in its reasoning before it received the new information) and in this case computes at each level $(l-1)$ that $P_{l-1}=P_l \cup \{X_j \mid X_j \text{ is retracted}\}$ and J_{l-1}

$= J_l - \{X_j \mid X_j \text{ is retracted}\}$, where $1 \leq j \leq k$ and $0 < l \leq m$. If m=k, then $P_k = \varnothing$ and $J_k = J$. Of course, defaults contained in the same level have the same number of assumptions.

To illustrate this idea, schematically, consider the following norm, which involves three conditions:

$Y \leftarrow X_1 \wedge X_2 \wedge X_3$

The corresponding lattice structure of height 4 is shown in Figure 7.1 where each level contains the following defaults:

Level 0: $\{ D_{0,1} \equiv X_1 \wedge X_2 \wedge X_3 : \text{true} / Y \}$

Level 1: $\{ D_{1,1} \equiv X_1 \wedge X_2 : X_3 / Y, \quad D_{1,2} \equiv X_1 \wedge X_3 : X_2 / Y, \quad D_{1,3} \equiv X_2 \wedge X_3 : X_1 / Y \}$

Level 2: $\{ D_{2,1} \equiv X_1 : X_2, X_3 / Y, \quad D_{2,2} \equiv X_2 : X_1, X_3 / Y, \quad D_{2,3} \equiv X_3 : X_1, X_2 / Y \}$

Level 3: $\{ D_{3,1} \equiv \text{true} : X_1, X_2, X_3 / Y \}$

$D_{\text{level,number}}$ denotes the level of the default and its identification number within its level, and it is used to facilitate reference.

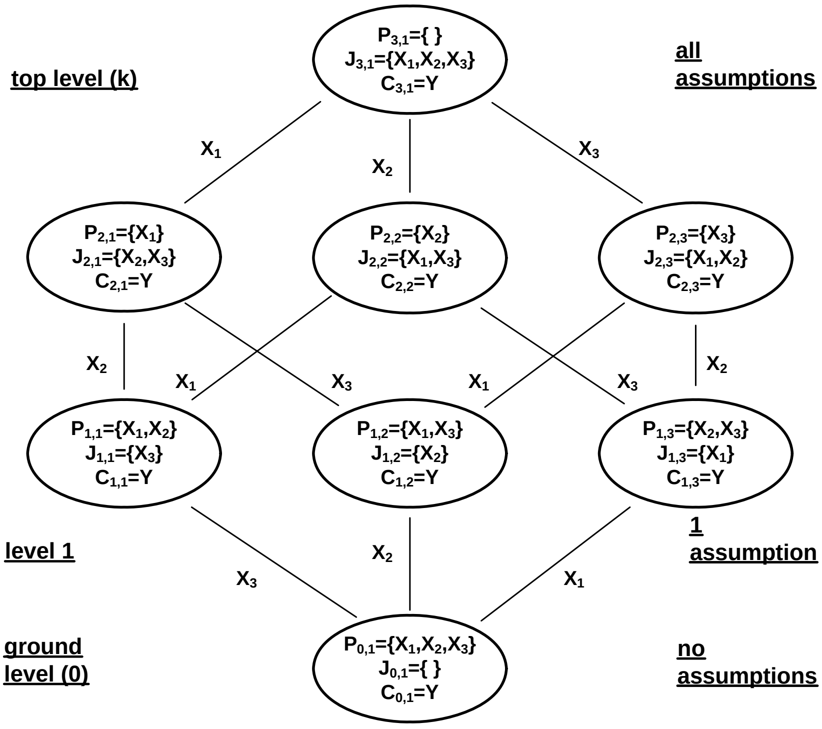


Figure 7.1 Expansion/contraction of KH status

In Figure 7.1, the assumption space expands, and the corresponding knowledge base contracts, when the agent moves upwards in the lattice structure. Conversely, the assumption space contracts and the corresponding knowledge base expands, when the agent moves downwards.

Mathematical Properties of Lattices

Note that, although, the proposed mapping of a norm of the form (1) into defaults results in an exponential number of defaults, this is manageable assuming a deterministic number of conditions in the initial norms. In fact, let $|L|$ denote the total number of defaults contained at level L, where $0 \leq L \leq k$, and k is the total number of conditions in an initial rule of the form (1). Then, it is easy to verify that the following properties hold:

- $|L| = 1$ if $L = 0$
- $|L| = (k - L + 1) * |L\text{-}1| / L$ if $L \neq 0$

Also, note that, an agent that places itself on a specific node in a level, on the basis of its current knowledge, either attempts to move upwards by expanding its assumption space or attempts to move downwards by expanding its knowledge. In both cases the agent does not need to examine all the defaults that are contained in the next level upwards or the previous level downwards. This is because of the binary relation that holds among the defaults and causes them to be partially ordered on the basis of the number of assumptions employed at each level. Specifically, consider that an agent finds itself at some state al level L. Then, in the case the agent reasons on the basis of new assumptions, i.e. moving upwards, the possible new defaults that the agent should examine are restricted to $k - L$, i.e. all the possible children defaults that may be computed form its current status (the parent node); in the case the agents reasons nonmonotonically, i.e. moving downwards, the possible defaults that the agent should examine are L, i.e. all the possible parent defaults that may be computed form its current status (the child node). Therefore, the lattice structure, first, enables an agent to position itself among all its possible KH states, and second, enables an agent to move upwards or downwards, i.e. reason hypothetical or nonmonotonically, via the parent/child relation that hold among the lattice nodes.

Recall that, an agent in order to populate the lattice levels needs to compute the power set $\mathscr{P}(S)$ of a given set S. For instance, given a set $S = \{a, b, c, d\}$ of $n=4$ elements, the corresponding power set is the set $\mathscr{P}(S)$ that contains 2^n elements as follows:

$\mathscr{P}(S) = \{ \{\}, \{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,c\}, \{a,d\}, \{b,c\}, \{b,d\}, \{c,d\}, \{a,b,c\}, \{a,b,d\}, \{a,c,d\}, \{b,c,d\}, \{a,b,c,d\} \}$

As shown in the previous section 6.3, the elements of $\mathscr{P}(S)$ may be seen as binary integers of length n, i.e.:

$\mathscr{P}(S) = \{ \{0000\},\{1000\},\{0100\},\{0010\},\{0001\},\{1100\},\{1010\},\{1001\},\{0110\},\{0101\},\{0011\},\{1110\},\{1101\},\{1011\},\{0111\},\{1111\} \}$

or as their decimal equivalents:

$\mathscr{P}(S) = \{ 0, 8, 4, 2, 1, 12, 10, 9, 6, 5, 3, 14, 13, 11, 7, 15 \}$

This particular view, also, helps us towards the computation of the relations that hold among the lattice nodes. Let us call the binary integers and their decimal equivalents of level 1 as *base* nodes, and the set that contains them as the *base set*. For instance, given the above set $S = \{a, b, c, d\}$ of $n=4$ elements, the base elements are those contained in the base set $S = \{\{0001\}, \{0010\}, \{0100\}, \{1000\}\}$ or their decimal equivalents $S = \{1, 2, 4, 8\}$. These nodes are nodes that employ only one assumption. We see that whenever an agent find itself in a node N it either needs to move upwards or needs to move downwards. For both of the above cases the agent seeks to find all possible neighbour nodes to move to, i.e. child nodes that are at the next level upwards when reasoning hypothetically, or parent nodes that are at the previous level downwards when reasoning non-monotonically. It is clear that these neighbours are being computed via the manipulation of lists, binary decimal integers, as shown in Algorithm 5 (Table 7.1).

This algorithm is based on the Algorithm 2 (Table 6.1), as shown in the previous chapter, for the construction of the single-norm KH structure. Given a node N and the set of base elements that correspond to this node, we may calculate the neighbours of N by adding and abstracting the decimal integers of the base elements to the decimal integer that correspond to the element N. We use an auxiliary prodedure named *NumberOfAssumptions* which calculates the total number of aces in the binary format that correspond to an element, i.e. either element N or their potential children/parents. Whenever, a potential child or parent node employs more or less assumptions, correspondingly, and belongs within the lattice limits ($0 \leq \text{decimal integer} \leq 2^k-1$), then we may characterize the node explicitly, either as a child or as a parent node.

Table 7.1 Algorithm 5: Computation of the neighbour nodes

Algorithm 5

```
PROCEDURE NeighbourhoodForANode(N, L, BL)

VARIABLES
    INTEGER: i, j, b, max, min,  TempP, TempN
                        N // node represented as an decimal integer
                        L // integer representing the level where N is contained
    LIST: BL   // list that contains the base elements represented as decimal integers

START

   b=Size(BL);   // Size(LIST BL) returns the number of elements in the list BL,
                              i.e. b is the total number of the base elements

   max ← $2^b$ -1 // maximum decimal integer, i.e. element at the top level
   min ← 0      // minimum decimal integer, i.e. element at the ground level

   FOR i=1 TO i=k STEP 1 DO {

      FOR j=0 TO j=b-1 STEP 1 DO {

          TempP  ← N + BL[j]
          TempN  ← N - BL[j]

          // NumberOfAssumptions(INTEGER N) coverts the decimal integer N to its equivalent
          //       binary and, counts the number of aces (1), i.e. the umber of assumptions employed

         IF ( (NumberOfAssumptions(TempP) >  NumberOfAssumptions(N)) AND TempP < max +1 ) THEN {
                    TempP  is a child node
         ELSE IF ( (NumberOfAssumptions(TempN) <  NumberOfAssumptions(N)) AND TempN > min - 1 )
THEN {
                    TempN  is a parent node
         }   // end of IF

      }   // end of FOR

   }   //end of FOR

END_ PROCEDURE
```

Up to now we enabled an agent to seek for useful assumptions and indicate a set of appropriate formulations of defaults to place in the D set of the DfT representing the norms that it will use during its reasoning. The second question that arises towards assumption identification is “which substitution instance of initial norm conditions that were marked as assumptions should be chosen and employed in the inference procedure”, i.e., tantamount to the question “which is the appropriate ground instance of a default to use during the reasoning”, for each of the norms in the initial set of norms. To this end, we adopt *Herbrand semantics* for the FOL language. The *Herbrand universe* of a FOL language is the set of all ground terms. The *Herbrand* *base* of a FOL language is the set of all ground atoms formed using elements of the Herbrand universe as arguments. Thus, an agent may identify possible grounded assumptions searching among grounded formulae computed on the basis of the Herbrand semantics.

7.3 Theory Construction

Of course, systems are typically subjects to multiple norms, each of which may be formulated as a default, and candidate default formulations are organized in a lattice, such as the one described above. Hence, the DfT representation of a set of norms is a pair of the form (W, D), where W is a set of logic formulae that represent currently available knowledge, and D is a set of lattices, each containing the possible formulations of a norm as a default. Note that each initial norm may be mapped to one of many candidate defaults, and during its reasoning the agent will employ only one default formulation for each initial norm.

To illustrate this, let us assume that a normative system comprises two norms of the form:

$R1 \equiv Y_1 \leftarrow X_1 \wedge X_2$ and $R2 \equiv Y_2 \leftarrow X_3 \wedge X_4 \wedge X_5$

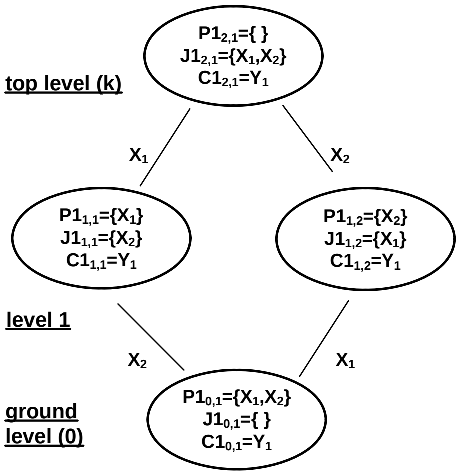


Figure 7.2 Expansion/contraction of KH status for rule R1

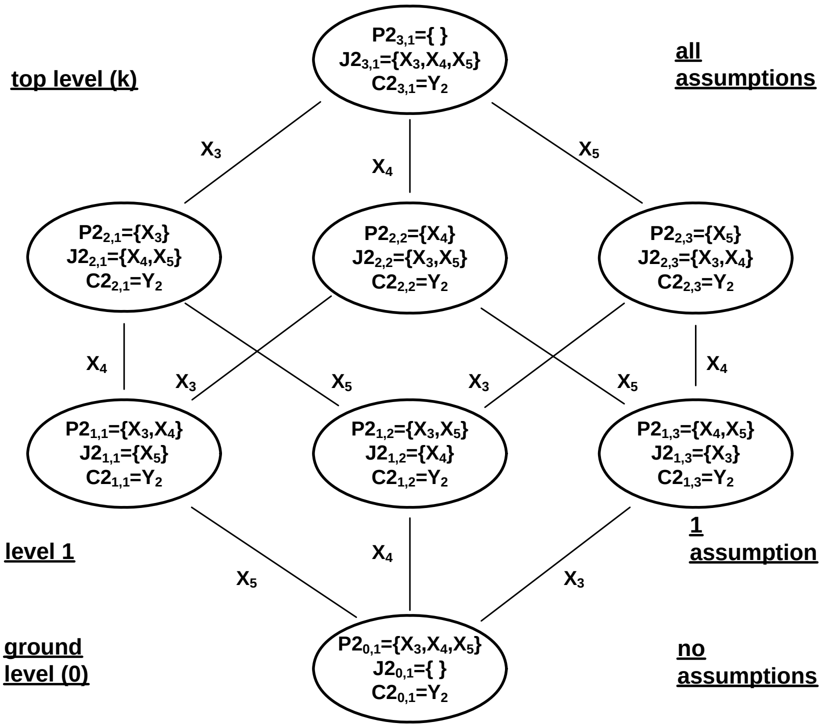


Figure 7.3 Expansion/contraction of KH status for rule R2

Single-norm Lattice Structure for R1:
Level 0^{R1}: { $D1_{0,1} \equiv X_1 \wedge X_2$: true / Y_1 }

Level 1^{R1}: { $D1_{1,1} \equiv X_1 : X_2 / Y_1$, $D1_{1,2} = X_2 : X_1 / Y_1$ }

Level 2^{R1}: { $D1_{2,1} \equiv$ true : $X_1, X_2 / Y_1$ }

Single-norm Lattice Structure for R2:
Level 0^{R2}: { $D2_{0,1} \equiv X_3 \wedge X_4 \wedge X_5$: true / Y_2 }

Level 1^{R2}: { $D2_{1,1} \equiv X_3 \wedge X_4 : X_5 / Y_2$, $D2_{1,2} = X_3 \wedge X_5 : X_4 / Y_2$, $D2_{1,3} = X_4 \wedge X_5 : X_3 / Y_2$ }

Level 2^{R2}: { $D2_{2,1} \equiv X_3 : X_4, X_5 / Y_2$, $D2_{2,2} = X_4 : X_3, X_5 / Y_2$, $D2_{2,3} = X_5 : X_3, X_4 / Y_2$ }

Level 3^{R2}: { $D2_{3,1} \equiv$ true : $X_3, X_4, X_5 / Y_2$ }

7.4 Inference Procedure

Reasoning starts with the agent to attempt to position itself on specific nodes for each lattice contained in the D set of its theory. The initialization is useful for two reasons: first, the agent determines what is known and what is not, and, second, it sets a starting point for its inference by determining exactly what are the available options whenever the agent needs to reason hypothetically or non-monotonically by using the parent-child (moving upwards) or child-parent (moving downwards) relationships for its current state (node). Next the agent starts to apply defaults in order to derive conclusions. For now, consider a quantitative criterion that helps the agent to decide which default to apply, in the case where two or more defaults apply, that correspond to two or more initial rules, at the same time. This criterion depends on the number of knowledge/assumptions employed for each default, i.e., minimality of assumptions is considered as the criterion. Later, we discuss why this criterion is inadequate for rational agents and we propose alternatives. Each time a default applies its conclusions are included in the current extension that is being computed. In our previous approach to assumption-based reasoning, as explained in chapter 6, we noted that, whenever, there are no further defaults that can be applied in a level this signals to the agent that it needs to employ further assumptions in order to proceed, and inference continues by examining defaults that lie in the next level upwards. The case where reasoning is possible using only defaults from the ground levels is identical to inference in classical logic, but here we are also able to preserve consistency of entailment, if we want to employ appropriate variations of DfL such as Constrained Default Logic [225] as we discuss later.

Now, we see that, during inference, the agent needs to update its KH status, for two reasons: first each time a default applies the derived conclusions may influence the current state of the agent in the lattices; second, we have considered an environment that is open and thus information and agents come and leave in a vaguely manner. Therefore, an agent needs to update its internal state, i.e. its position on lattices.

The inference is done according to the Algorithm 6 (Figure 7.4). The *Listener* is a procedure that detects any changes of the world and refers these to the agent in order to update its world. Thus, the agent’s world is affected (updated) first by endogenous factors, e.g. its inference process, and second, by exogenous factors, e.g. changes in the environment or other agent’s valid actions.

To illustrate the reasoning process, consider the previous example with the two norms R1, R2 and the corresponding lattices shown in Figures 7.2 and 7.3. Here are some possible scenaria, with different initial knowledge available each time, in the beginning of the reasoning process:

- if $W=In(0)=\{X_1, X_2\}$ and $Out(0)=\varnothing$ then the agent initially places itself to the nodes $D1_{0,1}$ and $D2_{3,1}$ for $R1$ and $R2$ respectively. First, the extension $In(1)=\{X_1, X_2, Y_1\}$ ($Out(1)=\{\ \}$) is computed on a factual basis by applying default $D1_{0,1}$. Now the agent is able to update its world and reposition itself into the KH lattices for the defaults that have not fired yet. No other changes are detected so the extension $In(2)=\{X_1, X_2, Y_1, Y_2\}$ is computed by making the assumption that X_3, X_4 and X_5 hold ($Out(2)=\{\neg X_3, \neg X_4, \neg X_5\}$) and by applying default $D2_{3,1}$.

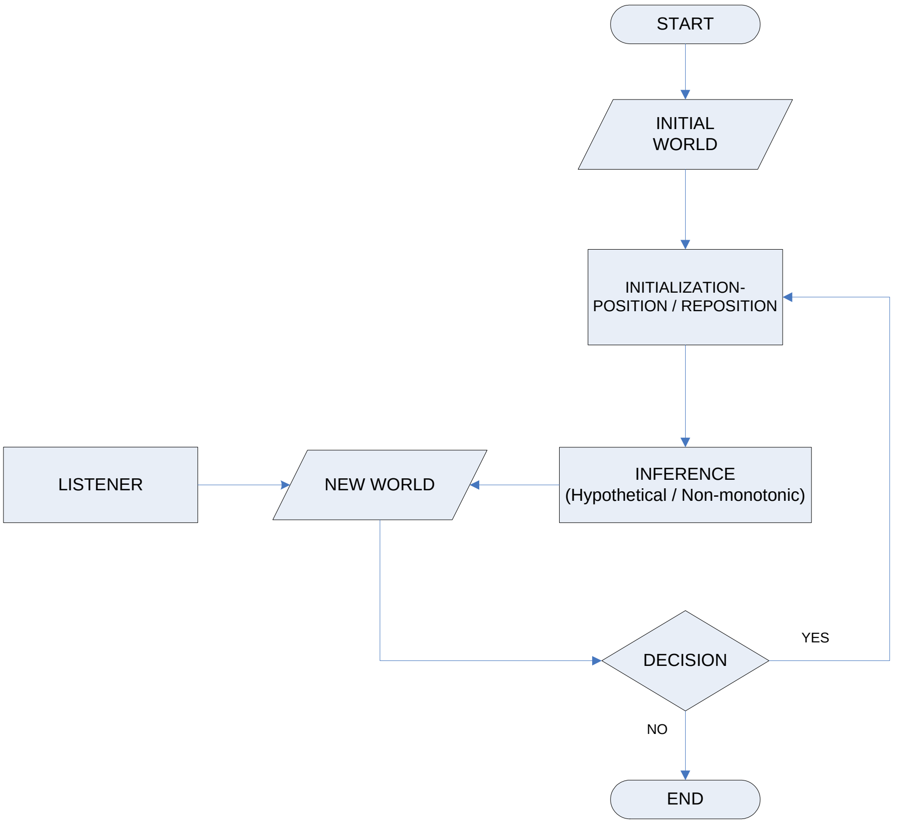


Figure 7.4 Algorithm 6: Life-cycle of reasoning in open environments

- if $W=In(0)=\{X_1, X_2, X_3\}$ and $Out(0)=\varnothing$ then the agent initially places itself to the nodes $D1_{0,1}$ and $D2_{2,1}$ for $R1$ and $R2$ respectively. First, the extension $In(1)=\{X_1, X_2, X_3, Y_1\}$ ($Out(1)=\{\ \}$) is computed on a factual basis by applying defaults $D1_{0,1}$. Now the agent is able to update its world and reposition itself into the KH lattices for the defaults that have not fired yet. No other changes are detected so the extension $In(2)=\{X_1, X_2, X_3, Y_1, Y_2\}$ is computed by making the assumption that X_4 and X_5 hold ($Out(2)=\{\neg X_4, \neg X_5\}$) and by applying default $D2_{2,1}$.

- if $W=In(0)=\{X_1, X_3, X_4, X_5\}$ and $Out(0)=\varnothing$ then the agent initially places itself to the nodes $D1_{1,1}$ and $D2_{0,1}$ for R1 and R2 respectively. First, the extension $In(1)=\{ X_1, X_3, X_4, X_5, Y_2\}$ ($Out(1)=\{ \}$) is computed on a factual basis by applying defaults $D2_{0,1}$. Now the agent is able to update its world and reposition itself to the KH lattices for the defaults that have not fired yet. No other changes are detected so the extension $In(2)=\{ X_1, X_3, X_4, X_5, Y_2, Y_1\}$ is computed by making the assumption that only X_2 holds ($Out(2)=\{\neg X_2\}$) and by applying default $D1_{1,1}$.

Note that although a level may contain two or more defaults that correspond to the same initial norm (e.g. $D2_{1,1}$ or $D2_{1,2}$ or $D2_{1,3}$) there is no need for some kind of prioritization amongst them. If two or more defaults of the same level, which are derived from the same initial norm, were to apply simultaneously, then the more general default contained in the immediately lower level should have applied. Moreover, because inference involves one lattice level at a time in a step-wise manner, the agent employs the fewest possible hypotheses.

During its reasoning, the agent will need to remember which default formulation it chose for each of the norms that it reasons with, i.e., it needs to remember which node it chose for each of the lattices, in order to be able to answer question 2H. Moreover, as its reasoning progresses and new information becomes available, either merely augmenting its knowledge base, or updating some part of it, the agent will need to update its choice of default formulations, moving upwards or downwards within each lattice structure. Upward moves correspond to the agent trying to answer question 1H, while downward moves correspond to the agent trying to answer question 3H.

In order to illustrate all three issues of interest (1H - 3H) at the same time, let us assume that a normative system comprises two norms of the form:

$R1\equiv X_3\leftarrow X_1\wedge X_2$ and $R2\equiv Y_2\leftarrow X_3\wedge X_4\wedge X_5$

Thus, the corresponding lattices' levels contain the defaults (lattices are identical in structure to the lattices shown in Figures 7.2 and 7.3):

Single-norm Lattice Structure for R1:

Level 0^{R1}: $\{ D1_{0,1} \equiv X_1\wedge X_2 : true / X_3 \}$

Level 1^{R1}: $\{ D1_{1,1} \equiv X_1 : X_2 / Y_1, D1_{1,2} = X_2 : X_1 / X_3 \}$

Level 2^{R1}: $\{ D1_{2,1} \equiv true : X_1,X_2/ X_3 \}$

Single-norm Lattice Structure for R2:

Level 0^{R2}: { $D2_{0,1} \equiv X_3 \wedge X_4 \wedge X_5 : true / Y_2$ }

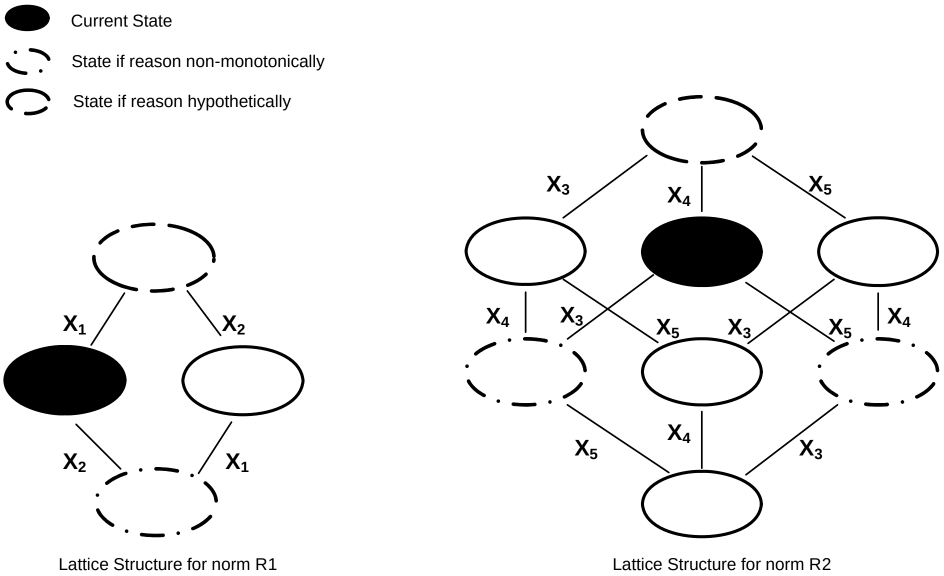
Level 1^{R2}: { $D2_{1,1} \equiv X_3 \wedge X_4 : X_5 / Y_2$, $D2_{1,2} = X_3 \wedge X_5 : X_4 / Y_2$, $D2_{1,3} = X_4 \wedge X_5 : X_3 / Y_2$ }

Level 2^{R2}: { $D2_{2,1} \equiv X_3 : X_4, X_5 / Y_2$, $D2_{2,2} = X_4 : X_3, X_5 / Y_2$, $D2_{2,3} = X_5 : X_3, X_4 / Y_2$ }

Level 3^{R2}: { $D2_{3,1} \equiv true : X_3, X_4, X_5 / Y_2$ }

Here is a possible scenario:

- if $W=In(0)=\{X_1, X_4\}$ and $Out(0)=\varnothing$ then the agent initially places itself to the nodes $D1_{1,1}$ and $D2_{2,2}$ for R1 and R2 respectively (Figure 7.5). First, the extension $In(1)=\{X_1, X_4, X_3\}$ ($Out(1)=\{\neg X_2\}$) is computed by applying default $D1_{1,1}$.

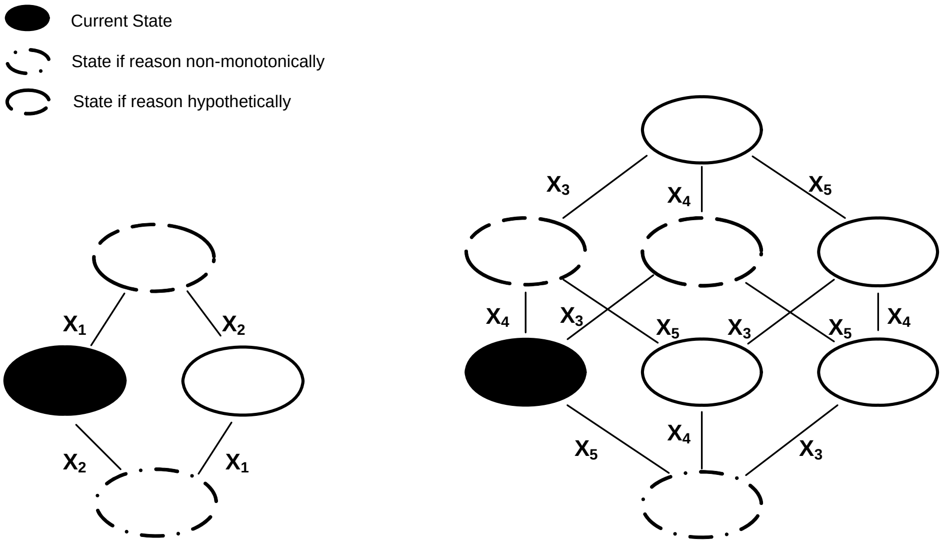


Lattice Structure for norm R1

Lattice Structure for norm R2

Figure 7.5 Initialization

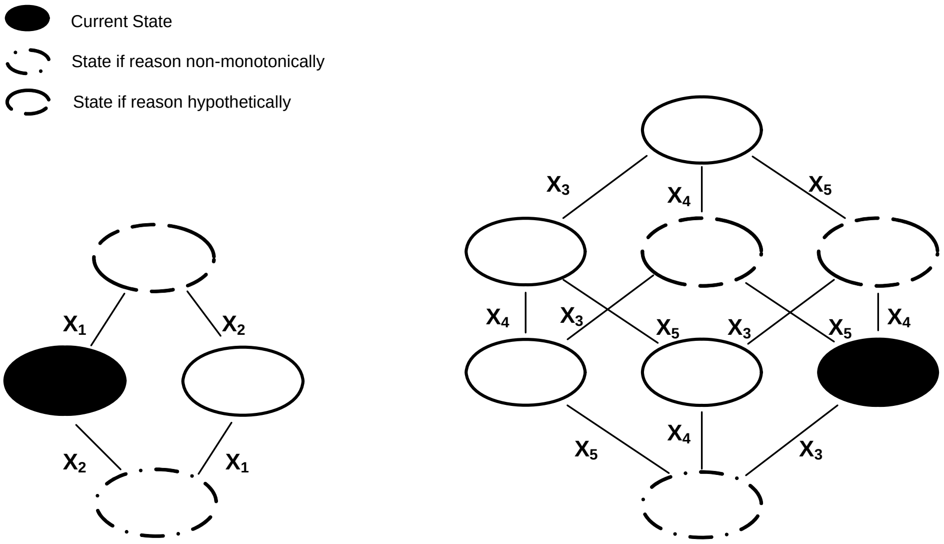
Now the agent is able to update its world and reposition itself into the KH lattices for the defaults that haven't fired, yet. Specifically, the agent now possesses the knowledge that X_3 holds, and consequently, it is able to change its position into the KH lattice for norm R2, and move to the node $D2_{1,1}$ (Figure 7.6). Note that, although, the agent moves downwards to the lower level that contains three nodes ($D2_{1,1}$, $D2_{1,2}$, $D2_{1,3}$), only the two of them ($D2_{1,1}$, $D2_{1,3}$) are acceptable nodes due to their connection with the child node ($D2_{2,2}$). Moreover, this connection indicates exactly which one of the nodes should the agent choose when moving downwards. As a result, the extension $In(2)=\{X_1, X_4, X_3, Y_2\}$ is computed by making the additional assumption that x_5 holds ($Out(2)=\{\neg x_2, \neg x_5\}$) and by applying default $D2_{1,1}$. This extension is computed on the initial commitment that x_2 holds. This commitment implies that x_3 holds and this is considered as factual knowledge on the next norm that fires. It is clear, that the agent is able to reason hypothetically (1H) and rely on its commitments (2H) in an argumentation-like manner.



Lattice Structure for norm R1

Lattice Structure for norm R2

Figure 7.6 Reposition



Lattice Structure for norm R1

Lattice Structure for norm R2

Figure 7.7 Reposition

This is the same fact that enables the agent to reason non-monotonically (3H). Consider that, at a later time point, the agent possesses the knowledge that $\neg X_2$ and X_5 hold. Now, the agent, first, should retract the employed assumption about X_2 and any conclusions derived on its basis, and second, needs to reposition itself again, into the KH lattices. Note that, in this case, the agent repositions itself to a different nodes from its initial states due to the augmented knowledge base, i.e. for $R1$ it still positions itself to the state $D1_{1,1}$, and for $R2$ it repositions itself to the state $D2_{1,3}$ (Figure 7.7). As a result, now, the extension $In(1)=\{X_1, X_4, \neg X_2, X_5, Y_2\}$ is computed by making the single assumption that X_3 holds ($Out(1)=\{\neg X_3\}$) and by applying default $D2_{1,3}$, only ($D1_{1,1}$ can not fire).

7.5 Managing the Space of Knowledge/Hypothesis

So far, we have argued that agents must resort to assumptions in order to reason in the presence of incomplete knowledge, and we have shown a way in which they may be able to identify candidate assumptions and employ them dynamically. We have shown that the space of possible hypotheses available to an agent is essentially infinite, since it treats any literal that it does not know about *explicitly* as a candidate assumption. As a result, it affords an agent the ability to use all available notions that the representation language supports, e.g. information about actions and time, deontic notions, physical and legal ability, roles, beliefs, etc. Such information may be relative to the agent itself, other agents or the environment. This suggests that our proposal is distinct from other approaches to assumption-based reasoning.

Other approaches to dynamic assumption-based reasoning (e.g. [52, 158, 51, 205, 206, 2, 192, 193, 185, 131, 233]) rely on a finite hypotheses space, which is either pre-specified (usually in this case it is referred to as *assumption pool*) and is explored dynamically, or is identified dynamically by goal-driven generation, i.e. the agent has specific conclusions that it wants to derive and identifies what assumptions are required in order to perform its derivations. Although the goal-oriented view on hypothetical reasoning facilitates the implementation of a tool, e.g. an agent, we think that it is something more than this, perhaps related to the agent’s personality and distinct approach to reasoning (e.g. cautious vs. risky). Our approach enables agents to be more *independent* and *open-minded* (one might say more like humans) and is distinct with respect to other approaches to hypothetical reasoning, due to the fact that we prefer to use appropriate mechanisms in order to manage the full assumption space.

There are three aspects to managing the space of hypotheses:

- How do we ensure that the set of hypotheses that an agent uses to draw a conclusion is consistent, and that the possible world models that the agent infers are rational?
- How do we restrict the space of hypotheses, when this is imperative, due to constraints that arise from a particular application domain?
- Is the order in which an agent identifies and employs hypotheses important, i.e. does it affect what conclusions it may draw, whether these are rational, and the extent to which the agent bases its reasoning on reality rather than wishful thinking?

Next, we discuss each of these issues in turn and propose qualitative criteria to manage the space of assumptions and possible defaults to apply.

7.5.1 Hypotheses Rationality and Consistency

One may argue that following Reiter’s original computation of extensions within DfT we may compute possible world models that are counter-intuitive.

For instance, according to the knowledge base:

$$\{\ \text{Has(Fledgling,Feathers)},\ \forall x\ (\text{Has(x,Feathers)} \wedge \text{Flies(x)}) \rightarrow \text{Is(x,Eagle)},\ \forall x\ (\text{Has(x,Feathers)} \wedge \neg\text{Flies(x)}) \rightarrow \text{Is(x,Pengiun)}\ \}$$

an agent may compute an extension where the fledgling bird is an eagle and a penguin on the basis of the simultaneous assumptions that it can and cannot fly, i.e.:

$\text{In}(2) = \{\ \text{Has(Fledgling,Feathers), Is(Fledgling,Eagle), Is(Fledgling,Pengiun)}\}$ and $\text{Out}(2) = \{\ \neg\text{Flies(Fledgling), Flies(Fledgling)}\}$.

Moreover, for instance, in a business realistic example, a buyer agent would infer an extension which suggests that it infers a possible version of the world, in which it bears an obligation to pay the seller agent, although no delivery from the seller agent is explicitly recorded in this world, and similarly that the seller agent bears an obligation to deliver, although this world does not explicitly record that the buyer’s agent order is valid (*best-guess reasoning*).

This view of extensions, separated from assumptions, as possible world models, is clearly undesirable. To overcome this problem we may employ Constrained Default Logic [225] and require joint consistency of default assumptions. The possible world model that the agent infers incrementally, for a CDfT, is the consistent set $In(i) \cup \neg Out(i)$. This is tantamount to saying that the possible world models inferred by the agent contain, besides previous knowledge, both the consequents and the assumptions of the applied defaults.

Moreover, note that it is important to consider the issue of consistency between assumptions employed during the reasoning process and new inferences derived as a result of the reasoning process. One of the reasons for which it is impossible to resort to proof for the construction of the DfT, is precisely because we need a revision mechanism in order to reconstruct the default rules as new information becomes available, so that the agent could prove literals from its updated knowledge, and hence identify them correctly as prerequisites or justifications, i.e. candidate assumptions. By constructing lattice structures we facilitate the requirement to revise the defaults. This is because inference on the lattice structure involves one level at a time, and should new information become available, the agent can move upwards or downwards to the required level of the lattice. Incidentally, in this way it is also guaranteed that during its inference, the agent will employ the fewest possible hypotheses, i.e. the conclusions it derives at any given time are committed to its current knowledge to the largest possible extent, and it only makes assumptions when it really has to.

Towards commitment to previous worlds, an agent may use special default rules such as normal and semi-normal defaults as already discussed in the subsection 6.3.3.

7.5.2 Hypotheses Restriction

It may be risky for an agent to employ assumptions in full freedom. Full autonomy in assumption identification may be unsafe and lead to undesirable situations such as lack of control, unpredictable effects and counter-intuitive worlds. Thus, a mechanism capable of *motivating* and *adjusting* the agents’ hypothetical reasoning is really essential.

For this purpose, we see that DfL’s syntax and semantics can be really helpful. Recall that during its reasoning, the agent computes the extension of its theory incrementally and at each step i of the reasoning process constructs the set $In(i)$, which contains all previously available knowledge together with any new derived knowledge. The $Out(i)$ set computed at each step of this reasoning process contains formulae that should not turn out to be true i.e., the negation of formulae that are employed as assumptions. By initializing the Out set appropriately, we may control the agent in its identification and deployment of assumptions, and hence we may control its autonomy.

Here are some possible scenaria, for the examples discussed above but now with different initializations of the Out set:

$R1 \equiv Y_1 \leftarrow X_1 \wedge X_2$ and $R2 \equiv Y_2 \leftarrow X_3 \wedge X_4 \wedge X_5$

The corresponding lattices' levels contain the defaults (Figures 7.2 and 7.3):

Single-norm Lattice Structure for $R1$:

Level 0^{R1}: $\{ D1_{0,1} \equiv X_1 \wedge X_2 : true / Y_1 \}$

Level 1^{R1}: $\{ D1_{1,1} \equiv X_1 : X_2 / Y_1, D1_{1,2} = X_2 : X_1 / Y_1 \}$

Level 2^{R1}: $\{ D1_{2,1} \equiv true : X_1, X_2 / Y_1 \}$

Single-norm Lattice Structure for $R2$:

Level 0^{R2}: $\{ D2_{0,1} \equiv X_3 \wedge X_4 \wedge X_5 : true / Y_2 \}$

Level 1^{R2}: $\{ D2_{1,1} \equiv X_3 \wedge X_4 : X_5 / Y_2, D2_{1,2} = X_3 \wedge X_5 : X_4 / Y_2, D2_{1,3} = X_4 \wedge X_5 : X_3 / Y_2 \}$

Level 2^{R2}: $\{ D2_{2,1} \equiv X_3 : X_4, X_5 / Y_2, D2_{2,2} = X_4 : X_3, X_5 / Y_2, D2_{2,3} = X_5 : X_3, X_4 / Y_2 \}$

Level 3^{R2}: $\{ D2_{3,1} \equiv true : X_3, X_4, X_5 / Y_2 \}$

- if $W = In(0) = \{X_1, X_2\}$ and $Out(0) = \{Y_2\}$ then extension $In(1) = \{X_1, X_2, Y_1\}$ is computed by applying only $D1_{0,1}$ default ($Out(1) = \{Y_2\}$). No assumptions are possible due to the initial restriction on what can be *inferred*.
- if $W = \{X_1, X_2, X_3\}$ and $Out(0) = \{X_4, X_5\}$ then extension $In(1) = \{ X_1, X_2, X_3, Y_1\}$ is computed by applying only $D1_{0,1}$ default ($Out(1) = \{X_4, X_5\}$). Again no assumptions are possible due to the initial restriction on what can be *assumed*.
- if $W = \{X_1, X_3, X_4, X_5\}$ and $Out(0) = \{X_2, Y_2\}$ then no extension is computed due to the initial restriction on what can be *inferred* and *assumed*.

A question that arises naturally is ‘What kind of information can be used to restrict the assumption space?’. The answer is simple: ‘Any kind of information!’. And this answer is what is really important in our proposal. It means that we are able to use all available notions that our representation language supports. For instance, we may include information about actions and time, deontic notions, physical and legal ability, roles, beliefs, etc. Such information may relate to the agent itself, other agents or the environment.

The next question that arises is “How must we initialize the $Out(i)$ set?”. We may use a Pre-constrained Default Theory. A PcDfT is a triple of the form (W, D, PC), where (W, D) is a CDfT and PC is set of formulae that are considered as the constraints of the theory [226]. For the first step of the process, i.e., for $i=1$, $In(0)=W$ and $Out(0) = PC$. It is clear that, the role of pre-constraints is identical to the role of the $Out(i)$ set in initializing and adjusting agents’ hypotheses. By initializing the PC set appropriately, an agent may specify and apply a certain strategy in its reasoning. Note that the formulae contained in PC define what the agent concedes or expects, and not factual knowledge. For this reason it is separated from W.

For example, in case of incomplete knowledge about the carrier’s agent validity to perform delivery and to accept payment on delivery, the buyer agent should make some assumptions in order to proceed with inference. A risky agent may accept that the carrier agent is legally (and practically, obviously) empowered to perform delivery and accept payment, thus the validity of the corresponding actions could be assumed. On the other hand, a cautious agent may accept the assumptions regarding the action of delivery but not regarding the action of accepting payment. To model this cautious strategy the agent may insert into PC that legal power or validity of the carrier agent to receive payment holds.

7.5.3 Hypotheses Sequence

So far, an agent is able to deal with the problem of incomplete knowledge by reconstructing the initial domain representation into a DfT. A question that arises during assumption-based reasoning is what is the reasonable sequence for employing assumptions? We believe that this question and its answer are strongly related to causality.

We may use stratification for a DfT [45, 46] or for its two variants (CDfT and PcDfT) [13]. Recall that, a DfT is called stratified iff there exists a stratification function s that assigns a natural number to each default. As a result of the application of a stratification function the set of default rules is ordered into strata.

Note that the way in which agents construct lattice structures, as per our proposal, resembles, in a way, stratification of a DfT. The possible default formulations of each initial norm are assigned to the various lattice levels, depending on the number of assumptions that each default formulation employs. That is, the number of assumptions may be regarded as somewhat similar to a stratification criterion applied to the set of possible default formulations for each initial norm.

The question that arises naturally, now, is how these two distinct ordering methods (one due to stratification and one due to the number of assumptions) relate to each other. Stratification aims at preserving causal relations between defaults, while the organization of defaults into lattice structures aims at ensuring that agents employ the fewest possible hypotheses at each step of their reasoning process, and thus base their conclusions on facts as much as possible, rather than on assumptions. The set of lattices that the agent possesses may be subjected to stratification, so that the agent chooses a reasonable order in which to apply default rules, and preserve any causal relations between defaults. Once a particular lattice, belonging to a particular stratum, is chosen, the agent establishes which node of the lattice corresponds to its current knowledge base (and therefore assumption requirements). Note that the agent may use different levels of different lattices. The precise level to be used in each lattice is determined by its current knowledge. The precise lattice to use at each point is determined by the stratification function. In this way an agent infers some knowledge, even on a (partially or totally) hypothetical basis, which causes the entailment of other knowledge in an argumentation-like manner (cf [198]**)** and we may characterize its conclusions in the same way that is used to characterize arguments (e.g. defeasible).

To illustrate this, let us assume the previous normative system containing the norms $R1$ and $R2$ enhanced with two additional norms $R3$ and $R4$ of the form:

$R3=X_2 \leftarrow X_6$ and $R4=X_5 \leftarrow X_7 \wedge X_8$

The corresponding lattices' levels for norms $R3$ and $R4$ contain the defaults:

Single-norm Lattice Structure for R3:

Level 0^{R3}: $\{ D3_{0,1} = X_6 : true / X_2 \}$

Level 1^{R3}: $\{ D3_{1,1} = true : X_6 / X_2 \}$

Single-norm Lattice Structure for R4::

Level 0^{R4}: $\{ D4_{0,1} = X_7,X_8 : true / X_5 \}$

Level 1^{R4}: $\{ D4_{1,1} = X_7 : X_8 / X_5,\ D4_{1,2} = X_8 : X_7 / X_5 \}$

Level 2^{R4}: $\{ D4_{2,1} = true : X_7,X_8 / X_5 \}$

Under a stratified theory it is clear that R3 and R1 as well as R4 and R2, are mapped into different strata because R1 and R2 use the consequents of norms R3 and R4, respectively, either in the prerequisites or the justifications sets. Here are some possible scenaria, with different initial knowledge available each time, in the beginning of the reasoning process:

- if $W=In(0)=\{X_1, X_6, X_3\}$ and $Out(0)=\varnothing$ then extension $In(4)=\{X_1, X_6, X_3, X_2, Y_1, X_5, Y_2\}$ is computed by making the assumption that X_7, X_8 and X_4 hold ($Out(4)=\{\neg X_7, \neg X_8, \neg X_4\}$) and by applying defaults $D3_{0,1}$, $D1_{0,1}$, $D4_{2,1}$ and $D2_{1,2}$ respectively. Note that, in this case, the agent first infers X_5 on the assumption of X_7, X_8 and later infers Y_2 only on the assumption of X_4, while it uses its previous decisions towards this scope.
- if $W=In(0)=\{X_1, X_2, X_3, X_7, X_8\}$ and $Out(0)=\varnothing$ then extension $In(3)=\{X_1, X_2, X_3, X_7, X_8, Y_1, X_5, Y_2\}$ is computed by making the assumption that X_4 hold ($Out(3)=\{\neg X_4\}$) and by applying defaults $D1_{0,1}$, $D4_{0,1}$ and $D2_{1,2}$ respectively. Note that, in this case, there is no reason for an agent to use any of the two possible default formulations for the initial norm R3, because this norm adds no useful information to its knowledge base.

To sum up, we see that, an agent, when uses stratification on the set of available lattice structures and then performs its reasoning within the lattices, does not miss any causal knowledge and avoids employing unhelpful assumptions.

7.6 Examples

So far in our discussion, we have used abstract examples, in order to facilitate focusing on concepts rather than the particulars of a specific domain of an application. Here we illustrate the points raised in the preceding discussion, with reference to one general example and one e-commerce example. Of course, the technique proposed in this thesis, is more generally applicable to other open multi-agent scenaria, where agents have to reason with incomplete or inconsistent knowledge and their behaviour is regulated by some norms (e.g. cooperative distributed problem solving, task allocation etc).

7.6.1 General Example

Consider the following normative system:

Specie(x,Mammal) ← Has(x,Hair) ∧ Produce(x,Milk),

Specie(x,Bird) ← Has(x,Feathers) ∧ Lays(x,Eggs),

Is(x,Tigger) ← Specie(x,Mammal) ∧ Color(x,Brown-Orange) ∧ Has(x,Stripes),

Is(x,Pengiun) ← Specie(x,Bird) ∧ ¬Flies(x) ∧ Swims(x),

Is(x,Platypus) ← Produce(x,Milk) ∧ Lays(x,Eggs) ∧ Swims(x),

Is(x,Echidna) ← Produce(x,Milk) ∧ Lays(x,Eggs) ∧ Has(x,Spines),

Note that these norms are syntactically identical to the norms considered in the abstract examples presented above. Thus, the corresponding lattices of candidate default formulations for each one coincide with the lattices shown previously.

Now imagine that the initial knowledge is:

W=In(0)={ Has(Animal,Hair) ∨ Has(Animal,Feathers), Is(x, x') ∨ Is(x, x'') },

that is, we know that Animal either has hair or feathers. On the basis of this knowledge and without resorting to any assumptions inference is not possible. In contrast, on the basis of assumptions an agent is able to compute possible future worlds. Consider that we use a stratified and constrained DDfL. In this case, two extensions, i.e. possible worlds, are computed:

- 1st possible world:

  In'(2)=In(0) ∪ { Has(Animal,Hair), Specie(Animal,Mammal), Is(Animal,Tigger) } on the basis of the assumptions that Animal produces milk, its color is brown-orange and has stripes, i.e., Out'(2)={ ¬Produce(Animal,Milk), ¬ Color(Animal,Brown-Orange), ¬ Has(Animal,Stripes) }
- 2nd possible world:

  In''(2)= =In(0) ∪ { Has(Animal, Feathers), Specie(Animal,Bird), Is(Animal,Penguine) } on the basis of the assumptions that Animal lays eggs, can not fly but swims, i.e., Out''(2)={ ¬Lays(Animal,Eggs), Flies(Animal), ¬Swims(Animal) }

Note that, under the ODA, in each extension only two of the four initial norms and their corresponding defaults are used. That is, not due to the initial disjunction of factual knowledge that animal has either hair or feathers (as a matter of fact we are free to employ any assumption and infer a conclusion on a totally hypothetical basis), but due to the restriction of joint-consistency of the constrained variation of the theory that we use, i.e., assumptions employed and conclusions entailed must be consistent with the current knowledge. Moreover, due to stratification we employ assumptions and infer new knowledge in an argumentation-like manner, e.g. Animal is considered to be a mammal on basis of the assumption that it produces milk, and correspondingly, Animal is considered to be a tiger on the inference that it is a mammal and the assumptions regarding its brown-orange color and its stripes.

Until now, we have illustrated the way the proposed technique is used to address issues of interest 1H and 2H, which are dynamic assumption identification and the commitment to them. Now, consider the following scenario that illustrates the way our technique address issue of interest 3H, i.e., non-monotonicity. Imagine that the agent followed the second of the above two extensions and inferred that Animal is a Penguin. At a later time point it is informed that Has(Animal,Hair), Produce(Animal,Milk) and Lays(Animal,Eggs). This new information affects its previously drawn conclusions on the basis of the assumptions employed. Now, the agent needs to traverse the lattices downwards in order to choose an alternative formulation for the norms compatible with its current knowledge. Thus, the agent retracts its previously drawn conclusions and follows one of the alternative extensions that are computed:

- 1st possible world:
  In'(2)= In(0) ∪ { Has(Animal,Hair), Produce(Animal,Milk), Lays(Animal,Eggs } ∪ { Specie(Animal,Mammal), Is(Animal,Platypus) } on the basis of the assumption that Animal swims, i.e., Out'(2)={ ¬Swims(Animal) }
- 2nd possible world:
  In''(2)= In(0) ∪ { Has(Animal,Hair), Produce(Animal,Milk), Lays(Animal,Eggs) } ∪ { Specie(Animal,Mammal), Is(Animal,Echidna) } on the basis of the assumption that Animal has spines, i.e., Out''(2)={ ¬Has(Animal,Spines) }
- 3rd possible world:
  In'''(2)= In(0) ∪ { Has(Animal,Hair), Produce(Animal,Milk), Lays(Animal,Eggs)} ∪ {Specie(Animal,Mammal), Is(Animal,Tigger) } on the basis of the assumptions that Animal produces milk, its color is brown-orange and has stripes, i.e., Out'''(2)={ ¬Produce(Animal,Milk), ¬Color(Animal,Brown-Orange), ¬Has(Animal,Stripes) }

7.6.2 Business Example

Consider a 3-party business transaction that takes place in an electronic marketplace populated by software agents, as already discussed in chapters 4 and 6. Let the set {BA, SA, CA, ...} denote agents in the e-market (where BA, SA, CA denote *buyer*, *seller* and *carrier* respectively). A buyer agent (BA) communicates with a seller agent (SA) and establishes an agreement for purchasing a certain product. Consequently, the seller agent communicates with a carrier agent (CA) and establishes a separate agreement for the safe and timely delivery of goods to the buyer agent.

7.6.2.1 Dealing with information gaps (1H)

Single-agent Autonomous Reasoning

Assume that the initial set of contract norms for the agreement between BA and SA may contain two rules, R1 and R2, among others. R1 states that agent2 is obliged to deliver to agent1 if agent1 orders from agent2 and the transaction is successfully compiled. R2 states that agent1 is obliged to pay agent3 who acts on behalf of agent2 if agent1 orders from agent2, agent3 delivers the products to agent1 and agent3 is empowered to accept payment from agent1 on behalf of agent2.

R={ R1≡ IsObligedToDeliver(agent2,agent1) ← OrdersFrom(agent1,agent2) ∧ E-shopFunctionsWell,

R2 ≡ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) ← OrdersFrom(agent1,agent2) ∧ DeliversTo(agent3,agent1)
∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) }

Note that these norms are syntactically similar to the norms considered in the examples above. Thus, the corresponding lattices of candidate default formulations for each one coincide with the lattices shown in Figures 7.2 and 7.3, i.e. they are as follows:

Single-norm Lattice Structure for R1:

Level 0: {
$D1_{0,1} \equiv$
OrdersFrom(agent1,agent2), E-shopFunctionsWell
: true
/ IsObligedToDeliver(agent2,agent1) }

Level 1: {
$D1_{1,1} \equiv$
OrdersFrom(agent1,agent2)
: E-shopFunctionsWell
/ IsObligedToDeliver(agent2,agent1),

$D1_{1,2} \equiv$
E-shopFunctionsWell

: OrdersFrom(agent1,agent2)
/ IsObligedToDeliver(agent2,agent1) }

Level 2: {
$D1_{2,1}$ ≡
true
: OrdersFrom(agent1,agent2), E-shopFunctionsWell
/ IsObligedToDeliver(agent2,agent1) }

Single-norm Lattice Structure for R2:

Level 0: {
$D2_{0,1}$ ≡
OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: true
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 1: {
$D2_{1,1}$ ≡
OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)
: IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,2}$ ≡
OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: DeliversTo(agent3,agent1)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,3}$ ≡
DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: OrdersFrom(agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 2: {
$D2_{2,1}$ ≡
OrdersFrom(agent1,agent2)
: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,2}$ ≡
DeliversTo(agent3,agent1)
: OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,3}$ ≡
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 3: {
$D2_{3,1}$ ≡
true
: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Suppose that the initial knowledge for agent BA is W={ OrdersFrom(BA,SA), E-shopFunctionsWell}, that is, BA knows that it placed an order, and that the electronic transaction compiled successfully. BA employs the default formulation $D1_{0,1}$ for norm R1 and it may only infer that SA is obliged to deliver products, on the basis of this knowledge, without resorting to any assumptions. But there are cases where BA needs to perform:

- *best-guess reasoning* i.e., the agent needs to plan its *future* activities on the assumption that certain events/actions will occur, and that its partners' actions will be valid. For instance, consider that BA has just ordered successfully from SA and also knows that CA is empowered to accept payment on behalf of SA, i.e., its current knowledge is:

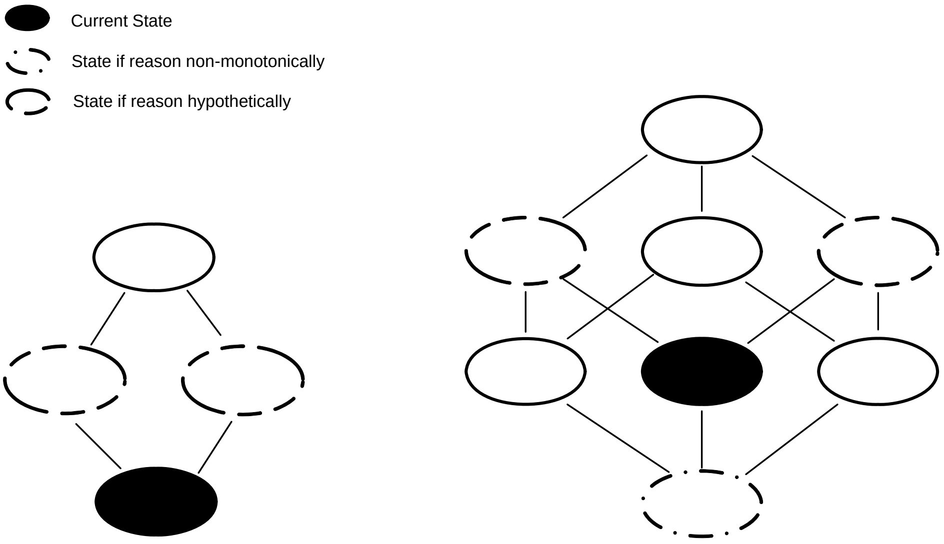
  W = In(0) = { OrdersFrom(BA,SA), E-shopFunctionsWell, IsEmpoweredToAcceptPaymentFromOnBehalfOf(CA,BA,SA) }

  In order to infer that it might find itself bearing an obligation to pay at some future time (and plan to have adequate funds available), it needs to assume that CA will deliver on time, i.e., that DeliversTo(CA,BA). In this case the agent uses the formulation $D1_{0,1}$ for norm R1 and the formulation $D2_{1,2}$ for norm R2 (Figure 7.8):

  $\Pi(2) = \{ D1_{0,1}, D2_{1,2} \}$

  In(2) = W ∪ { IsObligedToDeliver(SA,BA), IsObligedToPayOnBehalfOf(BA,CA,SA) }

  Out(2) = { ¬ DeliversTo(CA,BA) }



Lattice Structure for norm R1

Lattice Structure for norm R2

Figure 7.8 BA's position for norms R1 and R2

- *no-risk reasoning*, i.e., even though the agent may not know everything about the *past* and *present*, it may need to infer information, in order to protect itself from an undesirable situation in the future. For instance, consider that BA knows that it has just ordered successfully from SA and that CA delivered the goods to it, but it does not know explicitly whether CA is legally empowered to accept payment on behalf of SA, i.e., its current knowledge is:

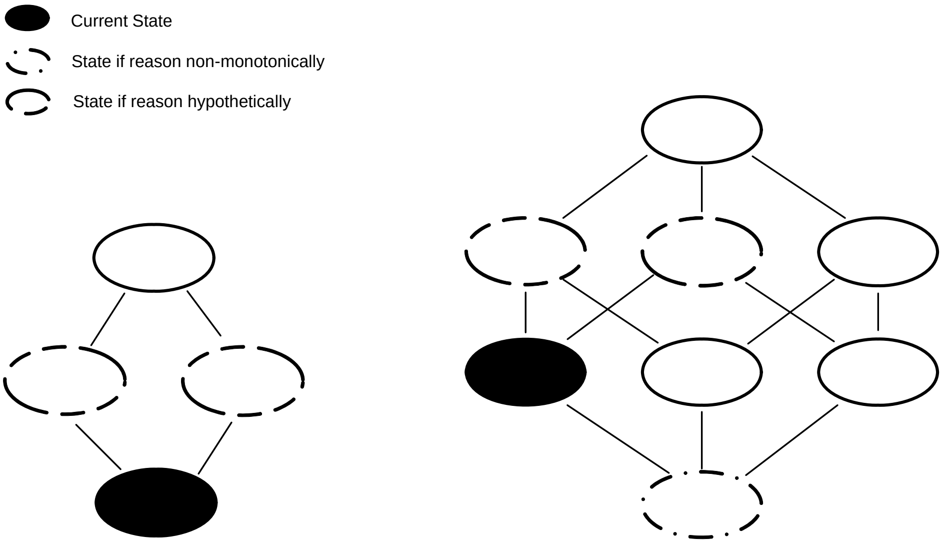
W = In(0) = { OrdersFrom(BA,SA), E-shopFunctionsWell, DeliversTo(CA,BA) }

In order to proceed and pay CA (and avoid finding itself in a situation where its payment is overdue) BA must be able to infer its obligation to pay CA, and this is possible only by resorting to the assumption that CA is legally empowered to accept payment on behalf of SA., i.e. that IsEmpoweredToAccept PaymentFromOnBehalfOf(CA,BA,SA). In this case the agent uses the default formulation $D1_{0,1}$ for norm R1 and the formulation $D2_{1,1}$ for norm R2 (Figure 7.9):

Π(2) = { $D1_{0,1}$, $D2_{1,1}$}

In(2) = W ∪ { IsObligedToDeliver(SA,BA), IsObligedToPayOnBehalfOf(BA,CA,SA) }

Out(2) = { ¬ IsEmpoweredToAccept PaymentFromOnBehalfOf(CA,BA,SA) }



Lattice Structure for norm R1

Lattice Structure for norm R2

Figure 7.9 BA's position for norms R1 and R2

Now, let us see an example where it is desirable, for some reason, to restrict the assumption space. What if we wanted our agent BA to avoid assuming that some agent is legally empowered to act as a representative for another agent in matters of payment? Then the Out(0) set (this is the set of pre-specified constraints, PC, that was mentioned earlier) must be initialized to contain the forbidden assumption:

PC=Out(0) = { IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) }

Now, it is impossible for BA to employ the above assumption, that is, it will only use the formulation $D1_{0,1}$ of norm R1, i.e.:

Π(1) = { $D1_{0,1}$}

In(1) = W ∪ { IsObligedToDeliver(SA,BA)}

Out(1) = { IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) }

The Out set may, also, be initialized to contain rules in order to restrict the assumption space. For example, CA must not be assumed to be empowered to accept payment on behalf of SA, if either SA or CA is a debtor. Then the Out(0) set must be initialized to contain the forbidden assumption, which in this case takes the form of a rule:

PC=Out(0) = { IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) ← Debtor(agent2),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) ← Debtor(agent3) }

Recall that when the Out set contains a literal, the intended semantics is that it should not be assumed. When the Out set contains a rule, the intended semantics is that, if the rule conditions hold, then the rule conclusion should neither be assumed nor concluded during inference, i.e., rules in the Out set act as constraints. Each condition in such a constraint may be known to the agent, i.e. it may be the case that it is explicitly contained in its initial knowledge, or it may be inferred during the reasoning process, or it may be assumed during the reasoning process.

Multi-agent Autonomous Reasoning

Consider now that the initial set of norms for the agreement between BA and SA contains, in addition to R1 and R2, norm R3, which states that agent2 is obliged to accept payment from agent3 on behalf of agent1 if agent3 delivers the products to agent1, agent1 pays for the products to agent3 and agent3 is empowered to accept payment from agent1 on behalf of agent2.

R = { R1 ≡ IsObligedToDeliverTo (agent2,agent1) ← OrdersFrom(agent1,agent2) ∧ E-shopFunctionsWell,

R2 ≡ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) ← OrdersFrom(agent1,agent2) ∧ DeliversTo(agent3,agent1)
∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2),

R3 ≡ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) ← DeliversTo(agent3,agent1) ∧ Pays(agent1,agent3)
∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) }

The corresponding lattices are as follows:

Single-norm Lattice Structure for R1:

Level 0: {
$D1_{0,1}$ ≡

OrdersFrom(agent1,agent2), E-shopFunctionsWell
: true
/ IsObligedToDeliver(agent2,agent1) }

Level 1: {
$D1_{1,1} \equiv$
OrdersFrom(agent1,agent2)
: E-shopFunctionsWell
/ IsObligedToDeliver(agent2,agent1),

$D1_{1,2} \equiv$
E-shopFunctionsWell
: OrdersFrom(agent1,agent2)
/ IsObligedToDeliver(agent2,agent1) }

Level 2: {
$D1_{2,1} \equiv$
true
: OrdersFrom(agent1,agent2), E-shopFunctionsWell
/ IsObligedToDeliver(agent2,agent1) }

Single-norm Lattice Structure for R2:

Level 0: {
$D2_{0,1} \equiv$
OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: true
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 1: {
$D2_{1,1} \equiv$
OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)
: IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,2} \equiv$
OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: DeliversTo(agent3,agent1)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,3} \equiv$
DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: OrdersFrom(agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 2: {
$D2_{2,1} \equiv$
OrdersFrom(agent1,agent2)
: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,2} \equiv$
DeliversTo(agent3,agent1)
: OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,3} \equiv$
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 3: {
$D2_{3,1}$ ≡
true
: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Single-norm Lattice Structure for R3:

Level 0: {
$D3_{0,1}$ ≡
DeliversTo(agent3,agent1), Pays(agent1,agent3))
∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: true
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 1: {
$D3_{1,1}$ ≡
DeliversTo(agent3,agent1), Pays(agent1,agent3)
: IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{1,2}$ ≡
DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: Pays(agent1,agent3)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{1,3}$ ≡
Pays(agent1,agent3), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: DeliversTo(agent3,agent1)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 2: {
$D3_{2,1}$ ≡
DeliversTo(agent3,agent1)
: Pays(agent1,agent3), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{2,2}$ ≡
Pays(agent1,agent3)
: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{2,3}$ ≡
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)
: DeliversTo(agent3,agent1), Pays(agent1,agent3)
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 3: {
$D3_{3,1}$ ≡
true
: DeliversTo(agent3,agent1), Pays(agent1,agent3),
IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2))
/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

In this case, agents BA and SA are subject to the same set of norms (the R set), and suppose that they possess different individual knowledge. Each of them needs to reason autonomously based on individual hypotheses, although they may share the same overall goal (e.g. to comply with the agreement).

We showed above (cf. the non-risk reasoning case) that if:

$W^{BA} = In(0)^{BA} = \{ \text{OrdersFrom(BA,SA), E-shopFunctionsWell, DeliversTo(CA,BA)} \}$,

BA needs to assume that CA is empowered to accept payment on behalf of SA in order to infer its obligation to pay.

On the other hand, SA may possess this kind of knowledge due to its separate agreement with CA, so it does not need to make such an assumption. However, it may need to employ other assumptions. For instance, let SA know that BA ordered from it successfully, that CA delivered goods to BA, and that CA is legally empowered to accept payment from BA on its behalf, i.e.

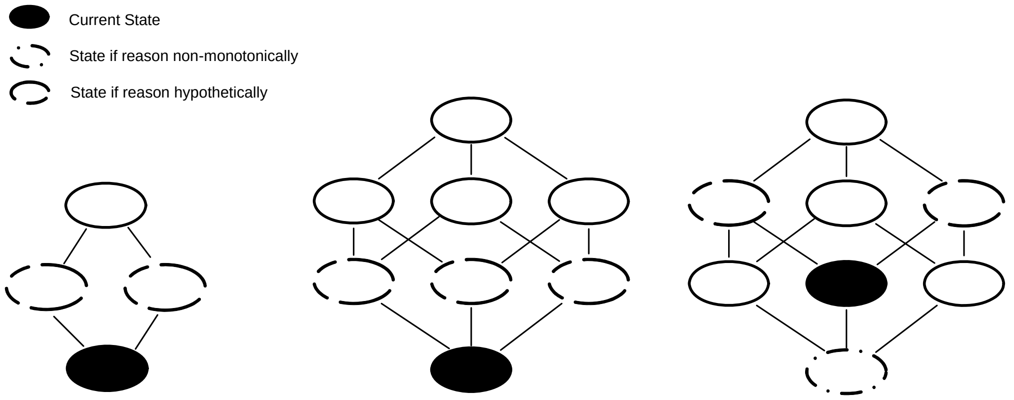
$W^{SA} = In(0)^{SA} = \{ \text{OrdersFrom(BA,SA), DeliversTo(CA,BA), E-shopFunctionsWell,}$
$\text{IsEmpoweredToAcceptPaymentFromOnBehalfOf(CA,BA,SA)} \}$

Then in order to recognize whether its partner (BA) complies with the agreement (and consider this business transaction completed) SA needs to assume that BA performs payment (Pays(BA, CA)), and this is possible with formulation $D3_{1,2}$ of norm R3 (Figure 7.10), i.e.:

$\Pi(3)^{SA} = \{ D1_{0,1}, D2_{0,1}, D3_{1,2} \}$

$In(3)^{SA} = W \cup \{ \text{IsObligedToDeliver(SA,BA), IsObligedToPayOnBehalfOf(BA,CA,SA),}$
$\text{IsObligedToAcceptPaymenFromOnBehalfOf(SA, CA,BA)} \}$

$Out(3)^{SA} = \{ \neg \text{Pays(BA, CA)} \}$



Lattice Structure for norm R1
Lattice Structure for norm R2
Lattice Structure for norm R3

Figure 7.10 SA's position for norms R1, R2 and R3

In the same way that we may have multiple agents reasoning with the same current knowledge, using different assumptions, we may impose the same or different restrictions on their assumption spaces, by appropriate initializations of their respective Out sets.

7.6.2.2 Commitment to Assumptions (2H)

Consider, again, the same set of norms for the agreement between BA and SA which contains R1, R2 and R3. We showed above (cf. the non-risk reasoning case) that if:

$W^{BA} = In(0)^{BA} = \{$ OrdersFrom(BA,SA), E-shopFunctionsWell, DeliversTo(CA,BA) $\}$

initially the BA positions itself as shown in Figure 7.11, i.e. in order to infer its obligation to pay, BA needs to assume that CA is empowered to accept payment on behalf of SA, i.e. IsEmpoweredToAcceptPaymentFromOnBehalfOf(CA,BA,SA).

$\Pi(2)^{BA} = \{ D1_{0,1}, D2_{1,1} \}$

$In(2)^{BA} = W \cup \{$ IsObligedToDeliver(SA,BA), IsObligedToPayOnBehalfOf(BA,CA,SA) $\}$

$Out(2)^{BA} = \{ \neg$ IsEmpoweredToAccept PaymentFromOnBehalfOf(CA,BA,SA) $\}$

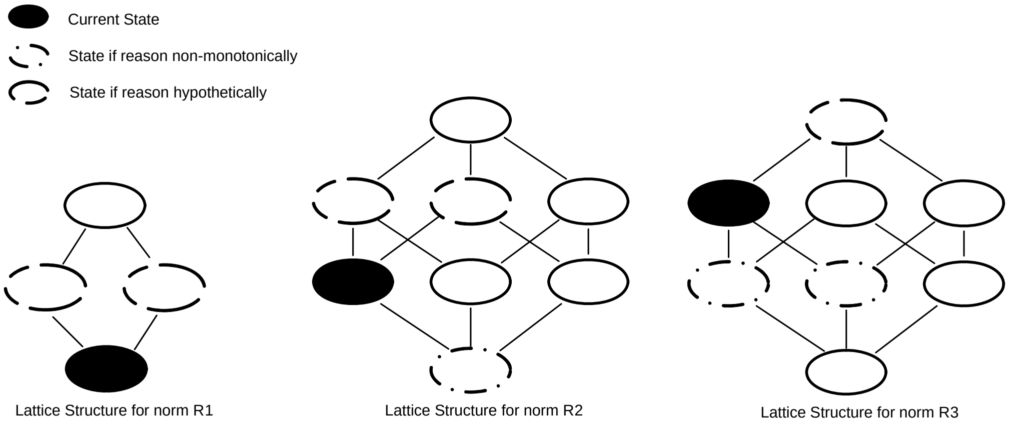


Figure 7.11 BA’s initial position for norms R1, R2 and R3

This assumption, if employed at some time point, affects BA’s subsequent inferences, and BA needs to reposition itself into the lattice structures of norms that haven’t fired yet as shown in Figure 7.11. For example, in order to infer SA’s obligation to accept its payment via its representative (CA) as norm R3 states, BA only needs either to assume payment (Pays(BA, CA) will be rendered true in its knowledge base), because it is still committed to its previous assumption about CA’s legal power, or to actually perform it. In other words, BA either needs to use the default formulation $D3_{1,2}$ or the default formulation $D3_{0,1}$, from the lattice corresponding to norm R3. For instance, the following world is computed (Figure 7.12):

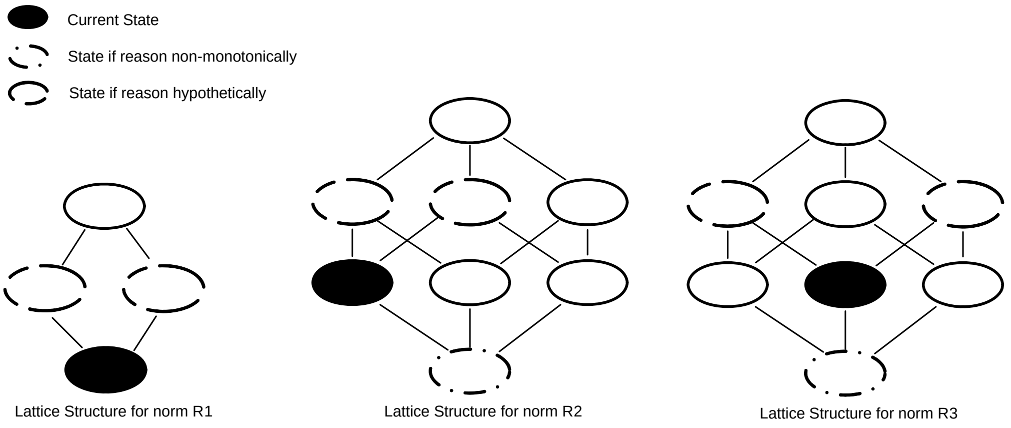


Figure 7.12 BA's reposition for norms R1, R2 and R3

$\Pi(3)^{BA}$ = { $D1_{0,1}$, $D2_{0,1}$, $D3_{1,2}$}

$In(3)^{BA}$ = W ∪ { IsObligedToDeliver(SA,BA), IsObligedToPayOnBehalfOf(BA,CA,SA) ,

IsObligedToAcceptPaymenFromOnBehalfOf(SA,CA,BA) }

$Out(3)^{BA}$ = {¬ IsEmpoweredToAccept PaymentFromOnBehalfOf(CA,BA,SA), ¬ Pays(BA, CA) }

The child-parent relations that hold between the lattice nodes indicate the path that the agent could follow.

7.6.2.3 Non-monotonic Reasoning (3H)

Consider, again the same set of norms R1, R2 and R3, for the agreement between BA and SA, but now the Out(0) set contains the following rules that restrict the assumption space:

PC=$Out(0)^{BA}$ = { IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) ← Debtor(agent2),

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2) ← Debtor(agent3) }

If the initial knowledge for agent BA is:

W^{BA} = $In(0)^{BA}$ = { OrdersFrom(BA,SA), E-shopFunctionsWell, DeliversTo(CA,BA) }

Then, as discussed above and shown in Figure 7.12, BA may infer its obligation to pay (default formulation $D2_{1,1}$) and SA's obligation to accept its payment (default formulation $D3_{1,2}$), sequentially, on the basis of the assumptions, first, that CA is empowered to accept payment on behalf of SA (the conditions of the constraints contained in the Out set are not satisfied), and second, that payment happens (again, the conditions of the constraints contained in the Out set are not satisfied).

Now imagine that at a later time point BA is informed that SA is indeed a debtor, i.e. Debtor(SA) is added to its current knowledge base. This new information affects its previously drawn conclusion. Now, BA needs to traverse the lattices downwards in order to choose an alternative formulation for the norms compatible with its current knowledge and any restrictions imposed on the assumption space. For example, BA had previously used the formulation $D2_{1,1}$ for norm R2 and the formulation $D3_{1,2}$ for norm R3 (Figure 7.12). Now that its knowledge base has expanded and contains new information, it traverses the corresponding lattices downwards from these nodes and reaches the formulations that lie at level 0 in each lattice, i.e. it retracts previously employed assumptions, along with any conclusions drawn on their basis (Figure 7.13). At this state, BA is not able to continue the incremental computation of extensions because neither norm R2 nor norm R3 apply, due to the restrictions applied, and, therefore, BA returns to the following extension:

$\Pi(1)^{BA} = \{ D1_{0,1} \}$

$In(1)^{BA} = W \cup \{ IsObligedToDeliver(SA,BA) \}$

$Out(1)^{BA} = \{ \}$

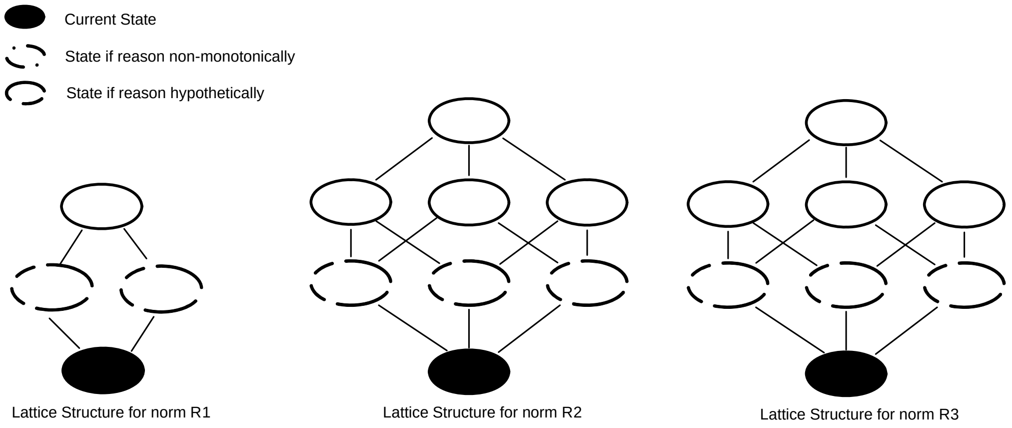


Figure 7.13 BA's reposition for norms R1, R2 and R3

7.7 Summary

The work presented in this chapter is motivated by the need for both hypothetical and non-monotonic reasoning in open normative multi-agent environments. We presented a technique that enables an agent, at any particular time point, to determine what is know and is not known and to position itself in the world, based on the explicit knowledge that it currently possesses and without resorting to proof. It turned out that the agent's knowledge/hypothesis space is, in fact, a lattice. Once the agent possesses self-knowledge, it finds out what assumptions are related to the node it occupies and may employ them in its reasoning. As the agent's knowledge changes over time, and consequently as its assumptions and conclusions need change, the agent re-positions itself on the lattice by moving on it from node to node. Moreover, we claimed that the degree of agent autonomy is related to the extent to which an agent is 'free' to make assumptions about anything it does not know about, and supported assumption identification and usage, without a priori restrictions on the agent, and without resorting to proof, which is prohibitive computationally. Instead, we showed how the variations of DfL enable agents to be more 'independent' and 'open-minded' (one might say more like humans) by self-managing their reasoning.

Research discussed in this chapter has been published or is under review in [100, 98, 86].

8 Related Work

8.1 Introduction

The previous chapters concerned with the introduction of symbolic knowledge representations and reasoning techniques for CSR of intelligent autonomous software agents that populate open computational environments. First, we were interested in identifying research questions that arise in these settings towards the need for CRS. Later, we proposed two initial approaches for assumption-based reasoning in open norm-governed environments, discussed their limitations, and, correspondingly, proposed a third approach to autonomous hypothetical non-monotonic reasoning that enable agents to manage their available knowledge, identify rational assumptions and consequently, to self-regulate their reasoning. During the last twenty years, various approaches were introduced towards these scopes. In this chapter, we organize the related work, which we found in the literature, in distinct thematic areas, discuss the motivation and special features of these approaches and provide a critical review that explains our decisions towards the adoption, adaptation or differentiation.

The rest of the chapter is organized as follows: section 8.2 discusses research approaches towards the dynamics of domains and especially the dynamics of Default Theories; section 8.3, first, reviews approaches towards assumption-based reasoning and, second, reviews approaches towards non-monotonic reasoning; section 8.4 discusses research approaches towards autonomy-oriented reasoning; and, finally, section 8.5 provides a summary.

8.2 Research on the Dynamics of Default Theories

There are many variations of Reiter’s Default Logic in the AI literature, such as the Justified DfL [163], the Constrained DfL [225], the Rational DfL [178], the Cumulative DfL [28], the Pre-constrained DfL [13, 226], the Stratified DfL [45, 46] and the Prioritized DfL [29], amongst others, but all accept a static theory and reasoning. Although DfL is a powerful approach to nonmonotonic reasoning, little work has been done towards the dynamics of Default Theories with respect the change of the theory itself. Antoniou in [15] examines the use of belief revision dynamic operators, i.e., revision and contraction, in DfL in order to reason with change. This work contrary to our approach focuses on changing facts and constraints while defaults remain unchanged. Also, Linder *et al.*, in [243], aim to place default reasoning in a dynamic context. They introduce actions to model the attempt to jump to conclusions on the basis of a set of beliefs. Beliefs are derived with the use of supernormal defaults, i.e. prerequisite-free normal default rules of the form true: C /C. Although this approach is close to our technique, we see that rule reformulation towards assumption employment, compared to belief generation via interspersed rules, affords us the ability to directly relate conclusions with employed assumptions in an argumentation-like manner which consequently facilitates nonmonotonicity.

8.3 Research on Assumption-based Reasoning and Nonmonotonic Reasoning

As already noted section 5.2, addressing the issue of reasoning with incomplete knowledge in OMAS, one must essentially address several questions. Here we remind them briefly:

1H. What assumptions are applicable to fill in information gaps?

2H. What is the relationship between assumptions and the current of future world?

3H. What happens when new information becomes available?

Work within the assumption-based reasoning community focuses on issue 1H, only. We review the main proposals to assumption-based reasoning, in order to put forward the main difference between these approaches and the work presented in this thesis. There is also work within the nonmonotonic reasoning community that is predominantly focused on questions 2H and 3H and does not address the issue of dynamic identification of assumptions satisfactorily, which we review next. Finally, since we argued that the degree of agent autonomy is related to the extend to which an agent is *free* to make assumptions, we review some of the main viewpoints of autonomy in MAS.

8.3.1 Assumption-based Reasoning

During the past twenty years various approaches to assumption-based or hypothetical reasoning have been proposed. These can be broadly grouped into:

- those that rely on a priori specification of the assumptions that can be employed during the reasoning process, i.e., those where assumption identification is static; and
- those that claim to support ad hoc identification of potentially useful assumptions during the reasoning process, that is those that purport to identify and employ assumptions dynamically.

8.3.1.1 Static Assumption-based Reasoning

Clearly, static approaches to assumption-based reasoning are not appropriate for open environments, but we discuss them here briefly.

Doyle in 1979 [72] described the representation and structure of a Truth Maintenance System (TMS). As argued, this work solves part of the belief revision problem and provides a mechanism for making assumptions. It is guided by the so called *problem of control* that is the problem of deciding on what will be the system's next inference. In other words, the agent needs an inference about which inference to make. New inferences are made by the Reasoner System (or overall Problem Solver) based on different assumptions that are statements believed without a particular reason. Consequently, different assumptions define different justified beliefs or reasoned arguments. A TMS, firstly, works as a cache by storing all inferences (justifications) ever made and, secondly, it makes any necessary revisions in the current belief set when the justifications-set, i.e. a set of justifications that represent different reasons for accepting a belief, is altered either by removing or adding a justification. In cases where a contradiction arises, a procedure, called *reasoned retraction of assumptions* is introduced. The procedure searches on each belief justification-set for at least one assumption to be removed or added in order to eliminate the contradiction. In 1986, de Kleer in [60, 61] presented a new kind of TMS that avoids certain previous pitfalls. Contrary to [72] this new approach, the so called Assumption-based Truth Maintenance System (ATMS), is based on manipulating not only justifications but assumptions too. In this way, each belief is labelled with the set of assumptions under which it holds, besides the justifications that support it. Later, in [210] and [62] respectively, Reiter and de Kleer proposed some extensions and generalizations of the ATMS that are concerned mainly with the way the system is able to manipulate clauses more general than Horn clauses. Based on the above ideas of TMS and ATMS, Kohals *et al*. in [144, 11] proposed an extension of the propositional assumption-based model with probabilities, the so called Assumption-based Evidential Language (ABEL). Consequently, hypotheses were, also, enhanced with notions such as support, quasi-support, plausibility and doubt.

Poole in [194, 195] presents Theorist that is a framework for default reasoning implemented in Prolog. Poole argues that no special logic is required for default reasoning and proposes a modification to classical logic to achieve default reasoning. He considers the simplest case of hypothetical reasoning where the user provides the form of possible assumptions in order to achieve explanation. Specifically, Theorist accepts from users a set of closed formulae called facts (F), and a set Δ of potential assumptions called possible hypotheses. A closed formula G is explainable from F and Δ if there is a set D of ground instances of Δ such that $F \cup D$ entails G, and $F \cup D$ is consistent[17]. Finally, in [196] a very interesting discussion is presented. Queries such as “What are the possible hypotheses?” and “Who makes the assumptions?” are answered based on the type of problem the agent faces, i.e. planning, diagnosis or default reasoning. Although, this approach is close to the technique that is presented in this thesis, there is a quite important difference. In Theorist, rules are taken as hypotheses, i.e. hypotheses are in a sense predefined. In our approach, we search for hypotheses among the components, i.e. conditions, of the rules. This gives us the

---

[17] As Poole points out, his assumptions are identical to Reiter’s supernormal default rules.

ability to abstract knowledge from not-knowledge and to reason in an argumentation manner.

Bondarenko *et al*. in [24] proposed an argumentation-based approach to hypothetical reasoning. This work is inspired by Dung's general argumentation framework and, specifically, it is based on the notions of *attack* and *counterattack* of the Argumentation Theory. An assumption is said to be acceptable if it is able to counterattack any other attacking set of assumptions. According to this view, definitions for admissible, complete, grounded, stable and preferred sets of assumptions were given. This fixed-assumptions framework is first introduced for logic programming, while an extension for its application to other formalisms of nonmonotonic reasoning is possible. Note that our previous comment about the EC representation viewed as a Logic Program with stable model semantics apply, here, also.

Kowalski and Sadri in [147, 148] compare the Situation Calculus [172, 209] and the EC. Both calculi are formulated as Logic Programs. As noted, the EC was intended primarily for reasoning about actual events and the Situation Calculus was primarily designed for reasoning about hypothetical actions. Thus the unification of the way both calculi handle hypothetical and actual events is proposed. Actual events are simply asserted in the knowledge base and their effects are considered valid. On the contrary, hypothetical events are also asserted in the knowledge base but nothing on their effects is stated. During the procedure of the assertion of events, integrity verification of the knowledge base is imperative. Integrity constraints are used to ensure that i) an event that happens is a possible event in the current situation and all its preconditions actually or hypothetically hold, and ii) no concurrent events are possible. Those constraints have a different role when dealing with actual or hypothetical events. In the first case constraints ensure that only possible events happen and in the second case constraints denote the context in which an assumption is possible.

Provetti in [201] also deals with the problem of actual and hypothetical actions in terms of the Situation Calculus and the EC. Contrary to the Kowalski's and Sadri's approach, that unifies both calculi, Provetti introduces: i) new predicates such as `HypHolds(fluent,situation)` to denote that a `fluent` is true in a `situation`, and ii) new ordered types of constants for denoting dates and functions denoting situations. A simple version of the EC formulated as an Extended Logic Program with answer sets semantics is presented and discussed as a tool for making assumptions on domains. Thus the new axiomatization of the EC is enhanced with new predicates and constants of the language.

Florea in [80] presents an assumption-based reasoning approach for MAS that is based on the TLI (Teoria Logica Implicita) logic. The proposed logic is a FOL enhanced with special notations that describe Reiter's original default rules and help to derive extensions. In this work, the notion of the assumption coincides with Reiter's original notion of assumption.

Tahara in [237] addresses the issue of inconsistency that arises in the knowledge base due to inconsistent hypotheses. In this work, different contradictory scenaria, comprising of facts and hypotheses, are formed based on different hypothesis sets. Contradictions may be overcome using a preference relation between hypotheses. Thus, a scenario is represented as a triple $(F, H, <)$, where F denotes the set of facts, H denotes the set of hypothesis and $<$ denotes the partial ordered preference relation that holds among hypotheses.

8.3.1.2 Dynamic Assumption-based Reasoning

The most notable approaches that fall into the second category, where assumptions are supposed to be identified and employed dynamically, include those of Cox and Pietrzykowski [52], Reichgelt and Shadbolt [205, 206], Abe [2], Pellier and Fiorino [192, 193], Jago [131] and Stamate [233]. Our work is, obviously, related mostly to this second category. However, it seems to us that assumption identification in these approaches is not truly dynamic. Before we discuss briefly each of these approaches, we make some general remarks on this issue:

Some of these approaches rely on the use of a *pre-specified pool* of assumptions, from which the agent must choose appropriate ones, whenever it identifies an information gap and needs to fill it, in order to proceed with its reasoning. A natural question that arises though, is whether it is realistic to expect that candidate assumptions can be identified in advance. It may be the case that in some application domains this is possible. However, in such cases, candidate assumption identification is not really dynamic, rather selection of an appropriate assumption from the pre-specified pool, may be carried out dynamically during the inference process. This selection though, requires deductive proof, which is notably computationally expensive. Other dynamic approaches that purport to support dynamic identification of assumptions, rely on finding appropriate assumptions in a goal-driven manner, that is, a particular conclusion that the agent wants to derive is given, and then the agent identifies the assumptions that are required, in order for this conclusion to be derivable. In some cases, such goal-driven identification of candidate assumptions requires proof. But more importantly, the problem that we perceive with purely goal-driven assumption identification is the following: although software agents, in general, are inherently goal-driven in planning their activity, their rationality (and consequently their performance measures) depends on the extent to which they are perceptive of their environment, so that they may exploit changes in it. A purely goal-driven identification of candidate assumptions does not leave much room for the agent to adapt to circumstances.

We now discuss each one of the approaches on dynamic assumption identification and usage, with some additional comments on each of them:

Cox and Pietrzykowski in [52] explore the problem of the derivation of hypotheses to explain observed events. This is equivalent to finding what assumptions together with some axioms imply a given formula. This is similar to what we refer to as no-risk reasoning, i.e. the identification and usage of assumptions about the past. They provide a method for computing causes of events that is based on linear resolution [158] and reverse Skolemization [51]. More importantly, this work proposes restrictions that guarantee that the derived assumptions are in some sense interesting for our causing events. A `cause` of an `event` is: i) minimal, ii) consistent with the knowledge base, iii) nontrivial in the sense that `cause⊃event` does not hold, and, finally, iv) basic iff every consistent cause of `cause` is trivial. In this work, the identification of assumptions is essentially goal-driven, and it requires proof, in order to establish that the observed event is implied by what is known (the axioms) and what is assumed.

Reichgelt and Shadbolt in [205, 206] present a way to analyze planning as a form of theory extension. Theory extension enables an agent to add further assumptions to its knowledge base, in order to derive potential plans towards goal achievement. This is similar to what we refer to as best-guess reasoning, i.e. the identification and usage of assumptions about the future. Their approach requires the use of a pre-specified assumption pool, where candidate assumptions are defined in advance, along with preconditions for their usage. The selection of an appropriate assumption from this pool is conducted in a goal-driven manner and requires that the preconditions associated with the assumption may be deductively proved from the knowledge base. If multiple assumptions have preconditions that are satisfied, selection amongst them is performed by checking them against pre-specified criteria, e.g. parsimony (the assumption with the fewest preconditions is selected) or generality (the more general assumption is preferred).

Abe in [2], also, deals with the problem of missing hypotheses for the explanation of an observation. He proposes a way to generate analogous hypotheses from the knowledge base when the latter lacks the necessary ones. This work extends the Clause Management System (CMS) proposed by Reiter and de Kleer [210] for abduction. A CMS, given an observation O that cannot be explained from the knowledge base KB ($KB \nvDash O$), returns as set of minimal clauses O' such that $KB \vDash O \cup O'$ and $KB \nvDash O'$. That is to say, O' is the minimal support for O with respect to KB, iff no proper subset of O' is support for O with respect to KB. Hypothesis generation is done in two distinct steps: i) using first abduction and then deduction, candidate hypotheses are searched in the knowledge base, and ii) in case where such candidate assumptions do not exist in the knowledge base, analogous hypotheses are generated by examining clauses in the knowledge base and the assumption requirements that were identified in the previous step. Hypotheses are generated ad hoc during the inference process, by exploiting predefined *analogy relationships* between clauses. This is an attractive approach, but it requires caution: in some applications it is difficult to define analogy relations between clauses, in advance; if no such definition for analogy is provided *a priori*, counterintuitive results may be produced: For instance, suppose that a buyer agent is obliged to pay a seller agent by some deadline, and that it actually proceeds to do so by cash deposit into the seller's bank account. Although the action of paying via a cash deposit is analogous to the action of paying in cash (in the sense that they have the same practical effect, the seller agent ends up possessing the required funds), the contract that regulates the exchange between the two agents may dictate that only payment in some specific form is deemed as acceptable. The two distinct forms of payment that seem analogous in terms of practical effects, may have different legal effects: one will result in the successful discharge of the buyer's obligation to pay the seller, while the other will result in a (technical) violation of this obligation.

Pellier and Fiorino in [192, 193] address Assumption-based Planning, and propose a mechanism by which an agent can produce "reasonable" conjectures, i.e. assumptions, based on its current knowledge. Any action precondition that cannot be proved from the knowledge base is considered to be a candidate assumption. A tentative plan (i.e. one that involves assumptions) becomes firm, and can be employed by the agent in order to achieve a specific goal, only when the agent can satisfy all of the conjectures, and this requires the agent to regard them as sub-goals and produce plans for them in turn. They distinguish two kinds as assumptions: i) *hypotheses* that are literals that do not belong to the current knowledge base, and ii) *fact negations* that are the negation of literals replacing facts that an agent believes. The planning mechanism is based on the Hierarchical Transition Network (HTN) [185] where an agent decomposes *non-primitive* tasks into smaller subtasks until *primitive* tasks are reached, but unlike HTN a branch and bound algorithm is used in order to compute as few conjectures as possible.

Jago in [131] uses the notion of context in making assumptions. A context is the current set of the agent beliefs. Nested contexts are used to model nested assumptions, and temporally ordered contexts are used to represent the agent's set of beliefs as it changes over time. Assumptions are not identified *a priori,* but rather during the reasoning process, either by guessing or in a goal-driven manner.

Finally, [159] and [233] adopt a numerical approach to assumption-based reasoning. Specifically, Loyer and Straccia in [159] presented the Any World Assumption, a generalization of Closed World Assumption and Open World Assumption, which allows any interpretation over the truth space to be a default assumption. The truth spaces were considered to be bilattices due to their interesting mathematical structure. They extended the many-valued logic programs rules, under stable model semantics, with computable functions that denote the truth values of their atoms and are used to compute and manipulate the truth value of the sentence itself. Stamate in [233] presented a relative approach to assumption-based reasoning. This work, also, uses multi-valued logics to express uncertainty in logic programs, but under well-founded and Kripke-Kleene semantics. Uncertainty is not only related to uncertain information but also to missing information, which is knowledge that is not derivable using the current knowledge and program rules. Atoms are assigned logical values representing various degrees of truth that may me combined and propagated by applying logic programs' rules. A three-valued logic is adopted where *true*, *false* and *unknown* are the logical values. Consequently, a *pessimistic* assumption is made whenever non derivable atoms are considered to be false. This case is identical to the Closed World Assumption. A *skeptical* assumption is made whenever non derivable atoms are considered to be unknown. This case is identical to the Open World Assumption. And finally an *optimistic* assumption is made whenever non derivable atoms are considered to be true.

8.3.2 Non-monotonic Reasoning

A temporal representation of a contract, alone, allows us to establish what factual and normative fluents are true at a given time point, through appropriate queries. Such representations, though, does not allow us to reason with incomplete knowledge dynamically. There are various approaches to reasoning with incomplete knowledge, such as the Closed World Assumption [207], Circumscription [173], Logic Programs [83, 84], and Defeasible Logic [187], and in fact these have been explored by many researchers (for example, [166, 17, 79, 211, 261, 111, 187, 191] among others).

8.3.2.1 Closed World Assumption

Under the CWA [207], an atomic formula is assumed false, unless it is explicitly known to be true. When an agent that uses a (possibly incomplete) EC contract representation, coupled with CWA, makes assumptions, these concern the *falsity* of certain formulae, rather than their truth. That is, in addressing question 1H, CWA dictates that an information gap be treated as negative information (one might call this the 'atheist' stance). In many realistic scenaria, however, it is important to be able to make assumptions about the *truth* (rather than the falsity) of certain formulae, i.e., to treat information gaps for what they are (absence of definite information) as potentially positive information (one might call this the 'agnostic' stance). For instance, suppose that a buyer agent has a rule determining whether it bears an obligation to pay a seller agent. Some of the rule conditions may be whether the buyer placed an order with the seller, whether the goods that were ordered were, in fact, delivered, and if so, whether such delivery met all possible requirements (e.g. the right quantity and quality of goods were delivered, to the right place of delivery, using the right delivery method, at the right time and so on). The buyer agent will assume that anything it does not know about explicitly is false, so if does not possess explicit information about one or more of these conditions, it will not infer that it bears the obligation to pay the seller agent through the application of the rule, and may infer, through CWA, that it does not bear such an obligation. The buyer agent, in this case, cannot exploit assumptions in order to perform no-risk or best-guess reasoning. Since any assumptions employed at some point of the inference process are not retained for future reference, there is no way to relate them to future inferences. Hence with CWA we cannot address 2H satisfactorily. When new information becomes available, possibly refuting some of the assumptions that were employed at earlier points in the inference process, there is no way to retract previously drawn conclusions, that is CWA does not address 3H satisfactorily. Of course, one may argue that such questions can be addressed, in a domain-specific manner, via the use of special purpose predicates (e.g. by recording assumptions used during the inference of each specific conclusion). However, we argued that by resorting to Default Logic we obtain a more general-purpose solution to the problem of dynamic assumption identification, which is also compatible with our common intuitions.

8.3.2.2 Circumscription

Circumscription [173] is a generalization of the CWA, and might be used instead of it (work described in [261] is in this direction). Here, special predicates are used, in order to denote abnormal (unexpected) events and effects of actions, and the inference strategy attempts to minimize abnormality. The agent possesses explicit information about abnormality, and the conclusions derived are those contained in the minimal models of the (augmented with special predicates about abnormality) knowledge base. It is now possible for the seller agent of our example above to perform best-guess and no-risk reasoning, for if it does not explicitly know that delivery was ‘abnormal’ in some sense (e.g. it never happened, or it happened at the wrong time, or the wrong quantity or quality of goods were delivered an so on), it will use its rule to infer that it has an obligation to pay. That is, the agent is able to treat an information gap as potentially positive (true) information. However, Circumscription poses some other problems, acknowledged by other researchers, as well: First, it requires that we define abnormal events, effects of actions and the like, *explicitly*, and, also, that we distinguish each abnormal individual from other individuals, explicitly [27] (page 222)]. Second, in order to decide which individuals to characterize as abnormal, we are required to anticipate the conclusions that we want to be able to derive [12] (page 149)]. Finally, in addressing 2H and 3H, Circumscription suffers from the same problems as CWA.

8.3.2.3 Logic Programs

The correspondence between Logic Programs (with stable model or answer set semantics) and default theories has been established in [156]. We might consider the EC contact representation as a (general) Logic Program, with stable model semantics [83], or as an extended Logic Program, with answer sets [84] – in fact, work described in [111] and [191] adopt the former view. In both cases, entailment is goal-driven. In stable model semantics, given a logic program LP we define its reduction LP_M with respect to a set of goal atoms M. A stable model may be computed following two steps. First, by removing all ground instances of rules contained in LP, that have in their body negative literals $\neg B$, where $B \in M$. Second, by removing all ground negative literals in the bodies of the rules that remain in LP. In answer sets semantics a similar procedure is used to compute the answer set of LP. Note that the elimination steps, described above, for the computation of a stable model or an answer set, presuppose the rejection of all rules that either contradict the set of goal atoms, or are irrelevant with the goal and, furthermore, these steps presuppose the falsity of all assumptions. The absence of an atom A from a stable model of LP is taken to signal that A is false. The absence of an atom A from an answer set of an extended LP is taken to mean that A is unknown. In the light of the comments we made earlier, when discussing CWA, it seems to us that answer set semantics are preferable to stable model semantics, for they enable us to treat information gaps in a more open-minded way (the agnostic vs. the atheist stance). What we find problematic in both cases though, for the purposes of assumption-based reasoning (and specifically in relation to question 1H), is the fact that potential assumptions can only be spotted in a goal-driven manner. The agent needs to decide *a priori* what conclusion it wants to derive, in order to be able to identify which assumptions are essential to make, in order to be able to actually derive it.

8.3.2.4 Defeasible Logic

Finally, there is another approach to default reasoning with e-contracts, namely Defeasible Logic [187], which is used by [105] and [191]. Defeasible Logic allows us to define which conclusions are retractable, by making a distinction between strict and defeasible rules. Knowing that some information is defeasible enables an agent to treat it as a potential assumption. A question that arises is whether it is possible to determine, *a priori*, during the construction of the rule base, what is and what is not defeasible. In some situations (such as the examples shown in [105] and [191]) we are, indeed, able to determine this on the basis of some specific domain information. In this case though, the agent does not discover potentially useful assumptions for itself; rather it uses an implicitly pre-specified pool of assumptions. We can see a way out of this problem: we may adopt a more general view and consider all derived conclusions as defeasible. Rule conditions that are themselves defined through defeasible rules, are defeasible. Rule conditions that are not defined through rules must be provided either as strict facts or as defeasible facts. The agent will treat anything that it does not know about as a potential assumption. However, in order to establish whether some information gap exists, it will need to carry out proof on its knowledge base (to determine which defeasible rules fire), which is computationally expensive.

8.4 Research on Autonomy-oriented Reasoning

The concept of autonomy is central to the agency and has, thus, received attention by nearly all researchers in the field of MAS. Earlier work on autonomy focused on its relation to an agent’s goals, i.e. on the extent to which an agent could choose its goals and pursue them without external intervention [78, 81, 160, 161, 19, 18], whether the latter were due to humans, other agents or the environment [21, 32, 78, 168, 255, 160, 169, 38, 40, 161, 18, 224]. More recently, autonomy is examined in relation to an agent’s reasoning process in general, i.e. in relation to the mechanisms by which an agent chooses not only its goals and course of action, but also its beliefs and desires, as well as its motivations.

Notably, Luck and d' Inverno in [160, 161] view motivations as higher-level non-derivative components that characterize the nature of agents and can be regarded as desires or preferences affecting their behaviour. This view is similar to the way we use the Out set to restrict the assumption space used by an agent.

Barber and Martin in [19, 18] use a metric to represent the degree of an agent's autonomy, and this metric is determined by an agent's goals. We believe that autonomy amounts to more than choice of goals, and that it is relevant to the reasoning process of an agent, where the latter may concern the establishment of goals, or actions, or beliefs, or norms. We claim that since in OMAS, it is unavoidable for agents to employ assumptions in their reasoning process, the extent to which an agent is autonomous depends on the extent to which the agent is independent in identifying and employing assumptions.

In [247, 246] and [50] autonomy is examined in relation to an agent's ability to choose which norms to adopt and subject itself to, and in this spirit we explore how agents may make such choices by identifying and using assumptions dynamically.

8.5 Summary

This chapter provides a critical review on various approaches, found in the literature, and the way these approaches address the issues discussed in the previous chapters 5, 6 and 7, such as the dynamics of default logic theories, assumption-based reasoning, non-monotonic reasoning, and autonomous agency.

Research discussed in this chapter has been published or is under review in [91, 90, 92, 95, 97, 100, 98, 85, 86].

9 Conflict Management

9.1 Introduction

In the previous chapters, we proposed and discussed the representation of contractual norms as default rules that are constructed dynamically from an initial temporal representation and argued why and how this representation is suitable to address various issues of interest, such as temporal reasoning, assumption-based reasoning and non-monotonic reasoning. But, all of the above issues require some sort of conflict management: it is expected that in most realistic scenaria an agent will find itself in a situation where multiple, possibly conflicting, norms apply. The agent will need some way to detect that such conflicts exist, so that subsequently, it may deploy some resolution mechanism, in order to infer what it should do at a given point in time.

The analysis, representation and management of normative conflicts have been the focus of much research in recent years, from a variety of perspectives, such as:

- Distributed Systems Management: conflicts between roles and policies. Moffett *et al.* [181], Lupu *et al.* [164] and Dunlop *et al.* [73, 74] address conflicts from the Distributed Systems Management viewpoint and view policies as a way to determine and influence management behaviour. Cholvy *et al.* [47, 48] view normative conflicts as the result of role conflict and propose a solution based on hierarchies of roles.
- Multi-agent Interaction: conflicts between commitments and BDI notions, or conflicts between goals. Broersen *et al.* [31] deal with conflicts that arise between an agent's mental attributes such as beliefs, obligations, intentions

and desires. Kowalski [146] considers normative conflicts that arise as a result of conflicting goals and presents an approach that unifies logic and decision theory. Kollingbaum *et al.* [145, 245] focus on practical reasoning agents and Virtual Organizations. They use either instantiation graphs or unification to detect conflicts.

- Legal reasoning and E-Commerce applications: conflicts between normative notions that are analysed in some nonmonotonic logic. Grosof, Governatori and Paschke, in [111, 105, 191] respectively, consider normative conflicts in e-commerce applications and assign static priorities to business rules in order to overcome conflicting situations.

In this thesis, we see that the analyses offered so far from these different perspectives should be integrated fruitfully, so as to facilitate e-commerce application development. To this end, in this chapter, we:

- identify a set of primitive patterns for normative conflicts,
- show how the conflicts identified by other researchers may be seen as instances of these primitives,
- identify some patterns of normative conflicts that have not been identified in other proposals, and finally,
- argue that the representation of contractual norms as default rules facilitates both conflict detection and resolution.

The rest of the chapter is organized as follows: sections 9.2 and 9.3 discuss the theoretical basis of normative conflicts, discuss an example scenario, which we use for illustration purposes, and presents all normative conflicts that an agent may face up; section 9.4 discusses the way the representation of contractual norms as default rules facilitates conflict detection; section 9.5 presents the way this representation is used for conflict resolution proposes; section 9.6 discusses the relation between conflicts and assumptions; and finally, section 9.7 presents related work and summarizes our research approach for conflict management.

9.2 Preliminaries

At first, legal philosophers [139, 140, 115, 157] identified normative conflicts through the impossibility-of-joint-compliance test. According to this test a normative conflict arises when and only when there is a pair of norms having opposite subject, i.e. the compliance with one norm causes the violation of the other. In the language of Deontic Logic this test is expressed with the consistency principle [125]:

$$\mathrm{Obligation}(A) \rightarrow \neg\mathrm{Obligation}(\neg A)$$

where A denotes a specific action to be done.

According to H.L.A. Hart [116] the impossibility-of-joint-compliance test is similar to an *obedience* statement which may be constructed for deontic norms. Specifically, a normative conflict arises when the obedience statements of two deontic imperatives is logically inconsistent. But, such a view is totally restrictive [116, 124], because: (i) it accepts conflicts only between deontic norms that have an imperative character (i.e. a deontic permission cannot conflict with another deontic modality[18]) and (ii) it does not deal with conflicts other than conflicts between deontic modalities (e.g. conflicts between powers, roles, policies etc). To deal with such situations, H.L.A Hart put in the place of obedience the looser concept of *conformity*.

Later, Munzer in [184] set a new basis on the normative conflicts identification problem. Specifically, he placed normative conflicts of rules in *particular occasions*. In such an occasion the norms in question must *clash* or *collide*. Similarly, Hans Kelsen used the notion of *tension*. In this way, normative conflict analysis moves away form logical contradictions. Now conflicting norms are considered as two forces operating in opposite directions whether the forces meet (a clash), or whether the forces pull in different directions (a tension) [124].

In the early nineties, Sartor's [222] and Horty's [127] work on normative conflicts set the theoretical basis for conflict management. According to Sartor [222] a conflict arises when "(possibly) valid norms establish incompatible qualifications for the same concrete state". The cornerstone in this approach is a norm set. This may be either inconsistent, if a contradiction is logically derivable from it, or potentially

---

[18] A fruitful discussion on conflicts between imperative and permissory norms and conflicts between permissions can be found in [124].

inconsistent, if it may lead to contradiction in an upcoming state. In similar spirit Horty in [127, 128] addresses moral conflicts: an agent is in moral conflict if it ought to do an action A and, at the same time, it ought to do another action B, but it is impossible to do both.

For the purposes of illustration consider the electronic marketplace, populated by software agents that establish and perform e-contracts on behalf of some real world parties, as already discussed in chapter 4. Let the set Agents={Agent1, Agent2, Agent3,…..} denote distinct identifiers for the various agents, and the set Roles={BA, SA, CA, …} denote distinct roles that agents may assume in the e-market (where BA, SA, CA denote *buyer*, *seller* and *carrier* respectively). Consider a two-party business transaction. Agent1 that acts as a buyer orders some goods from the seller Agent3. The terms of the agreement between these two agents are: Agent3 should see to it that the goods be delivered to Agent1 within 10 days from commencement (e.g., the date that the order takes place). Agent1, in turn, should see to it that payment be made within 21 days from the date it receives the goods. If Agent3 does not deliver on time, then a fixed amount is to be deducted from the original price of the goods for each day of delay and it should see to it that delivery be made by a new deadline. If Agent1 does not perform payment on time, then a fixed amount is to be added to the original price of the goods for each day of delay and it should see to it that payment be made by a new deadline.

As mentioned in chapter 4, we may adapt an informal view of the business transaction that is regulated by the agreement as a state diagram. Normative propositions of the form:

NN(agent1, role1, action, agent2, role2)

express that agent1 that acts as role1 is in legal relation NN towards agent2 that acts as role2 to perform action, where NN may be Obligation, Prohibition, Permission and legal Power. We do not employ the axiomatization of any particular system of Deontic Logic; specifically, we do not employ the axiomatization of SDL, in which these notions are modelled as operators that are inter-defined. This is because in SDL (and any system where the D[19] scheme is valid) it is not possible for an agent to bear conflicting obligations because of the D scheme. Yet, in most realistic situations, indeed in our everyday life, agents do find themselves in normative conflict. Moreover, if we were to employ

---

[19] $\neg O\bot$ where O denotes obligation.

SDL, permission, obligation and prohibition would be interdefined, and so all of the patterns we present in subsection 9.3 would be reduced to three of all six patterns (we explain this in detail in the Appendix B); thus the representation would be less distinguishing.

The first step of conflict management involves the detection of conflicts. To this end, in subsection 9.3.1, we identify primitive patterns of normative conflicts, in subsection 9.3.2 we discuss other analyses of normative conflicts and show how these may be seen as instances of the primitive patterns, and finally in subsection 9.3.3 we identify additional cases of normative conflict, which are not discussed already in the existing literature.

9.3 Normative Conflict Patterns

9.3.1 Primitive Patterns of Normative Conflicts

Table 9.1 presents a list of primitive patterns of normative conflicts. Next we discuss each one in turn.

Table 9.1 Primitive conflict patterns

IDENTIFIER	EXPLANATION	EXAMPLE
A	Conflict between a normative notion (NN) and its negation	NN(action) vs ¬NN(action) e.g. Obligation(action) vs ¬ Obligation (action)
B	Conflict between the prohibition to perform an action and the simultaneous permission or obligation to perform the same action	e.g. Prohibition(action) vs Permission(action) or Prohibition(action) vs Obligation(action)
C	Conflict between an obligation to perform action and the simultaneous obligation or permission to perform ¬action	e.g. Obligation(action) vs Obligation(¬action)
D	Conflict between the power to perform an action and	e.g. Power(action) VS Prohibition(action)

	the simultaneous prohibition to perform the same action	
E	Conflict between two obligatory distinct actions, when it is impossible to perform both at the same time	e.g. Obligation(action1) VS Obligation(action2)
F	Conflict between an obligation and the negation of the agent’s permission or power to perform it	e.g. Obligation(action) VS ¬ Permission(action) or Prohibition(action) VS ¬ Power(action)

Pattern A

Conflict between a normative notion (NN) and its negation. The general pattern is:

NN(agent1, role1, action, agent2, role2)

¬NN(agent1, role1, action, agent2, role2)

This is the common syntactical conflict that arises when an agent has contradictory knowledge. For example, an instance of this conflict arises whenever BA (Agent1) is obliged towards SA (Agent3) and simultaneously it is not obliged towards the same agent to perform payment (Payment) for a product it ordered. All other approaches, without any exception, refer to this type of conflict. In policy-based approaches, when the normative notion is obligation it is called positive-negative conflict of modalities [181].

Pattern B

Conflict between the prohibition to perform an action and the simultaneous permission or obligation to perform the same action. The general pattern is:

Sub-pattern B1: prohibition vs permission

Prohibition(agent1, role1, action, agent2, role2)

Permission(agent1, role1, action, agent2, role2)

Sub-pattern B2: prohibition vs obligation

Prohibition (agent1, role1, action, agent2, role2)

Obligation(agent1, role1, action, agent2, role2)

For example, an instance of this conflict arises whenever SA (Agent3) is permitted or obliged to perform delivery (Delivery) towards BA (Agent1) but it is also prohibited to deliver due to the some fact or assumption (i.e. that BA is a well known debtor). Once again, all previous research approaches refer to this type of conflict. In [181] and [164] these conflicts are called *conflicts between authority policies* (sub-pattern B1) and *conflict between authority and imperatival policies* (sub-pattern B2) respectively. Also, in [245] sub-pattern B1 is called *conflict*, while sub-pattern B2 is called *inconsistency*.

Pattern C

Conflict between an obligation to perform action and the simultaneous obligation or permission to perform ¬ action. The general pattern is:

Obligation(agent1, role1, action, agent2, role2)

Obligation(agent1, role1, ¬action, agent2, role2)

Here ¬ action denotes a negative action, and the issue of representing negative actions has concerned researchers (e.g. [213] regards them as actions that do not lead to the successful fulfilment of a norm). We have not developed special semantics for the representation of negative actions; we merely regard such expressions as denoting either performance of some action other than the negative one, or as idleness (non performance of any action).

For example, a conflict of this type arises whenever SA (Agent3) is obliged to perform delivery (Delivery) towards BA (Agent1) assuming that it will become regular but it is also obliged to not deliver (¬Delivery) due to some fact or assumption (i.e. that BA is a well known debtor). This case arises, also, in Lee [152] and Abrahams [3] who use the term Waive.

Pattern D

Conflict between the power to perform an action and the simultaneous prohibition to perform the same action. The general pattern is:

Power(agent1, role1, action, agent2, role2)

Prohibition(agent1, role1, action, agent2, role2)

For example, an instance of this conflict arises whenever SA (Agent3) is empowered to perform delivery (Delivery) towards BA (Agent1) but it is also prohibited to deliver due to some fact or assumption. This type of conflict is also noted in [3].

One may argue that in this case there is no conflict and, consequently, that there is no need for conflict resolution. Indeed, legal power to perform an action goes hand-in-hand with permission to exercise it, according to formal definitions of institutional power ([165, 136]). Hence, there is a conflict here, albeit some may perceive it as a conflict between permission and prohibition to exercise a certain power.

Pattern E

Conflict between two obligatory distinct actions, when it is impossible to perform both at the same time. The general pattern is:

Obligation(agent1, role1, action1, agent2, role2)
Obligation(agent1, role1, action2, agent2, role2)

This corresponds to Horty's moral dilemma [127]. For example, an instance of this conflict arises whenever SA (Agent3) is obliged to perform Delivery1 and Delivery2 towards BA (Agent1) but cannot perform both simultaneously.

Pattern F

Conflict between an obligation and the negation of the agent's permission or power to perform it. The general pattern is:

Obligation(agent1, role1, action, agent2, role2)
¬Permission/Power(agent1, role1, action, agent2, role2)

The negation of an agent's permission/power to perform an action may be explicitly derived from the agent's knowledge base (sub-pattern F1) or it may be derived from a possibly incomplete knowledge base, through the absence of explicit information (sub-pattern F2).

For example, an instance of this conflict arises whenever SA (Agent3) is obliged to perform Delivery towards BA (Agent1) assuming that it become regular but it is also not permitted to deliver due to some fact or assumption (i.e. that BA is a well known debtor).

9.3.2 Other analyses of normative conflicts

In this section we review some of the main ideas that other researchers have proposed in their analyses of normative conflict and discuss how these may be regarded as instantiations of the primitive patterns presented in the previous section. Although all the patterns discussed in this section may be regarded as special cases of the primitive patterns we introduced, they merit a separate discussion because they contain additional information that may be useful for efficient conflict resolution.

9.3.2.1 Policy-based Conflicts

Intra-policy conflicts

Dunlop *et al.* [73] refer to an *internal policy conflict*, when contradictory policies are assigned to a single role. A policy in their approach corresponds to what we call a single norm.

Consider, for example, the two distinct obligations of Agent3 (a seller) to perform delivery towards two distinct buyers (Agent1 and Agent2).

Obligation(Agent3,SA,Delivery1,Agent1,BA)

Obligation(Agent3,SA,Delivery2,Agent2,BA)

The conflict arises from the fact that contradictory policies are assigned to *Agent*3 when acting as seller. Apparently, this specific case can be mapped onto pattern E. In the same manner, other examples of this kind may be seen as instances of other primitive patterns.

Inter-policy conflicts

Dunlop *et al.* [73] refer to an *external policy conflict*, when an agent simultaneously assumes different roles that contradict “in co-existence”.

Consider, for example, that when Agent3 acts as a seller it is obliged to perform delivery towards Agent1 while when it acts as a mediator it is prohibited to perform the same action.

Obligation(Agent3,SA,Delivery,Agent1,BA)

Prohibition(Agent3,MA,Delivery,Agent1,BA)

This specific example can be mapped onto pattern B2.

9.3.2.2 Role-based Conflicts

Intra-role conflicts

Cholvy *et al.* [47], consider conflicts only among different roles. In their approach a role is defined through a set of consistent norms. We believe that for a variety of applications it is not realistic to insist on consistent role definitions, and thus we accept intra-role conflicts. Typical examples of this kind of conflict are *authority conflicts* [181] and conflicts that are related with the notion of power.

Consider the case where Agent3 who acts as a seller is both permitted and prohibited to perform delivery towards the buyer Agent1. This inconsistency may arise depending on the assumptions that are made, such as the ones presented earlier on the relation of the buyer with a well known debtor.

Permission(Agent3,SA,Delivery,Agent1,BA)

Prohibition(Agent3,SA,Delivery,Agent1,BA)

Apparently, this case can be mapped onto pattern B1.

Inter-role conflicts

Cholvy *et al.* [47] and Dunlop *et al.* [73] identify an inter-role conflict when contradictory norms arise as a result of multiple roles being assigned to an agent.

For example, when Agent3 acts as a carrier it is obligatory to perform delivery. If, at the same time, the same agent assumes the role of seller, then such delivery is not obligatory.

Obligation(Agent3,CA,Delivery,Agent1,BA)

¬Obligation(Agent3,SA,Delivery,Agent1,BA)

This case can be mapped onto pattern A.

Obviously intra-policy and intra-role conflict patterns, as well as inter-policy and inter-role conflict patterns are conceptually related. The respective authors use the terms “policy” and “role” differently, and the only reason for discussing them separately is to facilitate comparison.

9.3.2.3 Conflicts related to Interest/Duty

Conflicts of interest

Moffett *et al.* in [181] define *conflicts of interest* as the situation where “the same subject can perform management tasks on two different sets of targets”. This type of conflict can be seen as an instance of inter-role conflict or inter-policy conflict or, correspondingly, of the primitive pattern E (conflict between two obligations).

Conflicts of Duty

Moffett *et al.* in [181] and later Lupu *et al.* in [164] define *conflicts of duties* and *application specific conflicts* respectively. They refer to situations where the same agent should not be allowed to perform two distinct actions (e.g. the same agent should not be allowed both to enter a payment and to sign the payment cheque). Such conflicts may be seen as instances of inter-role conflict or inter-policy conflict or, correspondingly, of the primitive pattern E (conflict between two obligations).

9.3.2.4 Exceptions

This type of conflict arises generally in norm-governed systems. As Sartor [222] notes such conflicts emerge when “exceptions to norms state that particular norms, unambiguously identified, do not apply in a given situation”.

Consider, for example, that the buyer Agent1 who holds a discount card orders goods from the seller Agent3. Based on a policy rule the buyer gets a 10% discount due to the discount card. On the other hand, based on another policy rule the buyer should get a 20% discount because it places an order during the sales period. The described conflict is of type E:

Obligation(Agent3,SA,Discount10%,Agent1,BA)

Obligation(Agent3,SA,Discount20%,Agent1,BA)

9.3.2.5 Temporal Normative Conflicts

Dunlop *et al.* [73] present a temporal logic based approach for the detection of normative conflicts. In this section we present briefly a modification of our representation of normative relations, which takes into account the *external* time of a norm (i.e. the time at which it comes into force) and the *internal* time of a norm (i.e. the time stipulated for its satisfaction, its deadline) (cf. [166]). A formula of the form:

NN(agent1, role1, action, time2, agent2, role2, time1)

denotes that at time point time1 agent1 (acting as role1) is in legal relation NN towards agent2 (acting as role2) to perform action by time2.

Now, we may discuss normative conflicts of the types described by the primitive patterns B-F, in a temporal setting. For the purposes of illustration consider the primitive pattern E, in which the following norms are in conflict:

Obligation(Agent3,SA,Delivery1,IT1,Agent1,BA,ET1)

Obligation(Agent3,SA,Delivery2,IT2,Agent2,BA,ET2)

where IT1, and ET1 are the internal/external time points for the first norm, and IT2, ET2 are the internal/external time points for the second norm. Temporally well formed norms are those whose internal time is subsequent to their external time, so each normative proposition corresponds to an interval; the intervals for the example we use here are ΔT1=[ET1, IT1] and ΔT2=[ET2, IT2].

A conflict arises in the following situations (these are depicted as shadowed in Figure 9.1):

- $ET1 = ET2$ and $IT1 = IT2$: when $\Delta T1$ coincides with $\Delta T2$ (Figure 9.1 (a)).
- $ET1 \leq ET2 < IT2 \leq IT1$: when $\Delta T1$ fully overlaps $\Delta T2$ (Figure 9.1 (b), 9.1 (c) and 9.1 (d)).
- $ET1 < ET2 < IT1 < IT2$: when $\Delta T1$ partially overlaps $\Delta T2$ (Figure 9.1 (e)).
- $IT1 = ET2$: when $\Delta T1$ meets $\Delta T2$ (Figure 9.1 (f)). This conflict holds only at time point $IT1 = ET2$.

Note that for completeness, one should also consider the symmetrical cases.

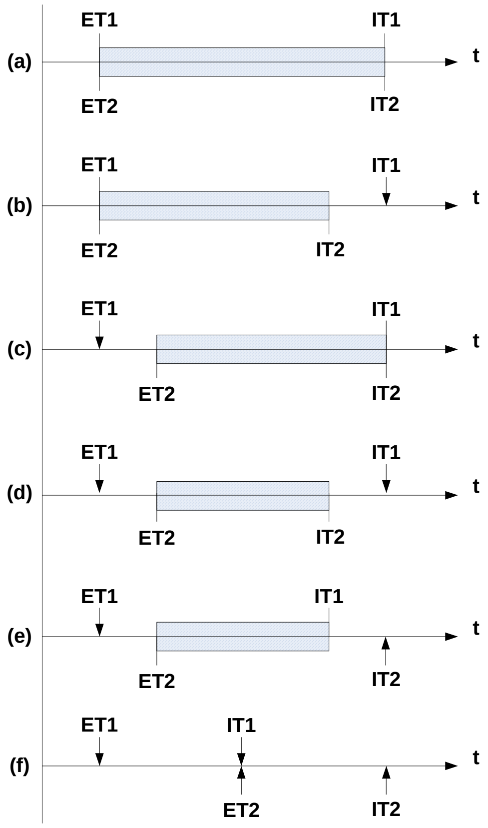


Figure 9.1 Time interval-based conflicts

9.3.3 Additional Patterns

Here are some additional cases of normative conflicts, which are not discussed already in the existing literature. We mention them separately because, although they may be reduced to the primitive patterns, there is additional information that may be exploited to facilitate conflict resolution.

9.3.3.1 Type of action-based conflicts

A common feature of e-contracts is the so called Contrary-to-Duty structures [200]. An agent’s contractual obligations may be divided in two types. *Prima facie* obligations that concern the performance of actions that are in principle stipulated by the agreement and secondary obligations that concern the performance of reparatory actions; the latter apply only when violations of *prima facie* obligations happen.

An agent may, thus, bear two distinct obligations (for instance of the kind described in pattern E or in the intra-policy conflict), where one of them is primary and the other is secondary (as the result of a violation). This qualification may be helpful in conflict resolution, as will be discussed in section 4. The general pattern is:

Obligation(agent1, role1, action, agent2, role2)

Obligation(agent1, role1, reparatoryaction, agent3, role3)

9.3.3.2 Agreement-based conflicts

An agent may find itself in a conflicting state because it is engaged in multiple contracts. For instance a seller may be obliged to perform two distinct deliveries to two distinct buyers as dictated by two distinct agreements. This situation may be regarded as the generalization of the intra-policy conflict and, consequently, of the pattern E. But, in this specific case the important information is the distinction between the contracts. The additional information that the two norms stem from two agreements, may be exploited for the purposes of conflict resolution. The general pattern is:

Obligation(contract1, agent1, role1, action1, agent2, role2)

Obligation(contract2, agent1, role1, action2, agent3, role3)

Normative propositions of the form:

NN(contract, agent1, role1, action, agent2, role2)

express that according to contract, agent1 that acts as role1 is in legal relation NN towards agent2 that acts as role2 to perform action.

Note that this conflict pattern is different form the one presented in [117]. The key notion here is the different contracts an agent has to comply with. Different contracts may be established towards different agents or even towards the same agent.

9.3.3.3 Conflicts between assumptions and knowledge

A conflict may arise not only as a result of an agent’s explicit knowledge but also between its knowledge and its current assumptions or even between distinct assumptions.

For example, consider that Agent3 (a seller) assuming that it is permitted to perform delivery towards Agent1 (a buyer) entails that is obliged to perform that delivery. Moreover, the same agent assuming that Agent1 is related to a well known debtor entails that it is prohibited to perform that delivery. Note that in this scenario the prohibition that derives from the second rule contradicts not only with obligation that derives from the first rule, but also with the assumption of the first rule (permission).

9.4 Conflict Detection

9.4.1 Conflict Detection with Default Logic

We represent the norms of an agreement as default rules. For instance, the following default rule expresses that if an order from Agent1 (acting as a buyer) towards Agent3 (acting as a seller) holds, and it is consistent to assume that Agent1 will become a regular client, then we may infer that Agent3 is legally obliged towards Agent1 to perform delivery:

$$\frac{\begin{array}{c}\text{Order(Agent1, BA, Agent3, SA)}\\ :\\ \text{BecomeRegularClient(Agent1)}\end{array}}{\text{Obligation(Agent3, SA, Delivery, Agent1, BA)}}$$

An agent that engages in some agreement-governed transaction essentially reasons with a default theory. At each time point during the business transaction the agent attempts to compute the extensions of its current DfT. The DfT contract representation allows us to detect normative conflicts by examining extensions, which are essentially possible worlds. A conflict may be detected either between multiple courses of extensions or between the same course of extensions, i.e. between some extension and the current knowledge of the agent. Where a conflict is detected between multiple courses, the latter represent alternative courses of futures for the agent; let us call these *inter-extension* conflicts. Where a conflict is detected between an extension and the current knowledge of the agent; let us call these *intra-extension* conflicts. The role of conflict detection is, thus, to assist an agent to choose a course of action so that normative violations may be *predicted* and *avoided*.

To illustrate this interpretation consider the following DfT (*W, D*):

W = { P1, P2, P3 }

D={D1 ≡ P1:J1/C1, D2 ≡ P2:J2/C2}

where the following cases arise:

- Case 1: Assume that P3 and C2 represent a conflict pattern of type A (between a normative notion and its negation). The only possible extension is In(1)={P1, P2, P3, C1}, Out(1)={¬J1} for Π(1)={D1}. In this case an intra-extension conflict is avoided (Figure 9.2). The discontinuous arrow between the nodes denotes that step is not feasible due to normative conflicts that arise.

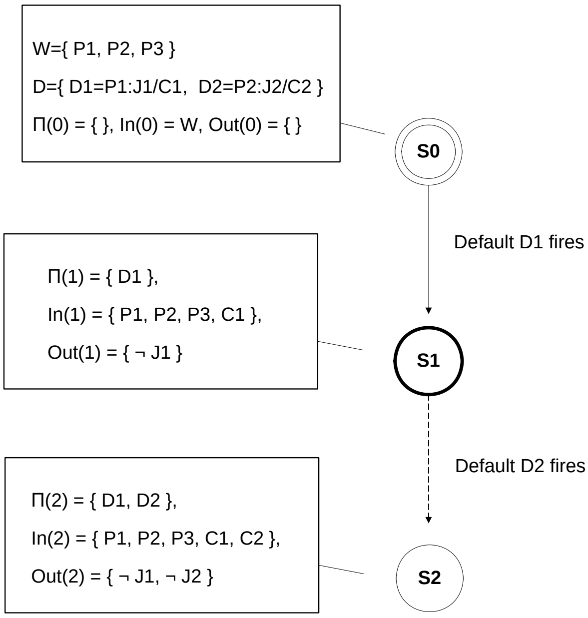


Figure 9.2 Detection of Normative Conflicts - Case 1

- Case 2: assume that P3 and C2 represent a conflict pattern other than type A. A possible extension is In(2)={P1, P2, P3, C1, C2}, Out(2)={¬J1, ¬J2} for Π(2)={D1, D2}. In this case an intra-extension conflict occurs among P3 and C2 (Figure 9.3).

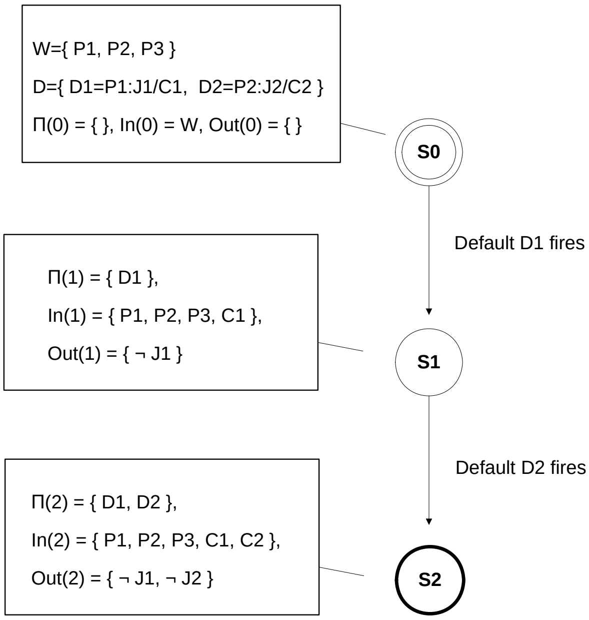


Figure 9.3 Detection of Normative Conflicts - Case 2

- <u>Case 3</u>: Assume that C1 and C2 represent a conflict pattern of type A (between a normative notion and is negation). There are two possible extensions, i.e. In(1)={P1, P2, P3, C1}, Out(1)={¬J1} for Π(1)={D1} or In(1)={P1, P2, P3, C2}, Out(1)={¬J2} for Π(1)={D2}. In this case an inter-extension conflict is detected and arises between C1 and C2 and a course of action should be followed. This is a general dilemma for an agent to choose which rule, D1 or D2, to apply (Figure 9.4). The discontinuous arrows between nodes denote that these steps are not feasible due to normative conflicts that arise (intra-extension conflicts are avoided).
- <u>Case 4</u>: Assume that C1 and C2 represent a conflict pattern other than type A. Similarly, there are two possible courses, i.e. In(1)={P1, P2, P3, C1}, Out(1)={¬J1} for Π(1)={D1} or In(1)={P1, P2, P3, C2}, Out(1)={¬J2} for Π(2)={D2}. So far, this case is identical to case 3, that an inter-extension conflict is detected and arises between C1 and C2. In the next step of the process Π the new extensions are In(2)={P1, P2, P3, C1, C2}, Out(2)={¬J1, ¬J2} for Π(2)={D1, D2} or In(2)={P1, P2, P3, C2, C1}, Out(2)={¬J2, ¬J1} for Π(2)={D2, D1}, respectively. Of course, those extensions seem identical, but they record different courses of actions. Now, intra-extension conflicts between C1 and C2 occur (Figure 9.5).

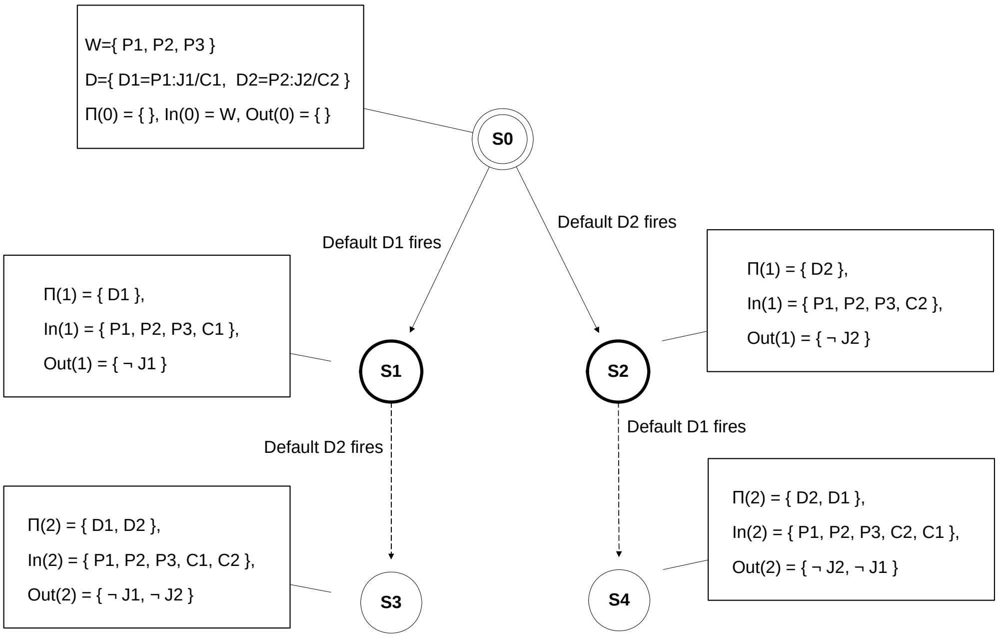


Figure 9.4 Detection of Normative Conflicts - Case 3

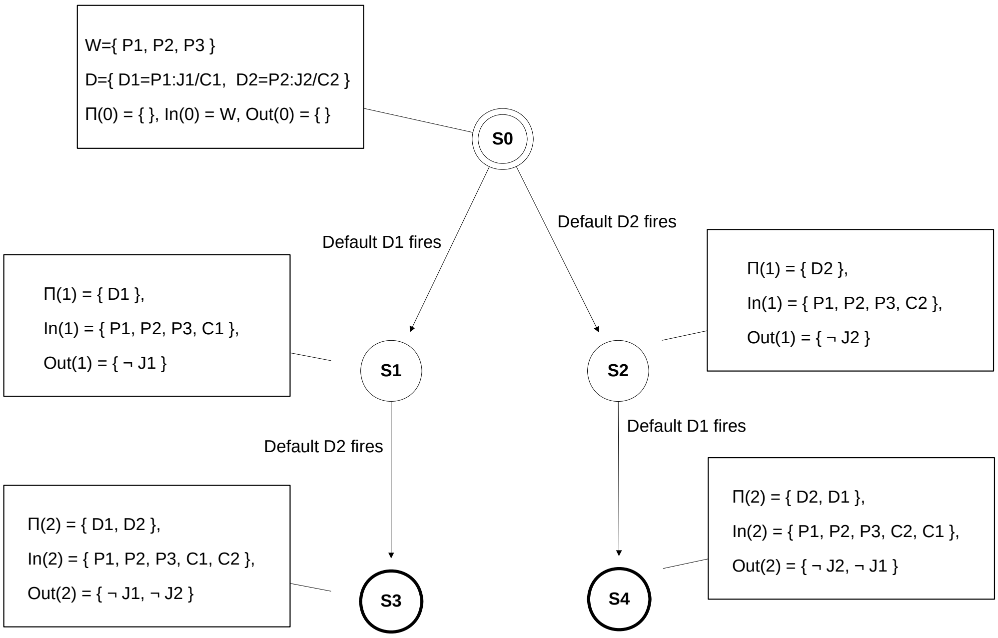


Figure 9.5 Detection of Normative Conflicts - Case 4

Note that for completeness, one should also consider the analogous cases regarding hypotheses J1 and J2, as mentioned to the last conflict pattern, i.e. conflicts between assumptions and knowledge. Of course, more complicated cases may also be defined with the combination of more conflicting patterns in the same example.

On the whole, we could record that:

- in case 1 the conflict is syntactic and it never actually arises or occurs due to the property of DfL to preserve consistency. The role that DfL plays is that of the prevention of conflicts.
- in case 2 the conflict is semantic and actual. The agent finds itself in a conflicting state.
- in case 3 the conflict is syntactic and actual in means that the agent faces a dilemma. Although the agent is not and will not find itself in a conflicting state, due to the property of DfL to maintain consistency, it faces a query which course of action to choose.
- in case 4 the conflict is semantic and actual. Initially the agent is not in a conflicting state but has a dilemma as in the previous case. Finally, when both defaults apply, sequentially, the agent ends in a conflicting state.

Note that by *syntactic* we refer to the conflict pattern A, while by *semantic* we mean all other conflict patterns. Moreover, we relate the notions of *actual conflict* and *non-actual conflict* with the way DfL entails new knowledge, its property to maintain consistency and the situation where an agent finds itself in a situation where a general query of the form “What should I do?” arises. Specifically, a non-actual conflict is the one explained in case 1, where contrary to other approaches where this type of conflict arises, here it is avoided and the agent never addresses a query. Additionally, the notion of actual conflict has dual semantics. We identify it not only with the situation where an agent faces a query such as “Which norm should I apply?”, but also with the situation where an agent has semantically contradictory knowledge about the current world and queries itself “Which normative relation should I comply with?” or “Which normative relation should I concede?”. Dual semantics of an actual conflict arises due to the DfL property to entail possible world views by computing extensions. The computation of extensions is like having a short run look in possible worlds, while only the by accepting a single extension the agent finds itself in this world. This characteristic gives agent the ability to address conflicts in a pro-active way. We discuss this issue in detail in the subsection 9.4.2.

In what follows, we present examples of primitive, as well as, other patterns of conflicts by representing agreement rules as defaults. Moreover, through these examples we illustrate the way DfL facilitates conflict detection by searching and examining clauses among sets.

9.4.2 Patterns of Normative Conflicts Represented and Detected via Defaults

Pattern A

Conflict between a normative notion (NN) and its negation. This type of conflict never actually occurs in our representation, where norms are represented as defaults, because the derivation of extensions preserves consistency, i.e. this intra-extension conflict is avoided. It may, however, arise as an inter-extension conflict, when multiple extensions are computed as the result of the application of norms that infer conflicting consequences.

Consider the following DfT where[20]:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}{\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{WellKnownDebtor(Agent1)}}{\neg\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}$$

}

The first default denotes that if an order from Agent1 (acting as buyer) towards Agent3 (acting as seller) holds then we may infer that Agent3 is obliged to perform delivery if it is consistent to assume so. Similarly, the second default expresses that if an order from Agent1 towards Agent3 holds, and it is consistent to assume that Agent3 is related to a well known debtor, then we may infer that Agent3 is not obliged to perform delivery towards Agent1. There are two possible extensions, i.e.

---

[20] Note that special terms, such as WellKnownDebtor(agent), BecomeRegularClient(agent) or IsRegularClient(agent) among others, are used only for the purposes of illustration and are not binding to the characterization of domain-independent conflict patterns.

In(1)={ Order(Agent1,BA,Agent3,SA), Obligation(Agent3,SA,Delivery,Agent1,BA)}

and

In(1)={ Order(Agent1,BA,Agent3,SA), ¬Obligation(Agent3,SA,Delivery,Agent1,BA)}

Agent3 should choose a course of action.

<u>Pattern B</u>

Conflict between the prohibition to perform an action and the simultaneous permission or obligation to perform the same action.

Consider, for instance, the following default theory (W,D) where:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ : \\ \text{WellKnownDebtor(Agent1)}\end{array}}{\text{Prohibition(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ : \\ \text{Permission(Agent3,SA,Delivery,Agent1,BA)}\end{array}}{\text{Permission(Agent3,SA,Delivery,Agent1,BA)}} \}$$

The first default denotes that if an order from Agent1 (acting as buyer) towards Agent3 (acting as seller) holds, and it is consistent to assume that Agent1 is related to a well known debtor then we may infer that Agent3 is prohibited to perform delivery. Similarly, the second default expresses that if an order from Agent1 towards Agent3 holds, and it is consistent to assume that Agent3 is permitted to perform delivery, then we may infer that Agent3 is permitted to perform delivery towards Agent1. Finally, Agent3 may find itself in a conflicting state (sub-pattern B1) after applying the two defaults sequentially (intra-extension conflict). The computed extension is:

In(2)={ Order(Agent1,BA,Agent3,SA), Permission(Agent3,SA,Delivery,Agent1,BA), Prohibition(Agent3,SA,Delivery,Agent1,BA)}

In the same spirit, consider the following DfT:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ : \\ \text{WellKnownDebtor(Agent1)}\end{array}}{\text{Prohibition(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ : \\ \text{Obligation(Agent3,SA,Delivery,Agent1,BA)}\end{array}}{\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}} \}$$

Once again Agent3 will end up in a conflicting state (sub-pattern B2). The corresponding extension is:

In(2)={ Order(Agent1,BA,Agent3,SA), Prohibition(Agent3,SA,Delivery,Agent1,BA), Obligation(Agent3,SA,Delivery,Agent1,BA)}

Pattern C

Conflict between an obligation to perform action and the simultaneous obligation or permission to perform ¬ action.

For example consider the following DfT where:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{BecomeRegularClient(Agent1)}}{\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{WellKnownDebtor(Agent1)}}{\text{Obligation(Agent3,SA,}\neg\text{Delivery,Agent1,BA)}} \quad \}$$

Finally, Agent3 may find itself in a conflicting state after applying the two defaults sequentially (intra-extension conflict). The computed extension is:

In(2)={ Order(Agent1,BA,Agent3,SA), Obligation(Agent3,SA,Delivery,Agent1,BA), Obligation(Agent3,SA,¬Delivery,Agent1,BA)}

Pattern D

Conflict between the power to perform an action and the simultaneous prohibition to perform the same action.

For instance consider the following DfT:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{Power(Agent3,SA,Delivery,Agent1,BA)}}{\text{Power(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\text{Order(Agent1,BA,Agent3,SA)} : \text{WellKnownDebtor(Agent1)}}{\text{Prohibition(Agent3,SA,Delivery,Agent1,BA)}} \quad \}$$

Once again, Agent3 may end up in a conflicting state after applying the two defaults sequentially (intra-extension conflict). The computed extension is:

In(2)={ Order(Agent1,BA,Agent3,SA), Power(Agent3,SA,Delivery,Agent1,BA), Prohibition(Agent3,SA,¬Delivery,Agent1,BA)}

Pattern E

Conflict between two obligatory distinct actions, where it is impossible to do both at the same time.

For instance consider the following DfT where:

W={Order(Agent1,BA,Agent3,SA), Order(Agent2,BA,Agent3,SA), no simultaneous performance of actions is possible}

and D={

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \text{BecomeRegularClient(Agent1)}\end{array}}{\text{Obligation(Agent3,SA,Delivery1,Agent1,BA)}}$$

$$\frac{\begin{array}{c}\text{Order(Agent2,BA,Agent3,SA)}\\ :\\ \text{IsRegularClient(Agent1)}\end{array}}{\text{Obligation(Agent3,SA,Delivery2,Agent2,BA)}} \quad \}$$

After applying the two defaults sequentially, Agent3 bears two obligations that cannot be simultaneously satisfied. The corresponding extension is:

In(2)={ Order(Agent1,BA,Agent3,SA) , no simultaneous performance of actions is possible,

Obligation(Agent3,SA,Delivery1,Agent2,BA), Obligation(Agent3,SA,Delivery2,Agent2,BA)}

Pattern F

Conflict between an obligation and the negation of the agent’s permission or power to perform it.

For instance consider the following DfT where:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \text{BecomeRegularClient(Agent1)}\end{array}}{\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \text{WellKnownDebtor(Agent1)}\end{array}}{\neg\text{Permission(Agent3,SA,Delivery,Agent1,BA)}} \quad \}$$

Finally, Agent3 may find itself in a conflicting state after applying the two defaults sequentially (intra-extension conflict). The computed extension is:

In(2)={ Order(Agent1,BA,Agent3,SA), Obligation(Agent3,SA,Delivery,Agent1,BA), ¬Permission(Agent3,SA,Delivery,Agent1,BA)}

The negation of an agent’s permission to perform an action derives from the agent’s knowledge base (sub-pattern F1). In the case of an incomplete knowledge base, it may derive via an assumption (sub-pattern F2).

To illustrate this consider a DfT that contains the first of the defaults above and in place of the second, the following:

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \neg\text{Permission(Agent3,SA,Delivery,Agent1,BA)}\end{array}}{\neg\text{Permission(Agent3,SA,Delivery,Agent1,BA)}}$$

If the agent's knowledge base does not contain an explicit permission, then the justification of this default will be satisfied, and hence its conclusion will be drawn. Once again, Agent3 may end up in a conflicting state after applying the two defaults sequentially. The extension is as computed above.

Up to now we have only referred to primitive patterns. Regarding all other conflicts, similar examples and representations may be recorded. We pass over these cases because all of them correspond to the primitive patterns. We will only refer to the last conflict pattern that concerns hypotheses and current knowledge.

Conflicts between assumptions and knowledge.

As, already, mentioned a conflict may arise not only as a result of an agent's explicit knowledge but also between its knowledge and its current assumptions or between its assumptions.

For example, consider the following DfT:

W={Order(Agent1,BA,Agent3,SA)}

and D={

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \text{Permission(Agent3,SA,Delivery,Agent1,BA)}\end{array}}{\text{Obligation(Agent3,SA,Delivery,Agent1,BA)}}$$

$$\frac{\begin{array}{c}\text{Order(Agent1,BA,Agent3,SA)}\\ :\\ \text{WellKnownDebtor(Agent1)}\end{array}}{\text{Prohibition(Agent3,SA,Delivery,Agent1,BA)}} \quad \}$$

As in previous examples, Agent3 will find itself in a conflicting state after applying the two defaults sequentially. The computed sets for Π(2)={D1, D2} are:

In(2)={ Order(Agent1,BA,Agent3,SA), Obligation(Agent3,SA,Delivery,Agent1,BA), Prohibition(Agent3,SA,Delivery,Agent1,BA)}

and

Out(2)={ ¬Permission(Agent3,SA,Delivery,Agent1,BA), ¬WellKnownDebtor(Agent1)}

Note that in this case the prohibition that derives from the second default contradicts not only with obligation that derives from the first default, but also with the assumption of the first default (permission). Of course, due to the fact that the Out()

set contains what should not become true in the knowledge base, i.e. the negation of the assumptions made, when conflicts are related with assumptions (justifications) a correct form of a conflicting formula should be searched for.

One may argue that following Reiter's original computation of extensions within DfT [208] we may compute possible world models that are counter-intuitive because contradictory assumptions are possible (e.g. by assuming that the buyer (Agent1) is related to a well known debtor (WellKnownDebtor(Agent1)) or it is not (WellKnownDebtor(Agent1)). For example, consider the following DfT (W, D):

$W=\emptyset$

$D=\{ D1 \equiv true:J/C1,\ D2 \equiv true:\neg J/C2 \}$

According to the original computation of extensions an extension including C1 and C2, based on contradictory assumptions (J and ¬J), is computed. This view of extensions, separated from assumptions, as possible world models is clearly undesirable. Thus a technique to preserve consistency and detect conflicts over justifications is also imperative. The proposed representation of contract rules as defaults enables us to achieve this aim in two ways. Either by searching for conflict patterns in the Out() set, also, besides the In() set, or by employing Constrained Default Logic [225]. The possible world model that the agent infers, for a Constrained Default Theory, is the consistent set $In(i) \cup \neg Out(i)$. This is tantamount to saying that the possible world models inferred by the agent contain, besides previous knowledge, both the consequents and the assumptions of the applied defaults.

9.5 Conflict Resolution

Various approaches for conflict resolution have been proposed in the last decade. It seems that the common ground for most of them is the ascription of priorities to norms [222, 145], policies [181, 164, 74], roles [48], based on some criterion, which may be domain dependent or independent. Belief revision [222], goal reduction and decision based on utility [146], conflicting provision voidance [3] and instantiation graphs/unification [145, 245] are some of the other proposed strategies for conflict resolution.

According to the dual semantics of the notion *actual conflict* as we presented it in the subsection 9.4.1, conflict resolution needs to be done in such a way that facilitates answering both queries “Which norm should I apply?” and “Which normative relation should I comply with?”. Conflict resolution in DfL may be performed using Brewka’s [30] proposal that enables us to define and apply priorities on default rules dynamically.

Brewka in [30] defined a prioritized DfT as a triple $(W, D, name)$, where $name$ is a function that assigns names to default rules D. The extension of a PDfT is derived in the same way as in a DfT. As noted in subsection 3.3.2.5, priorities over defaults can either define preference on extensions that are, eventually, preferred transaction plans when dealing with the query “Which norm should I apply?”, or define preference on normative relations that already hold, that is an answer to the query “Which normative relation should I concede?” given based on the priorities of defaults that entailed these normative relations.

To illustrate this interpretation consider the DfT of the previous example of section 9.4.1 where (W, D):

$W=\{P1, P2, P3\}$

$D=\{ D1 \equiv P1{:}J1/C1,\ D2 \equiv P2{:}J2/C2 \}$

where $D1$ has priority over $D2$. According to case 3 (section 9.4.1) where $C1$ and $C2$ represent a conflict pattern of type A (between a normative notion and is negation) and based on the priority relation the agent may end up choosing the first extension due to the fact that $D1$ has priority over $D2$. In a similar way, according to case 4 (section 9.4.1) where $C1$ and $C2$ represent a conflict pattern other than type A and after applying both defaults ($\Pi=\{D1, D2\}$) the agent may decide to comply with normative relation $C1$ due to the fact that it is the consequent of the default that overrides all other defaults.

What makes PDfTs really useful is that the ascription of priorities to default rules may, itself, be done dynamically. Using dynamic priorities, we generate preferred extensions, each of which indicates a distinct transaction plan to follow. Specifically, priorities amongst ground defaults may be defined dynamically either by making different assumptions or by specifying domain-dependent criteria. The general pattern for ascribing priorities dynamically takes the form of a default rule:

$$\frac{\begin{array}{c} Rule(d1,v1) \wedge Rule(d2,v2) \wedge criterion \\ : \\ assumptions \end{array}}{d1<d2}$$

Here $d1$, $d2$ are variables that denote names of ground defaults; $Rule(d1 ,v1)$ denotes a ground default $d1$ and its set of entities of interest $v1$. The intended interpretation of this rule is: if two defaults $d1$ and $d2$ apply and some criterion is satisfied between entities of interest, then $d1$ takes priority over $d2$, if certain assumptions may consistently be made. We see that the criterion of interest may also be a consistent assumption.

Table 9.2 Resolution Strategies for Normative Conflict Patterns

Strategy	Conflict Pattern	Criterion
Hierarchy	• Primitive normative conflicts • Inter-policy/inter-role conflicts • Conflict of duties/interests • Agreement-based conflicts • Type of action-based conflicts • Conflicts between assumptions and knowledge	e.g. prohibitions overrides all other normative notions, explicit knowledge overrules assumptions, obligations of reparatory actions should be met first, a regular client has priority
Temporality	• Time interval-based conflicts	e.g. the oldest obligation takes priority, or the shortest deadline takes priority
Specificity	• Exceptions	e.g. the most specific rule overrides all others

Three general strategies for defining such criteria have been discussed in the literature, namely hierarchies of entities of interest, time and specificity of norms. Table 9.2 summarizes one possible way in which the patterns of normative conflicts that we discussed may be used by specific strategies. Given a particular normative conflict, different resolution strategies may be applied depending on our specific criterion of interest.

For instance, consider the case where two norms ($D1$ and $D2$) that define conflicting obligations for $Agent3$ are active (Figure 9.6). The first one is initiated at $ET1$ and it is towards buyer $Agent1$ who is a regular client. It sets an obligation to perform delivery until $IT1$. The second one is towards buyer $Agent2$, it is initiated at $ET2$ and defines a reparatory obligation to perform delivery until $IT2$. The relation between time points is as follows: $ET1 < ET2 < IT2 < IT1$. There is information that can be used to determine different conflict resolution criteria. The strategy of temporality based on external time may give priority to $D1$ as it was initiated first. On the other hand, temporality based on internal time may give priority to D2 since it has a shorter deadline. Another alternative, using the strategy of hierarchy is to give precedence to D1, because Agent1, as a regular client, takes precedence over Agent2. Or, we may give precedence to D2, because it concerns a reparatory action, if we choose to assign higher priority to secondary norms over primary ones. It should be clear that various combinations of these criteria may also be defined based on the agent's current knowledge and the assumptions it makes.

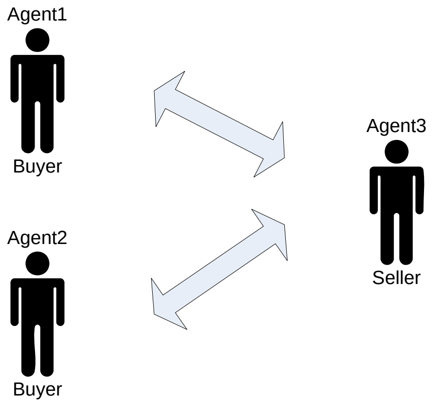


Figure 9.6 Buyer - Buyer - Seller Example Scenario

A fourth general strategy is also applicable by exploiting the fact that we employ DfL, i.e. an agent may ascribe priorities between default rules, based on the number or the type of assumptions used. In [97] we presented a technique where, under incomplete knowledge, initial contract rules are reformulated appropriately when needed, and thus inferencing is possible on a totally hypothetical basis via the dynamic identification and employment of appropriate candidate assumptions. Various rule formulations derive by populating appropriately the P and J sets on the basis of currently available knowledge. All possible formulations are organized in a hierarchical structure, where the binary relation that causes them to be partially ordered is the number of assumptions employed. In fact, this separation of inference rules may have a useful role during conflict resolution. For instance a cautious agent may give priority to rules with fewer assumptions, while a risky agent may give priority to rules with more assumptions. In this case priority ascription is based on a quantitative analysis of assumptions used. On the other hand a qualitative analysis is also useful. For instance consider the DfT as shown in conflict type F. In this case different consequents are inferred depending on different assumptions. A cautious agent may give priority to rules with semantically positive assumptions that infer deterrent consequents, while a risky agent may give priority to rules with semantically negative assumptions that infer incitement consequents.

Finally, a fifth general strategy, which is related to DfL feature to compute possible world views and a *utility factor*, is also applicable. Utility factors usually are quantitative notions such as profit/loss (amount of money earned/lost), number of obligations that arise/fulfilled or number of new conflicts. Using DfL the agent is able to derive all possible extensions and apply utility factor estimation on their conclusions. In this way a short-term analysis takes place that leads the agent to the dilemma resolution.

Consider the following example where the carrier Agent3 should choose among two distinct obligations to perform delivery towards two buyer agents (Agent1 and Agent2). Furthermore, consider that no simultaneous delivery is feasible and if the first obligation is being violated a 1000 euros penalty arises while if the second one is being violated then a 100 euros penalty arises but also new obligations that contradict with others come up. During this scenario Agent3 may resolve the dilemma either by adopting as the utility factor the pre-agreed compensation in case of violation and gives priority to the second default or by considering the new conflicting obligations and their effects on its plan and gives priority to the first default.

9.6 Conflicts and Assumptions

Another point that is worth exploring is the interaction between the ordering of defaults as this result from stratification and the construction of lattice structures, as per the proposal of this thesis, and other approaches such as the PDfL. Recall that, the technique described before, towards autonomous hypothetical and non-monotonic reasoning, resembles, in a way, stratification of a DfT. The possible default formulations of each initial norm are assigned to the various lattice levels, depending on the number of assumptions that each default formulation employs. We came to the conclusion that, an agent when using stratification on the set of available lattice structures before performing its reasoning within the lattices, it does not miss any causal knowledge and avoids employing unhelpful assumptions.

Naturally, a question that arises is how these distinct ordering methods (one due to stratification/lattices and one due to priorities) interact with each other. For stratification/lattices, we seek for relations among defaults. This relation is based, first, on propositions that are common in two or more rules, and second, on the number of assumptions employed in each default rule. Thus different strata define a causal hierarchy between lattices. On the contrary, in PDfL, hierarchy/priority relations are not defined for all defaults but only among those that have conflicting consequents. For example, consider that we have two buyers BA and BA' that order goods from the same seller SA, but at different time points (T<T'<T1). Consequently, from the SA's perspective, two obligations for delivery hold. Suppose that only one action may be performed at any given time. In this case, the SA agent has two conflicting obligations to satisfy. If time is used as the criterion in the general pattern rule for priority ascription then the default that infer the SA's obligation to deliver to BA takes priority over the other default that infer its obligation to deliver to BA'.

A reasonable choice seems to be to give precedence to stratification/lattices. We came to this conclusion, because in this way we infer more knowledge, even on a hypothetical basis, before moving on with the reasoning procedure. In this way an agent forms a better view (even hypothetical) of the world and its potential pasts or futures. Then we may assign priorities among conflicting norms, if possible, within each stratum based on some conflict resolution strategy.

9.7 Related Work and Summary

It is clear from the above discussion that the analysis, representation and management of normative conflicts have been the focus of much research in recent years, from a variety of perspectives. Here, we presented a summary of these research approaches and perspectives, briefly.

Moffett *et al.* [181], Lupu *et al.* [164] and Dunlop *et al.* [73, 74] address conflicts from the Distributed Systems Management viewpoint by specifying policies as a way to determine and influence management behaviour. We have shown how the basic types of conflict presented in these approaches may be seen as instances of our primitive patterns in DfL. Both [181] and [164] focus on the detection and resolution of syntactic conflicts at compile-time, by proposing static priority assignment. On the other hand, work in [73, 74], which addresses temporal reasoning about conflicts, concentrates on run-time conflict detection. Our approach is intended for conflict detection and resolution at run-time. Cholvy *et al.* in [47, 48] accept only inter-role conflicts and propose a solution that is based on the concept of role and regulation respectively. Contrary to this approach we accept intra-role conflicts and have shown how their conflict patterns map onto our primitive ones. Note that none of the above approaches supports defeasible reasoning.

Broersen *et al.* in [31] deal with different kinds of conflicts: they are interested in conflicts arising between an agent’s beliefs, obligations, intentions and desires. Although they, too, use DfL, they only use normal defaults, thus requiring agents to have complete knowledge. They do not address conflicts in a temporal setting. Other approaches, such as [111, 105, 191], that also support nonmonotonic reasoning with e-contracts do not presented in detail a discussion on the conflict patterns they consider and priorities are statically defined.

Abraham and Bacon in [3] examine normative conflicts but their focus corresponds to only a part of our set of primitive patterns. Although the absence of implicit knowledge is mentioned as a conflicting pattern, no resolution is proposed because no assumption-based reasoning is supported. Kowalski [146] is concerned with goal-driven conflict detection and resolution and attempts to unify logic with decision theory. Finally, Kollingbaum *et al.* [145, 245] focus on practical reasoning agents and norm-regulated Virtual Organizations. Specifically, they are only interested in situations of our conflict pattern B. They use either instantiation graphs for actions or unification to detect and resolve conflicts.

To sum up, in this chapter we presented a set of normative conflict patterns that may be encountered in e-contracts, and discussed how other analyses of normative conflicts found in the literature of distributed systems, legal reasoning and multiagent interaction may be seen as instances of these patterns. We also identified some conflicts that have not been identified yet in other proposals. Finally, we discussed how the representation of contractual norms as default rules facilitates both conflict detection and dynamic conflict resolution in a total or partial factual/hypothetical setting.

Research discussed in this chapter has been published or is under review in [90, 94, 93, 99, 87]

10 Other Issues for Common-sense Reasoning

10.1 Introduction

During a business transaction that is regulated by some agreement, other issues of interest for an agent towards contract performance monitoring are to establish:

- *Factual information*, given a history of events that have occurred up to the point of its query. For instance, an agent for an e-commerce application may need to establish what facts are true of orders, payments, deliveries etc., that have occurred (who caused such events, when, whether they were carried out successfully and so on).
- *Prescriptive information*, given a history of events that have occurred up to the point of its query. That is, an agent needs to know what obligations, permissions, prohibitions and legal powers are active for itself and each other agent in its environment.

To answer such queries some kind of temporal reasoning and reasoning about actions and their effects is required. Many researchers (for example, [166, 17, 79, 211] among others) have adopted EC [149] for contract representation. However, the historical information available to an agent at the time point it poses its query may be incomplete, for various reasons: Information may be lost, or distorted by noise, and in a truly open system, where agents join or leave the system at different times, information delivery from agent to agent may simply be delayed. Therefore, in order to reason in the presence of incomplete historical knowledge, agents must be able to fill in information gaps, by employing assumptions about the past and the present time. In this chapter, first, we discuss issues such as reasoning with time, action and deontic modalities, and, second, we link the ideas presented in the previous chapters with a contract representation in EC towards reasoning with incomplete knowledge.

The rest of this chapter is organized as follows: section 10.2 presents the full representation of contract norms and the way this representation enables reasoning with time, actions and deontic modalities; section 10.3 illustrates the proposals of this thesis through an example where contract norms are represented in the full language; and finally, section 10.4 provides a discussion on related work and a summary.

10.2 Reasoning with Time, Action and Deontic Modalities

To establish the state of a business exchange, given a history of parties' actions, we may represent the agreement that regulates this exchange in some temporal logic. In fact, such representations have been constructed for various types of agreements by many other researchers in Event Calculus (e.g. [166, 17, 79, 211] among others). The basic elements of the language are *time points*, *fluents* and *actions* or *events*. Fluents are factual and normative propositions whose truth-value alters over time, as a result of the occurrence of an action or an event.

In this thesis, we adapt the simple EC formalism presented in [180]. In its original form, the formalism does not distinguish between events that are brought about through agents' actions, and *force majeure* events that are brought about independently of the agents. We preserve the distinction and use the term 'action' to refer to the former, and 'event' to refer to the latter. We use terms, such as `Order(agent1, agent2)`, for fluents that become true as a result of specific actions (here ordering `AOrder(agent1, agent2)`). We use terms of the form `NN(agent1, agent2, action, time)` for fluents that describe normative propositions and their intended reading is "`agent1` is in legal relation `NN` towards `agent2` to perform `action` by `time`". The legal relation `NN` may be obligation, prohibition or permission; although these notions are typically formalized in some system of Deontic Logic, we merely use them as descriptive names for fluents, and do not adopt any specific Deontic Logic axiomatization.

As [165, 17] note, the effects of an action apply only when the action is considered valid, and this, in turn depends on whether its agent has the legal and practical ability to perform it. An agent's legal and practical ability with respect to certain actions may be time-dependent, so we use the fluents IPower(agent, action) and PAbility(agent, action) respectively, and the fluent Valid(agent, action) to denote that an action performed by an agent is valid. We employ the six basic predicates of [180], shown in Table 3.1 in chapter 3; of those, Initiates and Terminates are used along with Happens in the specific description of a particular contract, to represent causal relations between fluents and actions/events. The other three are defined in a domain-independent manner. We modify the original definition of the HoldsAt predicate to take into account action validity, and have, consequently, extended the Happens predicate to include the agent of an action as an argument (for events, though, we use the original form of Happens).

For illustration purposes, some domain-independent definitions are shown below:

$$\mathrm{Clipped(time1,\ fluent,\ time2)} \leftarrow \mathrm{(Happens(agent,\ action,\ time)} \wedge \mathrm{Terminates(action,\ fluent,\ time)} \wedge \mathrm{time1} \leq \mathrm{time} < \mathrm{time2} \wedge \mathrm{HoldsAt(Valid(agent,\ action),\ time))}$$

$$\mathrm{Declipped(time1,\ fluent,\ time2)} \leftarrow \mathrm{(Happens(agent,\ action,\ time)} \wedge \mathrm{Initiates(action,\ fluent,\ time)} \wedge \mathrm{time1} \leq \mathrm{time} < \mathrm{time2} \wedge \mathrm{HoldsAt(Valid(agent,\ action),\ time))}$$

$$\mathrm{HoldsAt(fluent,\ time2)} \leftarrow \mathrm{(Happens(agent,\ action,\ time1)} \wedge \mathrm{Initiates(action,\ fluent,\ time1)} \wedge \mathrm{time1} < \mathrm{time2} \wedge \neg\mathrm{Clipped(time1,\ fluent,\ time2)} \wedge \mathrm{HoldsAt(Valid(agent,\ action),\ time1))}$$

$$\neg\mathrm{HoldsAt(fluent,\ time2)} \leftarrow \mathrm{(Happens(agent,\ action,\ time1)} \wedge \mathrm{Terminates(action,\ fluent,\ time1)} \wedge \mathrm{time1} < \mathrm{time2} \wedge \neg\mathrm{Declipped(time1,\ fluent,\ time2)} \wedge \mathrm{HoldsAt(Valid(agent,\ action),\ time1))}$$

$$\mathrm{HoldsAt(fluent,\ time2)} \leftarrow \mathrm{(HoldsAt(fluent,\ time1)} \wedge \mathrm{time1} < \mathrm{time2} \wedge \neg\mathrm{Clipped(time1,\ fluent,\ time2))}$$

$$\neg\mathrm{HoldsAt(fluent,\ time2)} \leftarrow \mathrm{(HoldsAt(fluent,\ time1)} \wedge \mathrm{time1} < \mathrm{time2} \wedge \neg\mathrm{Declipped(time1,\ fluent,\ time2))}$$

Note that the first definition for HoldsAt above reflects the establishment of a fluent as a result of an action, while the second one reflects the common sense law of inertia.

As stated in chapter 6, the EC representation of an e-contract may be characterized as a triple (H, R, A). Specifically, H corresponds to historical information and is a (possibly empty/incomplete) set of definitions for predicates $H_L = \{\mathrm{Happens},\ \mathrm{Holds}\}$, R corresponds to causal information and is a (possibly empty/incomplete) set of definitions for $R_L = \{\mathrm{Initiates},\ \mathrm{Terminates}\}$, and A is the (non-empty) set of definitions for the domain–independent predicates $A_L = \{\text{HoldsAt}, \neg\text{HoldsAt}, \text{Clipped}, \text{Declipped}\}$, that is, $A = \{Y \leftarrow X_1 \wedge \ldots \wedge X_k \mid Y \in A_L \text{ and } X_i \in A_L \cup H_L \cup R_L \cup T_L\}$, where T_L contains the first-order-logic predicates used to express temporal relations, i.e., $T_L = \{<, =, >, \geq, \leq\}$. So, the complete language is $H_L \cup R_L \cup A_L \cup T_L$.

Another point worth mentioning is the so called Contrary-To-Duty structures [200]. CTDs arise when a primary obligation is defined for a party, along with a rule that determines a secondary obligation for it, should the primary one be violated. For instance, consider that the seller agent is obliged to deliver within 10 days from the date the buyer agent order took place. If it does not do so, then it is obliged to deliver within the next 3 days and to claim a reduced price. We should note, that during this work it is not our purpose to analyse all possible cases of CTD structures as presented in [200]. We do not address issues that concern the persistence of norms or indeed periodicity. We assume that when primary obligations are violated, some reparation action may be specified in the same manner the primary obligations were defined.

10.3 Example

Consider a 3-party business transaction that takes place in an electronic marketplace populated by software agents, as already discussed in chapters 4, 6 and 7. Let the set {BA, SA, CA, ...} denote agents in the e-market (where BA, SA, CA denote *buyer*, *seller* and *carrier* respectively). A buyer agent (BA) communicates with a seller agent (SA) and establishes an agreement with it for purchasing a certain product. Consequently, SA communicates with a carrier agent (CA) and establishes another agreement with it for the timely and safe delivery of goods to BA.

The first agreement (between BA and SA) is to be conducted on the following terms: SA should see to it that the goods be delivered to BA within 10 days from the date BA's order happens. BA, in turn, should see to it that payment be made within 21 days from the date it receives the goods. The agreement may specify sanctions and possible reparations in case the two agents do not comply with their obligations, but we do not need to refer to them explicitly here. In the same spirit, the second agreement (between SA and CA) specifies obligations, deadlines and possible sanctions/reparations in case of violations.

Here is an extract of the H, R and A sets of the EC representation for the agreement between BA and SA. Recall that this information may be incomplete, i.e., an agent may possess only partial historical knowledge (here, BA knows that it ordered from SA at time point T) and partial causal knowledge (here, BA knows that placing an order imposes an obligation on the recipient of the order to deliver; it knows that this obligation is terminated/discharged successfully when delivery actually takes place; and it also knows that the occurrence of delivery imposes an obligation on itself for payment, which is terminated when payment is actually made):

H^{BA} = { Happens(BA, AOrder(BA, SA), T) }

R^{BA} = {

R1 ≡ Initiates(AOrder(agent1, agent2), Obligation(agent2, agent1, ADelivery(agent2, agent1), time1+10), time1) ←
Happens(agent1, AOrder(agent1, agent2), time1)

R2 ≡ Initiates(ADelivery(agent1, agent2), Obligation(agent2, agent1, APayment(agent2, agent1), time1+21), time1) ←
(Happens(agent1, ADelivery(agent1, agent2), time1)
∧ HoldsAt(Obligation(agent1, agent1, ADelivery(agent1, agent2), time2), time1)
∧ time1≤ time2)

R3 ≡ Terminates(ADelivery(agent1, agent2), Obligation(agent1, agent2, ADelivery(agent1, agent2), time2), time1) ←
(Happens(agent1, ADelivery(agent1, agent2), time1)
∧ HoldsAt(Obligation(agent1, agent2, ADelivery(agent1, agent2), time2), time1)
∧ time1≤ time2)

R4 ≡ Terminates(APayment(agent1, agent2), Obligation(agent1, agent2, APayment(agent1, agent2), time2), time1) ←
(Happens(agent1, APayment(agent1, agent2), time1)
∧ HoldsAt(Obligation(agent1, agent2, APayment(agent1, agent2), time2), time1)
∧ time1≤ time2)

}

A^{BA} = {

A1 ≡ HoldsAt(Obligation(agent2, agent1, ADelivery(agent2, agent1), time1+10), time2) ←
Happens(agent1, AOrder(agent1, agent2), time1)
∧ Initiates(AOrder(agent1, agent2),Obligation(agent1, agent1,ADelivery(agent2, agent1),time1+10),time1)
∧ ¬Clipped(time1, Obligation(agent2, agent1, ADelivery(agent2, agent1), time1+10), time2)
∧ HoldsAt(Valid(agent1, AOrder(agent1, agent2)), time1)

time1<time2

A2 ≡ HoldsAt(Obligation(agent2, agent1, APayment(agent2, agent1), time1+21), time2) ←
Happens(agent1, ADelivery(agent1, agent2), time1)
∧ Initiates(ADelivery(agent1, agent2),Obligation(agent2, agent1,APayment(agent2, agent1), time1+21),time1)
∧ ¬Clipped(time1, Obligation(agent2, agent1, APayment(agent2, agent1), time1+21), time2)
∧ HoldsAt(Valid(agent1, ADelivery(agent1, agent2)), time1)
∧ time1<time2

}

With reference to this representation, and given BA's current knowledge, only rule R1 may actually be used for inference, since its conditions are satisfied, and so BA may only infer that:

Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T)

But what if BA needs to perform *best-guess* or *non-risk* reasoning? In this case BA needs to identify rule conditions that it may use as assumptions, and we proposed that this is possible, if the initial set of contract rules is reformulated as default rules. Since many different formulations are possible for each contract rule, the agent need not commit (statically) to some specific one, and instead it may construct the lattice of possible default formulations, as we argued earlier. In this way, the agent will be able to use any of the possible formulations, depending on its currently available knowledge, which changes over time; essentially the agent will be identifying candidate assumptions dynamically.

As a result the agent constructs the following DfT (W^{BA}, D^{BA}):

W^{BA} = { Happens(BA, AOrder(BA, SA), T) },

that is, W^{BA} contains the historical information available to the agent, and D^{BA} is the set containing the corresponding lattices of default formulations for each rule contained in the R^{BA} and A^{BA} sets.

10.3.1 Hypothetical Reasoning (1H)

Now, BA is able to perform both *no-risk* and *best-guess* reasoning by employing some of these defaults, i.e. by employing assumptions in its knowledge base. For example, in the absence of information to the contrary, it may assume that its order is a valid action and that SA's obligation to deliver is not unexpectedly terminated, in order to infer that SA bears an obligation to deliver the ordered goods. BA may come to this conclusion by employing in its inference the defaults DR1 and DA1, respectively, and by computing the In and Out sets as shown below[21]:

DR1 ≡
Happens(BA, AOrder(BA,SA), T)
: true
/ Initiates(AOrder(BA, SA), Obligation(SA,BA,ADelivery(SA,BA), T+10), T)

DA1 ≡
Happens(BA, AOrder(BA, SA), T), Initiates(AOrder(BA,SA),Obligation(SA,BA,ADelivery(SA,BA),T+10),T)
: ¬Clipped(T, Obligation(SA, BA, ADelivery(SA, BA), T+10),T1), HoldsAt(Valid(BA, AOrder(BA, SA)), T), T < T1
/ HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T1)

In(2) ^{BA} = W^{BA} ∪ { Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T),
HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10),T1) }

Out(2) ^{BA} = W^{BA} ∪ { Clipped(T, Obligation(SA, BA, ADelivery(SA, BA), T+10), T1),
¬ HoldsAt(Valid(BA, AOrder(BA, SA)), T), ¬ (T< T1) }

In the same spirit, and on the assumptions that: SA's delivery will happen at some time point; such delivery will be valid; the effect of such delivery will be an obligation for BA to pay; and, finally, that such obligation will not be terminated by some other action, BA may infer what its potential payment period will be, relative to the time point of its assumptions. BA may come to this conclusion by employing in its inference the defaults DR2 and DA2, respectively, and by computing the In and Out sets as shown below[22]:

DR2 ≡
true
: Happens(SA, ADelivery(SA, BA), T'), HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T'), T' ≤ T'
/ Initiates(ADelivery(SA,BA),Obligation(BA,SA,APayment(BA,SA), T'+21), T')

DA2 ≡

---

[21] Note that in our example time is discrete. So agents may generate past or future time points in order to make their assumptions. The only requirement for agents when assuming the existence of time points is to position each new time point in the overall time sequence, by introducing their temporal relation to other, known or assumed, time points.

[22] Note that the use of time point T' is possible under the assumptions that $T1 < T' < T2$ and $T' \leq T+10$.

true

: Happens(SA, ADelivery(SA, BA), T’), Initiates(ADelivery(SA,BA),Obligation(BA,SA,APayment(BA,SA), T’+21), T’),
¬Clipped(T’, Obligation(BA, SA, APayment(BA, SA), T’+21), T2), HoldsAt(Valid(SA, ADelivery(SA, BA)), T’), T’ < T2
/ HoldsAt(Obligation(BA, SA, APayment(BA,SA), T’+21), T2)

In(2)^{BA} = W^{BA} ∪ { Initiates(ADelivery(SA,BA), Obligation(BA, SA, APayment(BA, SA), T’+21), T’),
HoldsAt(Obligation(BA, SA, APayment(BA, SA), T’+21, T2) }

Out(2)^{BA} = W^{BA} ∪ { ¬ Happens(SA, ADelivery(SA, BA), T’),
¬ HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T’), ¬ (T’≤ T’),
¬ Happens(SA, ADelivery(SA, BA), T’),
¬ Initiates(ADelivery(SA,BA),Obligation(BA,SA,APayment(BA,SA), T’+21), T’),
Clipped(T’, Obligation(BA, SA, APayment(BA, SA), T’+21), T2),
¬ HoldsAt(Valid(SA, ADelivery(SA, BA)), T’), ¬ (T’< T2) }

We may wish to restrict the assumption space. Suppose we wanted our agent BA to avoid assuming the validity of actions, and use information about action validity only when it explicitly knows about it. In this case, the PC=Out(0) set (the set of pre-constraints) must be initialized to contain the forbidden assumption. For example, in this case BA constructs a PcDfT where the PC set contains the following formula:

PC^{BA} = Out(0)^{BA} = { HoldsAt(Valid(agent, action), time) }

Now, neither DA1 nor DA2 defaults may be employed in the inference process.

10.3.2 Commitment to Assumptions (2H)

As noted earlier, following Reiter’s original computation of extensions of a DfT we may compute possible world models that are counter-intuitive: for instance, in our example above, BA would infer, after applying all of DR1, DR2, DA1 and DA2, the extension:

In(4) = W^{BA} ∪ { Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T),
HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T1),
Initiates(ADelivery(SA,BA), Obligation(BA, SA, APayment(BA, SA), T’+21), T’),
HoldsAt(Obligation(BA, SA, APayment(BA, SA), T’+21), T2) }

This extension seems to suggest that BA infers a possible version of the world, in which it bears an obligation to pay SA, although no delivery from SA is explicitly recorded in this world, and similarly that SA bears an obligation to deliver, although this world does not explicitly record that BA’s order is valid. As we explained earlier, if we employ Constrained Default Logic the assumptions employed by an agent at some time point constrain its future inferences, as we require joint consistency of assumptions. Moreover, if we use Stratified Default Logic, we ensure that assumptions are employed in a rational sequence, knowledge about causal relations between rules is preserved, and the agent resorts to assumptions only when it really has to do so.

10.3.3 Non-monotonic Reasoning (3H)

Consider, again the same initial knowledge and the same set of rules in H, R and A sets, for the agreement between BA and SA, but, now, with the restriction that if some agent is a known debtor, then all obligations that hold towards it are terminated, from the time point at which it becomes known that the agent is a debtor, onwards.

$$PC^{BA} = Out(0)^{BA} = \{\ \text{Clipped(time1, Obligation(agent1, agent2, action(agent1, agent2), time2), time2)} \leftarrow \text{HoldsAt(IsADebtor(agent2), time1)} \wedge \text{time1} < \text{time2}\ \}$$

With W^{BA} = { Happens(BA, AOrder(BA, SA), T) } BA may, initially, perform both *no-risk* and *best-guess* reasoning by employing assumptions as shown above. Now imagine that a later time point T’, BA is informed that SA is a debtor, i.e. HoldsAt(IsADebtor(SA), T’) is added in its knowledge base. This new information affects its previously drawn conclusion. In this case, BA needs to traverse the lattices downwards in order to retract its earlier assumptions and conclusions, and, if necessary, to choose alternative default formulations, compatible with its current, updated, knowledge.

10.3.4 Conflict Management

Consider, again the same set of rules in R and A sets, for an agreement, but, now, between a seller agent SA and two buyer agents BA’, i.e. a regular client, and BA’’, i.e. a new client. For SA, the initial information H may define conflicting obligations, i.e., SA may possess knowledge that, BA’ ordered from SA at time point T’ and BA’’ ordered from SA at time point T’’ > T’:

H^{SA} = { Happens(BA’, AOrder(BA’, SA), T’), Happens(BA’’, AOrder(BA’’, SA), T’’) }

Thus, SA may find itself in a conflicting state at time point T1>T’’>T’ (intra-extension conflict) where two conflicting obligations are active by employing in its inference the defaults DR1’, DR1’’, DA1’ and DA1’’, respectively, and by computing the In and Out sets as shown below:

DR1’ ≡
Happens(BA’, AOrder(BA’,SA), T’)

: true

/ Initiates(AOrder(BA’, SA), Obligation(SA,BA’,ADelivery(SA,BA’), T’+10), T’)

DA1’ ≡

Happens(BA’, AOrder(BA’, SA), T’), Initiates(AOrder(BA’,SA),Obligation(SA,BA’,ADelivery(SA,BA’),T’+10),T’)

: ¬Clipped(T’, Obligation(SA, BA’, ADelivery(SA, BA’), T’+10),T1), HoldsAt(Valid(BA’, AOrder(BA’, SA)), T’), T’ < T1

/ HoldsAt(Obligation(SA, BA’, ADelivery(SA, BA’), T’+10), T1)

DR1’’ ≡
Happens(BA’’, AOrder(BA’’,SA), T’’)

: true

/ Initiates(AOrder(BA’’, SA), Obligation(SA,BA’’,ADelivery(SA,BA’’), T’’+10), T’’)

DA1’’ ≡

Happens(BA’’, AOrder(BA’’, SA), T’’), Initiates(AOrder(BA’’,SA),Obligation(SA,BA’’,ADelivery(SA,BA’’),T’’+10),T’’)

: ¬Clipped(T’’, Obligation(SA, BA’’, ADelivery(SA, BA’’), T’’+10),T1), HoldsAt(Valid(BA’’, AOrder(BA’’, SA)), T’’), T’’ < T1

/ HoldsAt(Obligation(SA, BA’’, ADelivery(SA, BA’’), T’’+10), T1)

In(4)^{SA} = W^{SA} ∪ { Initiates(AOrder(BA’, SA), Obligation(SA, BA’, ADelivery(SA, BA’), T’+10), T’),
Initiates(AOrder(BA’’, SA), Obligation(SA, BA’’, ADelivery(SA, BA’’), T’’+10), T’’),
HoldsAt(Obligation(SA, BA’, ADelivery(SA, BA’), T’+10),T1) ,
HoldsAt(Obligation(SA, BA’’, ADelivery(SA, BA’’), T’’+10),T1) }

Out(4)^{SA} = W^{SA} ∪ { Clipped(T’, Obligation(SA, BA’, ADelivery(SA, BA’), T’+10), T1),
Clipped(T’’, Obligation(SA, BA’’, ADelivery(SA, BA’’), T’’+10), T1),
¬ HoldsAt(Valid(BA’, AOrder(BA’, SA)), T’), ¬ (T’< T1) ,
¬ HoldsAt(Valid(BA’’, AOrder(BA’’, SA)), T’’), ¬ (T’’< T1) }

As discussed in section 9.5, in this scenario, there is information that can be used to determine different conflict resolution criteria. According to the strategy of temporality, based on external time, we may give priority to the obligation stated in DA1’ as it was initiated first. This is possible via the use of the PDfL and the general pattern for ascribing priorities dynamically:

$$\frac{\begin{array}{c}\text{Rule(d1,v1)}\wedge\text{Rule(d2,v2)}\wedge\text{criterion}\\ :\\ \text{assumptions}\end{array}}{\text{d1<d2}}$$

Recall that, d1, d2 are variables that denote names of ground defaults; Rule(d1 ,v1) denotes a ground default d1 and its set of entities of interest v1. Specifically, for the above case the following ground default could give precedence to the obligation stated in DA1’:

$$\frac{\begin{array}{c}\text{Rule(DA1',T'), Rule(DA1'',T''), T'} < \text{T''}\\ :\\ \text{true}\end{array}}{\text{DA1'} < \text{DA1''}}$$

Alternative, using the strategy of hierarchy, we may give precedence to DA1’, because BA’, as a regular client, takes precedence over BA’’, i.e.:

$$\frac{\begin{array}{c}\text{Rule(DA1',BA'), Rule(DA1'',BA''), RegularClient(BA')}\\ :\\ \text{true}\end{array}}{\text{DA1'} < \text{DA1''}}$$

Or, we may give precedence to DA1’’, because BA’’, as a new client, takes precedence over BA’, because it is possible to become a regular client, i.e.:

$$\frac{\begin{array}{c}\text{Rule(DA1',BA'), Rule(DA1'',BA''), }\neg\text{RegularClient(BA'')}\\ :\\ \text{BecomeRegularClient(BA'')}\end{array}}{\text{DA1''} < \text{DA1'}}$$

It is clear that various combinations of these criteria may also be defined based on the agent’s current knowledge and the assumptions it makes.

10.4 Related Work and Summary

A representation in Event Calculus, allows us to establish what each party is obliged (or permitted, forbidden, empowered) to do at a given time point. It also allows us to determine whether each party complies with the agreement, and what, if any, reparatory mechanisms are stipulated, should violations arise. We may, also, spot potential conflicts, for example if such a query returns that a particular agent is both obliged and forbidden to perform a specific action at the same time.

This representation, though, does not allow us to reason with incomplete knowledge dynamically. Towards this scope we argued that e-contracts could be represented as DfT that are constructed automatically from initial Event Calculus-based contract representations. Of course, there are other temporal languages to reason with time, actions and their effects, (for example, [219, 55, 56, 218, 217]) or other research approaches to reason with time and actions or with incomplete knowledge that adopt an e-contract representation in EC, (for example, [166, 17, 79, 211, 143, 261, 111, 187, 191] among others). These approaches were reviewed and discussed in detail in chapters 4 and 8.

Research discussed in this chapter has been published or is under review in [91, 90, 92, 95, 98, 86].

Part IV
Conclusions and Future Work

11 Conclusions

This thesis presented work conducted within a project that is concerned with knowledge representation and common-sense reasoning in an open computational environment, populated by software agents, whose interactions are regulated by electronic agreements, i.e. e-contracts. Agents in such environments will need to be able to monitor their interactions with other agents against the agreements that they are involved in, in order to determine what actions to perform and when.

Specifically, this thesis focused on issues such as: (i) the agent will need to establish factual or prescriptive information, that is, given a history of events, what factual information is established and what norms are active for each party, (ii) if the history of events is incomplete, or if the agent possesses incomplete or inconsistent domain knowledge, or if the agent needs to plan its future activities, reasoning needs to employ assumptions; if more information become available later, rendering some of these assumptions false, any conclusions drawn will need to be retracted, and (iii) whether normative conflicts arise for the agent, that is, whether it finds itself in a situation where it bears norms that it cannot fulfill simultaneously.

Such reasoning is essential in:

- autonomous multi-agent systems and robotic systems, where systems need to manage the degree of their autonomicity and rationality. We believe that this is possible by managing appropriately the assumptions they employ, and correspondingly their actions.
- legal systems where obligations, permissions, prohibitions hold and violations and conflicts arise. In such situations we may need to establish or to assume that certain actions will occur or have occurred, or that certain causal relations

will be effected in the environment, or that agents bear a certain normative status (obligations, permissions, prohibitions, powers) towards other agents, in order to plan future activities or avoid undesirable situations.

- autonomic computing, where systems are self-regulated, self-monitoring and self-configured. Using the ideas presented in this thesis, systems may develop for themselves the laws and strategies according to which they regulate their behaviour and make their own inferences by relying on their own strategy.
- commercial applications, e.g. e-contracts, service level agreements, negotiation, and semantic web applications, such as service composition, where systems may compute possible worlds on the basis of different hypothetical scenaria in order to model check, verify and monitor services.

This chapter is organized as follows: section 11.1 provides a brief review of the work presented in this thesis; section 11.2 provides a listed view of the contributions of this thesis and its relation to other research approaches; and finally, section 11.3 presents directions for future research.

11.1 Summary

In order to establish the state of the business exchange, i.e. factual and prescriptive information, given the actions that parties perform or omit to perform, we employed a representation of the agreement in Event Calculus [149]. Specifically, we adapted the simple Event Calculus formalism presented in [169]. The agreement representation in Event Calculus, allows us to establish what each party is obliged (or permitted, forbidden, empowered) to do at a given time point. It also allows us to determine whether each party complies with the agreement, and what, if any, reparatory mechanisms are stipulated, should violations arise. This representation, though, does not allow us to reason with incomplete knowledge dynamically.

Therefore, on the basis of such a representation and in order to enable and support agents that perform dynamic and adaptive reasoning, we were inspired by Reiter’s Default Logic [208]. Generally, we discussed two reasons why it is useful for an agent to be able to reason on a dynamic basis. First, an agent may not know everything about the past and the present and thus the dynamic production of assumptions is needed in order to explain a given situation. Moreover, the agent may wish to plan its future activities on the assumption that certain events/actions will occur, and that certain causal relations will be effected, or that its partners' actions will be valid. In both cases we are interested in making specific assumptions about the truth of certain formulae in order to infer hypothetical possible worlds. Second, priorities over contract rules should be defined and reconsidered dynamically by specifying domain-dependent criteria.

Towards these directions, a representation of e-contracts, and generally, a representation of open computational environments, as Default Theories was proposed. Our approach were inspired by the syntax and semantics of Default Logic, without however resorting to proof, which is notably computationally hard. We chose Default Logic for three reasons:

- The syntax of Default Logic offers an intuitive way to represent separately what is known, what is assumed and what is concluded on the basis of this knowledge and assumptions; the schema of Default Logic rules comprises three distinct parts, namely prerequisites, justifications and consequents.
- The semantics of Default Logic and its variations offers a way to reason non-monotonically by preserving the relation of an assumption and any inferences drawn on its basis (in the sense of argumentation [198]) and to maintain consistency and rationality.
- Implementation is feasible without resorting to theorem proving, but, by resorting to set manipulation, i.e. by maintaining syntactically consistent sets of formulae, whose conditions part (prerequisites and justifications) is interpreted conjunctively and the conclusions part (consequent) is interpreted disjunctively, as in sequent calculus.

The final environment representation in Default Logic results as the outcome of the reconstruction of the e-contract's initial Event Calculus-based representation. Specifically, each formula of the initial representation is mapped into one of the corresponding possible default rules. We presented a first proposal for the dynamic theory construction, but this was computationally unacceptable, since it requires an agent to attempt to prove literals from its knowledge base, in order to decide whether to use them in the prerequisite or the justification part of each default that it constructs. In other words, the agent needs to attempt to prove literals (and fail in doing so) in order to determine which of these are candidate assumptions. In order to overcome this limitation we presented an alternative procedure by which an agent may determine assumptions dynamically and consequently construct the theory. This technique does not require the agent to prove literals from its current knowledge base, and therefore, it is suitable for implementation.

The main idea of the second proposal is the organization of all possible mappings of each initial contract rule into default rules in a hierarchical multi-level structure. Each level of the constructed structure contains one or more of the possible defaults, depending on the number of assumptions that these defaults employ. Of course, contracts (and normative systems in general) contain multiple rules, for each of which a structure may be constructed. Hence, a theory representation of an environment consists of a set of logic formulae that represent initially available knowledge, and a set of structures, each containing the possible formulations of a contract rule as a default rule.

First, we considered these structures, which denote what we called the single-norm knowledge/hypothesis space, to be represented as triangles and to be parsed upwards sequentially. Thus, the inference process starts from the ground level of the set of structures, which contain only justification-free defaults, by applying as many as possible given the agent's current knowledge. When there are no further defaults that can be applied in a level, this signals that assumptions are needed in order to proceed, and inference continues by examining defaults that lie in the next levels upwards.

Then, we re-introduce this incremental technique in a manner that enables agents to 'develop for themselves the laws and strategies according to which they regulate their behaviour (in the spirit of [219]) and to 'make their own inferences and reasoning and to rely on their own conclusions' (in the spirit of [40]). It turned out that the knowledge/hypothesis space is, in fact, a lattice. At any particular time point the agent may position itself on it, given the explicit knowledge that it currently possesses, i.e. without resorting to proof. Once the agent has positioned itself on this lattice, it finds out what assumptions are related to the node it occupies and may employ them in its reasoning. As the agent's knowledge changes over time, and consequently as its assumption needs change, the agent re-positions itself on the lattice by moving on it from node to node. The mathematical properties of the lattice structure that we used in order to represent knowledge/hypothesis spaces, first, suggests that an implementation is also feasible, relying only on set manipulation rather than proof, and, second, facilitates hypothetical nonmonotonic reasoning.

Moreover, we noted that, for an agent that employs assumptions in full freedom may be risky and unsafe or may lead to counter-intuitive and inconsistent worlds. Thus a technique that enables agents to control and adjust their hypotheses was presented. Specifically, we discussed the way Default Logic syntax, semantics and major variations are really helpful towards this scope.

Finally, this thesis addressed the issues of conflicts detection and dynamic conflict resolution. A set of primitive conflict patterns was presented and some patterns of normative conflict that have not been identified in other proposals were identified. We discussed how the proposed contract representation allows agents to detect conflicts by examining theory extensions. In general, a potential conflict arises when there are multiple extensions of a theory that represents a contract, and one of them contains a proposition that conflicts with a proposition contained in another, the so called inter-extension conflicts. Conflicts may also arise even when there is a single extension of the theory, if it contains conflicting propositions, the so called intra-extension conflicts. The detection of inter-extension conflicts is useful for an agent, which finds itself in a state that is not, yet, problematic, and has alternative courses of action to consider. The agent must decide upon a specific course of action – some way of preventing the potential conflicts from ever arising is required. The detection of intra-extension conflicts, on the other hand, essentially informs the agent that it is already or will be, in a problematic state. Again the agent needs a way to resolve the conflict and decide which norm to satisfy in a way that minimizes the damage done – since, unavoidably, some norm will be violated. Conflict resolution in our approach is performed using Brewka’s [30] proposal on prioritized theories that enables us to define and apply priorities on default rules. What makes prioritized theories really useful is that the ascription of priorities to default rules may, itself, be done dynamically. Using dynamic priorities, we generated preferred extensions, each of which indicates a transaction plan. Priorities amongst ground defaults may be defined dynamically either by making different assumptions or by specifying domain-dependent criteria. In this manner, we managed conflicts in a variety of ways, by specifying different criteria, such as hierarchies of entities of interest, time, specificity of norms, minimality or utility factors.

To sum up and regarding the requirements for knowledge representation and the specifications for a tool implementation that we identified in chapter 4, this thesis addresses requirements R2, R3, R4, R5, R7 and R8. Moreover, it establishes the need for requirement R10 in e-contracting frameworks and proposed a technique for hypothetical nonmonotonic reasoning towards the direction of requirements R9 and R10.

11.2 Contributions and Work in Context

The main contributions of this thesis are as follows:

- A critical review on various approaches on e-contract representation and performance monitoring. Through this analysis, first, we identified requirements that a representation of electronic agreements should meet, in order to facilitate the development of tools for contract performance monitoring, and second, helped us to review literature that is related to contract representation and contract performance monitoring and to identify which requirements each approach deals with.
- Identification of explicit research questions (1H - 3H) that arise in open norm-governed environment, where agents seek to establish missing information. The Open Default Assumption and the Dynamic Default Logic were presented in order to enable agents to common-sense reason within this setting.
- Two initial approaches to assumption-based reasoning within an open normative system were presented along with algorithms and a system architecture that support the implementation of a prototype.
- A revision of the initial approaches to assumption-based reasoning. The new proposal deals with hypothetical reasoning along with non-monotonic reasoning. The proposed technique enable agents to manage their reasoning (i.e. to maintain consistency, to restrict the assumptions they employ and to

entail rational conclusions) via the use of appropriate symbolic and schematic representations (i.e. lattices) of whatever knowledge agents possess and whatever knowledge agents ignore and may, possible, employ as assumptions. Through this attempt we have identified the role of assumption discovery in agents' autonomous reasoning.

- A critical analysis on various perspectives of normative conflicts management was made. This analysis: identified a set of primitive patterns for normative conflicts; showed how the conflicts identified by other researchers may be seen as instances of these primitives; identified some patterns of normative conflicts that have not been identified in other proposals. Moreover, we showed that the representation of contractual norms as default rules facilitates both conflict detection and resolution, first, by showing the way conflicts may be detected (i.e. inter-extension or intra-extension conflicts), and second, by showing how various strategies on conflicts resolution (i.e. hierarchy, temporality, specificity, quantitative or qualitative minimality of assumptions, utility factor) may apply in our approach.

Consequently, the consideration of this thesis into the research area of Artificial Intelligence is as follows:

- Regarding to the assumption-based reasoning approaches this thesis proposed a technique for the dynamic identification and usage of appropriate assumptions without resorting to a pre-specified pool of assumptions, or to a goal-oriented manner or even deductive proof. As discussed in chapter 8, some of the previous approaches to assumption-based reasoning rely on the use of a pre-specified pool of assumptions, from which the agent must choose appropriate ones, whenever it identifies an information gap and needs to fill it, in order to proceed with its reasoning. We believe that it is unrealistic to expect that candidate assumptions can be identified in advance. It may be the case that in some application domains this is possible. However, in such cases, candidate assumption identification is not really dynamic, rather selection of an appropriate assumption from the pre-specified pool, may be carried out dynamically during the inference process. This selection though, requires deductive proof, which is notably computationally expensive. Other approaches that purport to support dynamic identification of assumptions, rely on finding appropriate assumptions in a goal-driven manner, that is, a particular conclusion that the agent wants to derive is given, and then the agent identifies the assumptions that are required, in order for this conclusion to be derivable. In some cases, such goal-driven identification of candidate assumptions requires proof. But more importantly, the problem that we perceive with purely goal-driven assumption identification is the following: although software agents, in general, are inherently goal-driven in planning their activity, their rationality (and consequently their performance measures) depends on the extent to which they are perceptive of their environment, so that they may exploit changes in it. A purely goal-driven identification of candidate assumptions does not leave much room for the agent to adapt to circumstances.

- Regarding to the non-monotonic reasoning approaches for e-contracting this thesis focuses on the unavoidable changes of the world and the vital changes of the initial considerations, i.e. the dynamics of theories. As discussed in chapter 8, some of the previous approaches to non-monotonic reasoning require that an agent needs: to define abnormal events, effects of actions and the like, explicitly, and, also, to distinguish each abnormal individual from other individuals, explicitly; to decide a priori what conclusion it wants to derive, in order to be able to identify which assumptions are essential to make, in order to be able to actually derive it; to determine, a priori, during the construction of the rule base, what is and what is not defeasible. In this thesis, we followed a more open consideration and require theories where the agent is able to decide in an ad hoc manner whether a condition of a norm is defeasible or not.
- Regarding to the dynamics of the Default Theory, as discussed in chapters 5 and 8, some approaches focus on changing facts and constraints while defaults remain unchanged, while other approaches introduce actions to model the attempt to jump to conclusions on the basis of a set of beliefs via the use of supernormal defaults. Our approach is close to the latter approach, but we see that rule reformulation towards assumption employment, compared to belief

generation via interspersed rules, affords us the ability to directly relate conclusions with employed assumptions in in the sense of argumentation [198] which consequently facilitates nonmonotonicity and autonomy.

- Regarding to the issue of autonomous agency, as discussed in chapters 7 and 8, earlier work on autonomy focused on its relation to an agent’s goals, i.e. on the extent to which an agent could choose its goals and pursue them without external intervention and more recently, autonomy is examined in relation to an agent’s reasoning process in general. In this thesis, we focused on the most recent perspective. Our agent is expected to make inferences about which beliefs to adopt about its environment, other agents and norms in force, which goals to commit to, and which actions to perform, in the presence of incomplete or inconsistent information, and it is expected to be independent from external intervention in this reasoning process. We believe that the degree to which an agent's reasoning is autonomous is affected by the degree to which it is able to choose its assumptions autonomously. We claimed that an agent that answers the hypothetical reasoning problem, addresses also the autonomy problem, i.e. an agent that possesses self-knowledge and is self-regulated and self-managed.
- Regarding to the issue of conflict management, as discussed in chapter 9, this thesis provided a vital critical survey on various analyses of normative conflicts that were found in the literature of distributed systems, legal reasoning and multi-agent interaction. We showed that all conflicts that were identified, from a variety of perspectives, may be seen as instances of some patterns. Finally, regarding to conflict resolution, contrary to other approaches to e-contracting, we used priorities among rules that where assigned dynamically either by making different assumptions or by specifying domain-dependent criteria.

11.3 Future Work

This section provides directions for future research for both theoretical and practical issues and other potential applications of the proposals presented in this thesis.

The first in line of future work seems to be the implementation of a robust complete system that will combine all the ideas presented in this thesis. In chapter 6, we discussed a prototype and technologies towards the application of our initial proposal to assumption-based reasoning within the settings of a dynamic normative system. Latter, in chapter 7, we re-introduced our proposal in a more sophisticated manner and discussed vital changes on our first approach. This fact calls for the extension of the prototype in order to support knowledge management and hypothetical nonmonotonic reasoning as proposed in chapter 7. Additionally, towards knowledge management and the implementation of a complete framework, we could apply the techniques for conflict detection and conflict resolution as proposed in chapter 9.

In this thesis, we discussed the way an atheist agent reasons with incomplete knowledge, i.e. an information gap is treated as negative information, and moreover, we enhanced the need for agnostic agents, i.e. agents that treat information gaps for what they are (absence of definite information) as potentially positive information. Consequently, regarding the various ways an agent could entail conclusions in realistic scenaria and towards the theoretical extension of our proposals, it comes natural the need to identify criteria and metrics that will enable agents to decide whether it is essential for their reasoning with incomplete/inconsistent/conflicting knowledge to proceed on a hypothetical basis, i.e. to employ either negative or positive assumptions in order to fill in information gaps by accepting some risk, instead of standing still in their environment and performing nothing due to their lack of explicit knowledge. Such an extension can provide a robust approach to hypothetical non-monotonic reasoning via the use of quantitative reasoning approaches along with qualitative reasoning approaches.

Also, we want to explore a richer representation, where: dynamic assumption generation and deployment is applied to agents’ beliefs, desires, intentions, roles, policies and the like; norms are represented using more expressive versions of Event Calculus [183] or other temporal/action languages, such as C/C+ [103, 101, 151].

Since, the relation of Reiter's Default Logic to other approaches to non-monotonic reasoning is known, another direction for future work is towards the examination of possible ways that the technique for dynamic identification and usage of hypotheses can be applied to other approaches to non-monotonic reasoning such as Logic Programs (with stable model [83] or answer set semantics [84]) and Defeasible Logic [187]. Of course, we neither want to rely on goal-oriented processes nor to rely on approaches where defeasibility is predefined. For example, a potential application of the Open World Assumption to Defeasible Logic would be to consider all rules as defeasible rules. In this way, although, we facilitate dynamic reasoning without the need for pre-specified abnormalities, we see that we bound the reasoning capability of an agent. In our approach, we search for hypotheses among the components, i.e. conditions, of the rules. This gives an agent the ability to explicitly and accurately distinct knowledge from not-knowledge.

Another point worth examining is the interpretation of open defaults. There are four major approaches (cf. [208, 155, 195, 137, 138]) to the semantics of open Default Theories. In chapter 7, we discussed why we adopted the approach proposed by Kaminski, i.e. we accepted the Domain Closure Assumption [137, 138] as putting the world in quarantine temporarily. In this case we accept a partially open environment, even temporarily. A question that arises is, whether this assumption is a restriction for an agent's reasoning capability in a truly open system.

Finally, towards the examination of other possible applications, we see that the ideas presented in this thesis could be used for further research in application areas that concern with:

- the correctness of contracts [231, 188, 65] and, generally, with model checking of computational systems, where given a model of a system, it is required to test automatically whether this model meets a given specification under various pragmatic or hypothetical, possible conflicting, scenaria.
- services composition and negotiation [232, 179, 203, 236], where an agent finds itself in a truly open environment and, realistically, possesses

incomplete knowledge. In such a setting, the agent unavoidably needs to employ assumptions.

- autonomous robotic systems for both cooperative and antagonistic problems, and, generally autonomic computing applications [141, 235, 129], where systems are self-managed. In such settings system components and services need to: be verified during runtime, i.e. self-verification; collect and analyze data, and then react according to the results, i.e. self-monitor; be able to evolve dynamically in an open world, i.e. self-configuration, and, finally, detect, diagnose, and repair problems, i.e. self-protection and self-healing.

Part V
Appendices
Bibliography and Index

Appendix A Survey Summary on e-Contracting Frameworks

In this appendix we provide a summary of surveyed approaches that have emerged during the last decade, are related to contract performance monitoring and are based on logic. Each approach is summarized with respect to (i) its goal and main aspects of interest, (ii) the recorded requirements for efficient reasoning with electronic contracts and (iii) its integration to Semantic Web and tool presentation.

E-commerce

- Santos and Carmo: [219]
  Goal and main aspects of interest: Compliance to the agreement.
  Ontology representation: Yes
  Temporal information representation: Dynamic operator for action and time.
  Deontic Modalities: Deontic Logic operators for Obligation and Prohibition.
  Legal and Physical ability: No.
  The representation of normative violation: Yes.
  Contrary to Duty Structures: No.
  Normative conflict representation and resolution: No.
  Auxiliary calculations: No.
  Default, Defeasible and Nonmonotonic Reasoning: No.
  Autonomous and Adaptive Reasoning: No.
  Integrated to Semantic Web: No.
  Tool: No.

- Daskalopulu *et al.*: [55, 56]
  Goal and main aspects of interest: Temporal reasoning over deontic specifications.

Ontology representation: Yes.

Temporal information representation: Modal Action Logic.

Deontic Modalities: Deontic Logic operators for Obligation and Permission.

Legal and Physical ability: Operator to represent Legal Power.

The representation of normative violation: Yes.

Contrary to Duty Structures: Yes.

Normative conflict representation and resolution: No.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Sallé *et al.*: [217, 218, 216]

Goal and main aspects of interest: Support the whole life-cycle of the contract.

Ontology representation: Yes.

Temporal information representation: Meyer's Dynamic Logic.

Deontic Modalities: Deontic Logic operators for Obligation, Permission and Prohibition.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: Two types of sanctions are supported.

Normative conflict representation and resolution: No.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Farrell *et al.*: [79]

Goal and main aspects of interest: Contract state tracking for Service Level Agreements.

Ontology representation: Yes.

Temporal information representation: Event Calculus.

Deontic Modalities: Deontic Logic operators for Obligation and Permission.

Legal and Physical ability: Institutionalized Power.

The representation of normative violation: This requirement can be met although it is not explicitly discussed.

Contrary to Duty Structures: This requirement can be met although it is not explicitly discussed.

Normative conflict representation and resolution: No.

Auxiliary calculations: Yes.

Default, Defeasible and Nonmonotonic Reasoning: Although no specific discussion is made, this requirement can be met under various interpretations of Event Calculus.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: ecXML

Tool: Event Calculus State Tracking Architecture (ECSTA).

- Knottenbelt and Clark: [143]

Goal and main aspects of interest: Contract monitoring.

Ontology representation: Yes.

Temporal information representation: Event Calculus.

Deontic Modalities: Deontic Logic operator only for Obligation.

Legal and Physical ability: Institutionalized Power.

The representation of normative violation: Yes.

Contrary to Duty Structures: This requirement can be met although it is not explicitly discussed.

Normative conflict representation and resolution: As noted, this requirement can be met through the BOID architecture as presented in [31], but it is not explicitly discussed.

Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

Default, Defeasible and Nonmonotonic Reasoning: Although no specific discussion is made, this requirement can be met under various interpretations of Event Calculus.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: A Prolog contract representation is given.

- Grosof *et al.*: [112, 204, 113, 111, 22]

Goal and main aspects of interest: Representation and execution of business rules in the WWW framework. Defeasibility. Priorities over rules.

Ontology representation: Yes.

Temporal information representation: No.

Deontic Modalities: No.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: No.

Normative conflict representation and resolution: Mutual exclusion statements are defined for conflict detection. Static priorities over rules are defined for conflict resolution.

Auxiliary calculations: Yes.

Default, Defeasible and Nonmonotonic Reasoning: Courteous Logic Programs (CLP) and Situated Courteous Logic Programs (SCLP).

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: Common Rules, Business Rules Markup Language (BRML).

Tool: Yes.

- Governatori *et al.*: [104, 105, 107]

Goal and main aspects of interest: Defeasibility. Priorities over rules. Contrary to Duty Structures.

Ontology representation: Yes.

Temporal information representation: No temporal dimension is explicitly given but an extension to this direction is feasible as shown in [108, 106, 109].

Deontic Modalities: Deontic Logic Operators for Obligation and Permission.

Legal and Physical ability: No.

The representation of normative violation: Defeasible Deontic Logic of Violation (DDLV).

Contrary to Duty Structures: Defeasible Deontic Logic of Violation (DDLV).

Normative conflict representation and resolution: Incompatible literals are defined for conflict detection. Static priorities over rules are defined for conflict resolution.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: Nute’s Defeasible Logic.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: Yes.

Tool: DR-Contract.

- Paschke *et al.*: [191, 189, 190]

Goal and main aspects of interest: Execution and monitoring of Service Level Agreements.

Ontology representation: Yes.

Temporal information representation: Event Calculus and Event-Condition-Action rules.

Deontic Modalities: Deontic Logic operators for Obligation, Permission and Prohibition. An extension of Standard Deontic Logic with a role-based model and further logic formalisms is presented.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: Yes.

Normative conflict representation and resolution: Authorization Conflict (Permission vs Prohibition) and Obligation Conflicts (Obligation vs Prohibition) and other application specific conflicts seem to be considered. Nute’s Defeasible

Logic and Grosof's Generalized Courteous Logic Programs (GCLP) are used for conflict resolution via rule prioritization.

Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

Default, Defeasible and Nonmonotonic Reasoning: Nute's Defeasible Logic and Grosof's Generalized Courteous Logic Programs (GCLP).

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: Yes.

Tool: ContractLog.

- Xu: [257, 258, 259, 260]

  Goal and main aspects of interest: Specification of monitoring requirements for e-marketplaces. Proactive monitoring and violation prevention.

  Ontology representation: Yes.

  Temporal information representation: Propositional Temporal Logic.

  Deontic Modalities: No (Commitment-based approach where a Commitment is not related to an Obligation).

  Legal and Physical ability: No.

  The representation of normative violation: Proactive violation detection and prevention is supported via commitment graphs.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: No.

  Autonomous and Adaptive Reasoning: No.

  Integrated to Semantic Web: No.

  Tool: No.

- Yolum and Singh: [261], Chopra and Singh: [49], Desai *et al.*: [64], Udupi and Singh: [241]

  Goal and main aspects of interest: Specification and execution of protocols that regulate multi agent interactions.

  Ontology representation: Yes (Commitment-based approach where a Commitment functions like a directed Obligation).

  Temporal information representation: Event Calculus, C+.

  Deontic Modalities: Yes (Commitment-based approach).

  Legal and Physical ability: No.

  The representation of normative violation: Yes. Transformations are used to verify compliance.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No.

Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

Default, Defeasible and Nonmonotonic Reasoning: Circumscription, C+.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Rouached *et al.*: [211, 212]

  Goal and main aspects of interest: Regulation of Web Services to support cross-organizational collaborations.

  Ontology representation: Yes.

  Temporal information representation: Event Calculus.

  Deontic Modalities: Yes (Commitment-based approach where deontic clauses can be defined in terms of operations on commitments. Obligations, Permissions and Prohibitions are considered).

  Legal and Physical ability: No.

  The representation of normative violation: No.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No.

  Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

  Default, Defeasible and Nonmonotonic Reasoning: No.

  Autonomous and Adaptive Reasoning: No.

  Integrated to Semantic Web: No.

  Tool: No.

- Letia and Groza: [153], Vartic and Letia [244]

  Goal and main aspects of interest: Business agreement regulation by specifying sets of commitments.

  Ontology representation: Yes

  Temporal information representation: Temporalized Defeasible Logic.

  Deontic Modalities: Yes (Commitment-based approach where a commitment is considered to be identical to an obligation).

  Legal and Physical ability: No.

  The representation of normative violation: Yes (via attaching deadlines for fulfilment to each commitment).

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No specific conflict patterns are recorded. Conflict resolution may be addressed via priority assignment over rules as used in Defeasible Logic.

  Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

Default, Defeasible and Nonmonotonic Reasoning: Nute's Defeasible Logic.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Tan and Thoen: [238, 239]

  Goal and main aspects of interest: Linguistic perspective of e-contracts. Directed deontic modalities.

  Ontology representation: It deals with e-contracts from a linguistic perspective.

  Temporal information representation: Formal Language for Business Communication (FLBC) and Event Semantics.

  Deontic Modalities: Yes (Alternative definition for directed Obligation is presented and an alternative definition for directed Permission is proposed. Alternative interrelation from the one considered in Standard Deontic Logic is proposed).

  Legal and Physical ability: No.

  The representation of normative violation: No.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: No.

  Autonomous and Adaptive Reasoning: No.

  Integrated to Semantic Web: Yes.

  Tool: A Prolog contract rules representation is given.

- Alberti *et al.*: [6, 7, 8]

  Goal and main aspects of interest: Monitoring and verification of e-contracts at run-time. Abductive Logic Programming.

  Ontology representation: No.

  Temporal information representation: Events, actions and time are represented in SCIFF logic language.

  Deontic Modalities: Yes. A mapping of deontic operators to abductive expectations is proposed.

  Legal and Physical ability: No.

  The representation of normative violation: Yes.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: Yes. No specific normative conflict patterns are discussed.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: No. We see that it is possible to address this requirement via the relation of Abductive Logic Programs with (general) Logic Programs (under stable model semantics [83]) or with extended Logic Programs (under answer sets [84]).

Autonomous and Adaptive Reasoning: We see that Abduction Logic Programming can be used towards this scope.

Integrated to Semantic Web: Yes. The encoding of SCIFF contract rules in RuleML is proposed.

Tool: An inference engine called SCIFF Reasoning Engine (SRE) is given.

- Giannikis and Daskalopulu:

Goal and main aspects of interest: Nonmonotonic and Hypothetical reasoning with e-contracts. Conflict Detection and Resolution.

Ontology representation: Yes.

Temporal information representation: Event Calculus.

Deontic Modalities: Deontic Logic Operators for Obligation, Permission and Prohibition. No specific Deontic Logic axiomatization is adopted.

Legal and Physical ability: Operators for Institutionalized Power and Practical Ability.

The representation of normative violation: Yes.

Contrary to Duty Structures: This requirement can be met although it is not discussed in detail.

Normative conflict representation and resolution: Conflict detection is possible by examining Default Logic extensions. Conflict resolution is possible using Brewka’s Prioritized Default Theory.

Auxiliary calculations: This requirement can be met although it is not explicitly discussed.

Default, Defeasible and Nonmonotonic Reasoning: Reiter’s Default Logic and its variations.

Autonomous and Adaptive Reasoning: A technique that enables agents to reformulate the initial set of norms by identifying and employing appropriate candidate assumptions dynamically is proposed.

Integrated to Semantic Web: Yes (XML norm representation).

Tool: A prototype is presented.

E-business

- Davulcu *et al.*: [58, 59]

Goal and main aspects of interest: Specification, analysis and scheduling of workflows. Manage adversarial situations.

Ontology representation: Yes.

Temporal information representation: Concurrent Transaction Logic (CTR) and CTR-S.

Deontic Modalities: No.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: No.

Normative conflict representation and resolution: No.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: Although this work proposes a logic to specify contracts in Semantic Web Services, no direct formalizations or technologies are discussed towards the integration to Semantic Web.

Tool: No.

- Marjanovic and Milosevic: [167]

Goal and main aspects of interest: E-contract modelling. Temporal and Deontic Constraints. Role windows. Time maps.

Ontology representation: Yes.

Temporal information representation: Various time operators and temporal constraints are defined for the Reference Model of Open Distributed Processing (RM-ODP).

Deontic Modalities: Deontic Logic operators for Obligation, Permission and Prohibition.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: No.

Normative conflict representation and resolution: Deontic inconsistencies are detected through role windows.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Lubwig and Stolze: [162]

Goal and main aspects of interest: Model and manage contractual content. Promise.

Ontology representation: Yes.

Temporal information representation: Yes (no specific temporal logic is adopted).

Deontic Modalities: Yes (Obligations and Rights are considered, Simple Obligations and Rights Model (SORM)).

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: Although no specific discussion is made, obligation dynamics may be addressed via the proposed actions that modify (add, remove, change) obligation sets.

Normative conflict representation and resolution: No.

Auxiliary calculations: Yes.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: No.

- Cardoso and Oliveira: [33, 34, 35]

Goal and main aspects of interest: Dynamic Electronic Institutions. Institutional reality. Norm modelling for contracts establishment, monitoring and enforcement.

Ontology representation: Yes.

Temporal information representation: Yes (Timestamps, no specific temporal logic is adopted).

Deontic Modalities: Yes (Directed Obligations are considered).

Legal and Physical ability: Yes (Brute and Institutional facts).

The representation of normative violation: Yes.

Contrary to Duty Structures: Yes. CTD structures are modelled as “default rules” by defining default clauses for CTD situations.

Normative conflict representation and resolution: No.

Auxiliary calculations: No.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: Yes (XML norm representation).

Tool: No.

Virtual Communities

- Dellarocas: [63]

Goal and main aspects of interest: Social Contracts for multi agent society control.

Ontology representation: Yes.

Temporal information representation: No.

Deontic Modalities: No.

Legal and Physical ability: No.

The representation of normative violation: Yes.

Contrary to Duty Structures: Yes (via the definition of positive and negative sanctions).

Normative conflict representation and resolution: No.

Auxiliary calculations: Yes.

Default, Defeasible and Nonmonotonic Reasoning: No.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: Contractual Agent Societies (CAS).

- Dignum *et al.*: [70, 69, 71]

  Goal and main aspects of interest: Social Contracts for multi agent society regulation.

  Ontology representation: Yes.

  Temporal information representation: Temporal and Deontic Logic (BTLcont) and the branching-time temporal logic (CTL*).

  Deontic Modalities: Deontic Logic expressions for Obligations and Conditional Obligations with deadlines are considered.

  Legal and Physical ability: No.

  The representation of normative violation: Yes.

  Contrary to Duty Structures: Yes.

  Normative conflict representation and resolution: Role-based conflicts are represented and considered.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: No.

  Autonomous and Adaptive Reasoning: No.

  Integrated to Semantic Web: No.

  Tool: Organizations per Agents (OperA).

- Boella and van der Torre: [23], Broersen *et al.*: [31]

  Goal and main aspects of interest: Social Contracts for multi agent society regulation. Conflicts between Beliefs, Obligations, Intentions and Desires.

  Ontology representation: Yes.

  Temporal information representation: No.

  Deontic Modalities: Only Obligations are considered along with Beliefs, Intentions and Desires (BDI-based approach).

  Legal and Physical ability: No.

  The representation of normative violation: Yes.

  Contrary to Duty Structures: Yes (via the definition of sanctions).

  Normative conflict representation and resolution: Patterns of conflicts between Beliefs, Obligations, Intentions and Desires are presented. Conflict resolution is capable via priorities assignment among mental states.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: Although contact rules are represented as Reiter’s normal default rules to facilitate conflict detection between beliefs, desires, intentions and obligations, no specific discussion on nonmonotonic reasoning is given.

Autonomous and Adaptive Reasoning: No.

Integrated to Semantic Web: No.

Tool: A Prolog prototype is given.

- Wooldridge and van der Hoek: [256], Ågotnes *et al.:* [5], Walther *et al.*: [251]

  Goal and main aspects of interest: Social Contracts for multi agent society regulation. Link among ability and obligations. Explicit Agent Strategies are considered.

  Ontology representation: Yes.

  Temporal information representation: Variations of Alternating-time Temporal Logic (ATL) and Computational Tree Logic (CTL).

  Deontic Modalities: Indexed Deontic Logic operators for Obligations and Permissions are considered (Kripke semantics).

  Legal and Physical ability: No.

  The representation of normative violation: No.

  Contrary to Duty Structures: No.

  Normative conflict representation and resolution: No.

  Auxiliary calculations: No.

  Default, Defeasible and Nonmonotonic Reasoning: No.

  Autonomous and Adaptive Reasoning: No.

  Integrated to Semantic Web: No.

  Tool: No.

Appendix B Standard Deontic Logic and Conflict Management

We mentioned in chapter 9 that if one accepts Standard Deontic Logic then some of the conflict patterns that we presented are mapped onto others. Here we explain this further, and discuss why we find the adoption of such an axiomatization undesirable.

According to SDL the following inter-definability relations hold among operators for Obligation (O), Permission (P) and Prohibition (F):

It is obligatory that a: Oa

It is permitted that a: $Pa \equiv \neg O \neg a$

It is prohibited that a: $Fa \equiv \neg Pa \equiv O \neg a$

In Table B.1 we show the conflict patterns that arise, if one adopts an SDL axiomatization. As can be seen from this table, under SDL inter-definability of operators, we may consider that only three primitive conflict patterns arise, i.e. type A, type D and type E.

We noted in chapter 9 that in the representation of norms as default theories the conflict pattern A never actually arises in an extension, because the derivation of extensions preserves consistency. But, this does not hold for pattern B1, which is possible even as an inter-extension conflict. If we accept the inter-definability of Deontic operators, pattern B1 essentially collapses and becomes pattern A. Hence, B1 will never actually arise in an extension.

Adopting an SDL axiomatization of Deontic operators seems, thus, to lead to some sort of a priori pruning of potential normative conflicts. We find this undesirable, because it seems unrealistic to assume that fewer normative conflicts may arise for agents in a virtual environment, than in real world situations. We prefer, therefore, to maintain a more discriminating representation, which can express more conflict patterns, so that these may be detected and eventually resolved appropriately. [160]

Table B.1 Conflict patterns with and without SDL axiomatization

Pattern Type	Without SDL	With SDL
A	$(Oa$ vs $\neg Oa)$ or $(NNa$ vs $\neg NNa)$	$(Oa$ vs $\neg Oa)$ or $(NNa$ vs $\neg NNa)$
B1	$(Fa$ vs $Pa)$	$(Ob$ vs $\neg Ob)$ if $b \equiv \neg a$ is assumed
B2	$(Fa$ vs $Oa)$	$(Oa$ vs $O\neg a)$ or $(Oa$ vs $Ob)$ if $b \equiv \neg a$ is assumed
C	$(Oa$ vs $O\neg a)$	$(Oa$ vs $Ob)$ if $b \equiv \neg a$ is assumed
D	$(PWa$ vs $Fa)$[23]	-
E	$(Oa$ vs $Ob)$	$(Oa$ vs $Ob)$
F	$(Oa$ vs $\neg Pa)$	$(Oa$ vs $O\neg a)$ or $(Oa$ vs $Ob)$ if $b \equiv \neg a$ is assumed

---

[23] Here, PWa denotes that it is empowered that a

Bibliography

[1] ISO/IEC WD 15414, Open Distributed Processing - Reference Model - Enterprise Viewpoint. 1998.

[2] Akinori Abe. Two-sided hypotheses generation for abductive analogical reasoning. In *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99)*, pages 145–152, Washington, DC, USA, 1999. IEEE Computer Society.

[3] Alan S. Abrahams and Jean M. Bacon. The life and times of identified, situated, and conflicting norms. In *DEON'02: Deontic Logic in Computer Science, 6th International Workshop on Deontic Logic in Computer Science*, pages 3–20, London, England, May 2002.

[4] R.A. Kowalski A.C. Kakas and F. Toni. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770, 1993.

[5] Thomas Agotnes, Wiebe van der Hoek, Juan A. Rodrguez-Aguilar, Carles Sierra, and Michael Wooldridge. Σn the logic of normative systems. In Guido Boella, Leendert W. N. van der Torre, and Harko Verhagen, editors, *Normative Multi-agent Systems*, volume 07122 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum fór Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[6] Marco Alberti, Federico Chesani, Marco Gavanelli, Evelina Lamma, Paola Mello, Marco Montali, and Paolo Torroni. Web service contracting: Specification and reasoning with sciff. In Enrico Franconi, Michael Kifer, and Wolfgang May, editors, *ESWC*, volume 4519 of *Lecture Notes in Computer Science*, pages 68–83. Springer, 2007.

[7] Marco Alberti, Federico Chesani, Marco Gavanelli, Evelina Lamma, Paola Mello, Marco Montali, and Paolo Torroni. Expressing and verifying business contracts with abductive logic programming. *International Journal of Electronic Commerce (IJEC)*, 12(14):9–38, 2008.

[8] Marco Alberti, Marco Gavanelli, Evelina Lamma, Paola Mello, Paolo Torroni, and Giovanni Sartor. Mapping deontic operators to abductive expectations. *Computational & Mathematical Organization Theory*, 12(2-3):205–225, 2006.

[9] Tom Allen and Robin Widdison. Can computers make contracts? *Harvard Journal of Law and Technology*, 9(1), 1996.

[10] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[11] Bernhard Anrig, Rolf Haenni, Jórg Kohlas, and Norbert Lehmann. Assumption-based modeling using ABEL. In Dov M. Gabbay, Rudolf Kruse, Andreas Nonnengart, and Hans Jórgen Ohlbach, editors, *ECSQARU-FAPR*, volume 1244 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 1997.

[12] Grigoris Antoniou. *Nonmonotonic Reasoning*. MIT Press, 1997.

[13] Grigoris Antoniou. Stratification for default logic variants. *International Journal of Intelligent Systems*, 13(9):785–799, 1998.

[14] Grigoris Antoniou. A tutorial on default logics. *ACM Computer Surveys*, 31(4):337–359, 1999.

[15] Grigoris Antoniou. On the dynamics of default reasoning. *Int. J. Intell. Syst.*, 17(12):1143–1155, 2002.

[16] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J. Maher. Representation results for defeasible logic. *ACM Trans. Comput. Log.*, 2(2):255–287, 2001.

[17] Alexander Artikis, Jeremy Pitt, and Marek J. Sergot. Animated specifications of computational societies. In *1st International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1053–1061, Bologna, Italy, July 15-19 2002. ACM.

[18] K. S. Barber, A. Goel, and C. E. Martin. Dynamic adaptive autonomy in multi-agent systems. *Journal of Experimental and Theoretical Arti Intelligence*, 12:129–147, 2000.

[19] K. S. Barber and C. E. Martin. Agent autonomy: Specification, measurement, and dynamic adjustment. In *Proceedings of the Autonomy Control Software Workshop at Autonomous Agents (Agents'99)*, pages 8–15, 1999.

[20] Nick Bassiliades, Grigoris Antoniou, and Ioannis P. Vlahavas. DR-Device: A defeasible logic system for the semantic web. In Hans Jórgen Ohlbach and Sebastian Schaffert, editors, *PPSWR*, volume 3208 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2004.

[21] Russell Beale and Andrew Wood. Agent-based interaction. In Gilbert Cockton, Stephen W. Draper, and George R. S. Weir, editors, *BCS HCI*, pages 239–245. Cambridge University Press, 1994.

[22] Sumit Bhansali and Benjamin N. Grosof. Extending the SweetDeal approach for e-Procurement using SweetRules and RuleML. In Asaf Adi, Suzette Stoutenburg, and Said Tabet, editors, *RuleML*, volume 3791 of *Lecture Notes in Computer Science*, pages 113–129. Springer, 2005.

[23] Guido Boella and Leendert W. N. van der Torre. Contracts as legal institutions in organizations of autonomous agents. In *AAMAS*, pages 948–955. IEEE Computer Society, 2004.

[24] Andrei Bondarenko, Francesca Toni, and Robert A. Kowalski. An assumption-based framework for non-monotonic reasoning. In *LPNMR*, pages 171–189, 1993.

[25] Anthony J. Bonner and Michael Kifer. Concurrency and communication in transaction logic. In *Joint International Conference and Symposium on Logic Programming*, pages 142–156, 1996.

[26] Stefano Borgo. Coalitions in action logic. In Manuela M. Veloso, editor, *IJCAI*, pages 1822–1827, 2007.

[27] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[28] Gerhard Brewka. Cumulative default logic: in defense of nonmonotonic inference rules. *Artif. Intell.*, 50(2):183–205, 1991.

[29] Gerhard Brewka. Adding priorities and specificity to default logic. In *JELIA '94: Proceedings of the European Workshop on Logics in Artificial Intelligence*, pages 247–260, London, UK, 1994. Springer-Verlag.

[30] Gerhard Brewka. Reasoning about priorities in default logic. In *AAAI'94: Proceedings of the twelfth national conference on Artificial intelligence (vol. 2)*, pages 940–945, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.

[31] Jan Broersen, Mehdi Dastani, Joris Hulstijn, Zisheng Huang, and Leendert der van Torre. The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In Jφrg P. Móller, Elisabeth Andre, Sandip Sen, and Claude Frasson, editors, *AGENTS01: Proceedings of the Fifth International Conference on Autonomous Agents*, pages 9–16, Montreal, Canada, 2001. ACM Press.

[32] Scott M. Brown, Eugene Santos Jr., Sheila B. Banks, and Mark E. Oxley. Using explicit requirements and metrics for interface agent user model correction. In *Agents'98*, pages 1–7, 1998.

[33] Henrique Lopes Cardoso and Eugınio C. Oliveira. Towards an institutional environment using norms for contract performance. In Michal Pechoucek, Paolo Petta, and Lαszlσ Zsolt Varga, editors, *CEEMAS*, volume 3690 of *Lecture Notes in Computer Science*, pages 256–265. Springer, 2005.

[34] Henrique Lopes Cardoso and Eugınio C. Oliveira. A contract model for electronic institutions. In J.S. Sichman et al., editor, *The MALLOW Workshop on Coordination, Organization, Institutions and Norms in agent systems (COIN 2007)*, volume 4870 of *LNAI*, pages 27–40. Springer-Verlag Berlin Heidelberg, 2008.

[35] Henrique Lopes Cardoso and Eugınio C. Oliveira. Electronic institutions for b2b: dynamic normative environments. *Artificial Intelligence and Law*, 16(1):107–128, 2008.

[36] J. Carmo and Andrew J.I. Jones. *Handbook of Philosophical Logic*, volume N4, chapter Deontic Logic and Contrary-to-Duties, pages 287–363. Kluwer, 2001.

[37] Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In Victor R. Lesser and Les Gasser, editors, *ICMAS*, pages 41–48. The MIT Press, 1995.

[38] Cristiano Castelfranchi. Guarantees for autonomy in cognitive agent architecture. In *ECAI-94: Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents*, pages 56–70, New York, NY, USA, 1995. Springer-Verlag New York, Inc.

[39] Cristiano Castelfranchi. Again on agents' autonomy: A homage to alan turing. In *Proceedings of 7th International Workshop on Agent Theories Architectures and Languages (ATAL 2000)*, pages 339–342, July 2000.

[40] Cristiano Castelfranchi. Founding agents' "autonomy" on dependence theory. In Werner Horn, editor, *ECAI'00*, pages 353–357. IOS Press, 2000.

[41] Robert L. Causey. EVID: a system for interactive defeasible reasoning. *Decis. Support Syst.*, 11(2):103–131, 1994.

[42] Eugene Charniak and Drew McDermott. *Introduction to artificial intelligence*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1985.

[43] Brian F. Chellas. *Modal Logic: An Introduction*. Cambridge University Press, 1980.

[44] Roderick M. Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis*, 24:33–36, 1963.

[45] Pawel Cholewinski. Stratified default theories. In Leszek Pacholski and Jerzy Tiuryn, editors, *CSL*, volume 933 of *Lecture Notes in Computer Science*, pages 456–470. Springer, 1994.

[46] Pawel Cholewinski. Reasoning with stratified default theories. In V. Wiktor Marek and Anil Nerode, editors, *LPNMR*, volume 928 of *Lecture Notes in Computer Science*, pages 273–286. Springer, 1995.

[47] Laurence Cholvy and Frederic Cuppens. Solving normative conflicts by merging roles. In *ICAIL '95: Proceedings of the 5th international conference on Artificial intelligence and law*, pages 201–209. ACM Press, 1995.

[48] Laurence Cholvy and Frederic Cuppens. Reasoning about norms provided by conflicting regulations. In Henry Prakken and Paul McNamara, editors, *Norms, Logics and Information Systems. New Studies in Deontic Logic and Computer Science*, pages 247–264. IOS Press, Amsterdam, 1998.

[49] Amit K. Chopra and Munindar P. Singh. Contextualizing commitment protocol. In Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss, and Peter Stone, editors, *AAMAS*, pages 1345–1352. ACM, 2006.

[50] Rosaria Conte, Cristiano Castelfranchi, and Frank Dignum. Autonomous norm acceptance. In Jφrg P. Móller, Munindar P. Singh, and Anand S. Rao, editors, *ATAL*, volume 1555 of *Lecture Notes in Computer Science*, pages 99–112. Springer, 1998.

[51] Philip T. Cox and Tomasz Pietrzykowski. A complete, nonredundant algorithm for reversed skolemization. *Theor. Comput. Sci.*, 28:239–261, 1984.

[52] Philip T. Cox and Tomasz Pietrzykowski. Causes for events: Their computation and applications. In Jφrg H. Siekmann, editor, *CADE*, volume 230 of *Lecture Notes in Computer Science*, pages 608–621, New York, NY, USA, 1986. Springer-Verlag New York, Inc.

[53] Richard Craswell. *Contract Law: General Theories, in Encyclopedia of Law and Economics, Volume III, The Regulation of Contracts*, pages 1–24. Northampton, MA: Edward Elgar, 2000.

[54] Aspassia Daskalopulu. Legal contract drafting at the micro-level. In *Law in the Information Society. Proceedings of the 5th International Conference of the IDG-CNRI*, 1998.

[55] Aspassia Daskalopulu. Modeling legal contracts as processes. In *Legal Information Systems Applications, 11th International Workshop on Database and Expert Systems Applications (DEXA'00)*, pages 1074–1079. IEEE Computer Society, 2000.

[56] Aspassia Daskalopulu and T. S. E. Maibaum. Towards electronic contract performance. In A. Min Tjoa and Roland Wagner, editors, *DEXA Workshop*, pages 771–777. IEEE Computer Society, 2001.

[57] Aspassia Daskalopulu and Marek J. Sergot. The representation of legal contracts. *AI Soc.*, 11(1):6–17, 1997.

[58] Hasan Davulcu, Michael Kifer, C. R. Ramakrishnan, and I. V. Ramakrishnan. Logic based modeling and analysis of workflows. In *PODS '98: Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 25–33, New York, NY, USA, 1998. ACM Press.

[59] Hasan Davulcu, Michael Kifer, and I. V. Ramakrishnan. CTR-S: a logic for specifying contracts in semantic web services. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 144–153, New York, NY, USA, 2004. ACM Press.

[60] Johan de Kleer. An assumption-based TMS. *Artif. Intell.*, 28(2):127–162, 1986.

[61] Johan de Kleer. Extending the ATMS. *Artif. Intell.*, 28(2):163–196, 1986.

[62] Johan de Kleer. A general labeling algorithm for Assumption-based Truth Maintenance. In *Proceedings of the 7th National Conference on Artificial Intelligence. St. Paul, MN, August 21-26, AAAI Press / The MIT Press*, pages 188–192, 1988.

[63] Chrysanthos Dellarocas. Contractual agent societies: Negotiated shared context and social control in open multi-agent systems. *Workshop on Norms and Institutions in Multi-Agent Systems, 4th International Conference on Multi-Agent Systems (Agents-2000)*, 2000.

[64] Nirmit Desai, Amit K. Chopra, and Munindar P. Singh. Representing and reasoning about commitments in business processes. In *AAAI*, pages 1328–1333. AAAI Press, 2007.

[65] Nirmit Desai, Nanjangud C. Narendra, and Munindar P. Singh. Checking correctness of business contracts via commitments. In Lin Padgham, David C. Parkes, Jφrg Móller, and Simon Parsons, editors, *AAMAS (2)*, pages 787–794. IFAAMAS, 2008.

[66] Sandra K. Dewitz, Young Ryu, and Ronald M. Lee. Defeasible reasoning in law. *Decis. Support Syst.*, 11(2):133–155, 1994.

[67] Merriam-Webster's Dictionary. http://www.merriam-webster.com/, Online.

[68] Frank Dignum and Ruurd Kuiper. Specifying deadlines with continuous time using deontic and temporal logic, 1998.

[69] Virginia Dignum and Frank Dignum. Modelling agent societies: Co-ordination frameworks and institutions. In Pavel Brazdil and Alpio Jorge, editors, *IPIA*, volume 2258 of *Lecture Notes in Computer Science*, pages 191–204. Springer, 2001.

[70] Virginia Dignum, John-Jules Meyer, Hans Weigand, and Frank Dignum. An organizational-oriented model for agent societies. In *Proceedings of International Workshop on Regulated Agent-Based Social Systems: Theories and Applications. AAMAS'02, Bologna*, 2002.

[71] Virginia Dignum, John-Jules Ch. Meyer, Frank Dignum, and Hans Weigand. Formal specification of interaction in agent societies. In Michael G. Hinchey, James L. Rash, Walt Truszkowski, Christopher Rouff, and Diana F. Gordon-Spears, editors, *FAABS*, volume 2699 of *Lecture Notes in Computer Science*, pages 37–52. Springer, 2002.

[72] Jon Doyle. A truth maintenance system. *Artif. Intell.*, 12(3):231–272, 1979.

[73] Nicole Dunlop, Jadwiga Indulska, and Kerry Raymond. Dynamic conflict detection in policy-based management systems. In *EDOC '02: Proceedings of the Sixth International Enterprise Distributed Oblect Computing Conference*, pages 15–26, Washington, DC, USA, 2002. IEEE Computer Society.

[74] Nicole Dunlop, Jadwiga Indulska, and Kerry Raymond. Methods for conflict resolution in policy-based management systems. In *EDOC '03: Proceedings of the 7th International Conference on Enterprise Distributed Object Computing*, pages 98–111, Washington, DC, USA, 2003. IEEE Computer Society.

[75] Renee Elio. *Common Sense, Reasoning, and Rationality (New Directions in Cognitive Science)*. Oxford University Press US, 2002.

[76] E. Allen Emerson. Temporal and modal logic. pages 995–1072, 1990.

[77] David W. Etherington, Robert E. Mercer, and Raymond Reiter. On the adequacy of predicate circumscription for closed-world reasoning. *Computational Intelligence*, 1:11–15, 1985.

[78] Oren Etzioni and Daniel S. Weld. Intelligent agents on the internet: Fact, fiction, and forecast. *IEEE Expert: Intelligent Systems and Their Applications*, 10(4):44–49, 1995.

[79] Andrew D. H. Farrell, Marek J. Sergot, Mathias Sallı, and Claudio Bartolini. Using the event calculus for tracking the normative state of contracts. *Int. J. Cooperative Inf. Syst.*, 14(2-3):99–129, 2005.

[80] Adina Magda Florea. Assumption-based reasoning of intelligent agents. In *11th International Conference on Control Systems and Computer Science (CSCS'97)*, pages 126–133, 1997.

[81] L. Foner. What is an agent, anyway? a sociological case study. *Agents Memo, Media Lab, MIT*, pages 93–101, 1993.

[82] Nicoletta Fornara and Marco Colombetti. Defining interaction protocols using a commitment-based agent communication language. In *AAMAS*, pages 520–527. ACM, 2003.

[83] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. pages 1070–1080, 1988.

[84] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Comput.*, 9(3/4):365–386, 1991.

[85] Georgios K. Giannikis and Aspassia Daskalopulu. Assumption-based reasoning in dynamic normative agent systems. *invited journal paper under review*.

[86] Georgios K. Giannikis and Aspassia Daskalopulu. Hypothetical nonmonotonic reasoning in e-contracting. *journal paper under review*.

[87] Georgios K. Giannikis and Aspassia Daskalopulu. Normative conflicts in electronic contracts. *journal paper accepted, in revision process*.

[88] Georgios K. Giannikis and Aspassia Daskalopulu. Towards logic-based e-contract modelling and performance monitoring. *journal paper under review*.

[89] Georgios K. Giannikis and Aspassia Daskalopulu. *Web Intelligence and Intelligent Agents*, chapter Autonomous hypothetical reasoning: the case for open-minded agents, ISBN 978-953-7619-X-X, IN-TECH, accepted - in preparation.

[90] Georgios K. Giannikis and Aspassia Daskalopulu. Defeasible reasoning with e-contracts. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2006)*, pages 690–694, Hong Kong, China, 2006. IEEE Computer Society.

[91] Georgios K. Giannikis and Aspassia Daskalopulu. e-Contract representation for defeasible temporal reasoning and conflict management. Technical Report 4936/1, A004.678ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 4936/1, A004.678ΓΙΑ, September 2006.

[92] Georgios K. Giannikis and Aspassia Daskalopulu. E-contracting agents reasoning with assumptions. Technical Report 5343/1, A006.3ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 5343/1, A006.3ΓΙΑ, May 2007.

[93] Georgios K. Giannikis and Aspassia Daskalopulu. Normative conflict detection for e-contracts. In *Mexican International Conference on Artificial Intelligence (MICAI 2007)*, Aguascalientes, Mexico, 4-10 November 2007.

[94] Georgios K. Giannikis and Aspassia Daskalopulu. Normative conflict management with default logic. Technical Report 5154/1, A006.3ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 5154/1, A006.3ΓΙΑ, February 2007.

[95] Georgios K. Giannikis and Aspassia Daskalopulu. The representation of e-Contracts as default theories. In H.G. Okuno and M. Ali, editors, *Proceedings of 19th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2007)*, LNAI 4570, pages 963–973, Kyoto, Japan, 2007. Springer-Verlag Berlin Heidelberg.

[96] Georgios K. Giannikis and Aspassia Daskalopulu. The representation of electronic agreements: a survey of logic-based approaches. Technical Report 5216/1, A004.678ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 5216/1, A004.678ΓΙΑ, March 2007.

[97] Georgios K. Giannikis and Aspassia Daskalopulu. How agents know what to assume when? In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2008)*, pages 538–545, Sydney, Australia, 9 - 12 December 2008. IEEE Computer Society.

[98] Georgios K. Giannikis and Aspassia Daskalopulu. E-contracting agents reasoning hypothetically and nonmonotonically. Technical Report 7122/1, A006.3ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 7122/1, A006.3ΓΙΑ, April 2009.

[99] Georgios K. Giannikis and Aspassia Daskalopulu. Normative conflicts: Patterns, detection and resolution. In *International Conference Web Information Systems and Technologies (WEBIST 2009)*, pages 527–532, Lisbon, Portugal, 23-26 March 2009.

[100] Georgios K. Giannikis and Aspassia Daskalopulu. The role of assumption identification in autonomous agent reasoning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Budapest, Hungary, 10-15 May 2009.

[101] Enrico Giunchiglia, Joohyung Lee, and Vladimir Lifschitz. Causal laws and multi-valued fluents. In *In Proceedings of Workshop on Nonmonotonic Reasoning, Action and Change (NRAC)*, 2001.

[102] Enrico Giunchiglia, Joohyung Lee, Vladimir Lifschitz, Norman McCain, and Hudson Turner. Nonmonotonic causal theories. *Artif. Intell.*, 153(1-2):49–104, 2004.

[103] Enrico Giunchiglia and Vladimir Lifschitz. An action language based on causal explanation: Preliminary report. In *AAAI/IAAI*, pages 623–630, 1998.

[104] Guido Governatori. Representing business contracts in RuleML. *Int. J. Cooperative Inf. Syst.*, 14(2-3):181–216, 2005.

[105] Guido Governatori and Duy Hoang. A semantic web based architecture for e-contracts in defeasible logic. In *1st International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 145–159, 2005.

[106] Guido Governatori, Vineet Padmanabhan, and Antonino Rotolo. Rule-based agents in temporalised defeasible logic. In Qiang Yang and Geoffrey I. Webb, editors, *PRICAI*, volume 4099 of *Lecture Notes in Computer Science*, pages 31–40. Springer, 2006.

[107] Guido Governatori and Antonino Rotolo. Logic of violations: A gentzen system for reasoning with contrary-to-duty obligations. *The Australasian Journal of Logic*, 4:193–215, 2006.

[108] Guido Governatori, Antonino Rotolo, and Giovanni Sartor. Temporalised normative positions in defeasible logic. In *ICAIL*, pages 25–34. ACM, 2005.

[109] Guido Governatori and Paolo Terenziani. Temporal extensions to defeasible logic. In Mehmet A. Orgun and John Thornton, editors, *Australian*

Conference on Artificial Intelligence, volume 4830 of *Lecture Notes in Computer Science*, pages 476–485. Springer, 2007.

[110] Paul W. P. J. Grefen, Karl Aberer, Heiko Ludwig, and Yigal Hoffner. CrossFlow: Cross-organizational workflow management for service outsourcing in dynamic virtual enterprises. *IEEE Data Eng. Bull.*, 24(1):52–57, 2001.

[111] Benjamin N. Grosof. Representing e-commerce rules via situated courteous logic programs in RuleML. *Electronic Commerce Research and Applications*, 3(1):2–20, 2004.

[112] Benjamin N. Grosof, Yannis Labrou, and Hoi Y. Chan. A declarative approach to business rules in contracts: courteous logic programs in xml. In *ACM Conference on Electronic Commerce*, pages 68–77, 1999.

[113] Benjamin N. Grosof and Terrence C. Poon. SweetDeal: representing agent contracts with exceptions using XML rules, ontologies, and process descriptions. In *WWW*, pages 340–349, 2003.

[114] Jaap Hage. Law and defeasibility. *Artif. Intell. Law*, 11(2-3):221–243, 2003.

[115] J. W. Harris. *Law and Legal Science*. Oxford: Clarendon, 1970.

[116] H.L.A. Hart. Kelsen's doctrine of the unity of law. In H.E. Kiefer and M.K. Munitz, editors, *Ethics and Social Justice*, pages 171–199. SUNY Press, 1970.

[117] H. Herrestad and C. Krogh. Deontic logic relativised to bearers and counterparties. In *Anniversary Anthology in Computers and Law, Ed. J. Bing. and O. Torrund*, pages 453 – 522, 1995.

[118] Carl Hewitt. The challenge of open systems: current logic programming methods may be insufficient for developing the intelligent systems of the future. *BYTE*, 10(4):223–242, 1985.

[119] Carl Hewitt. Offices are open systems. *ACM Trans. Inf. Syst.*, 4(3):271–287, 1986.

[120] Carl Hewitt. Toward an open systems architecture. In *IFIP Congress*, pages 389–392, 1989.

[121] Carl Hewitt. Open information systems semantics for distributed artificial intelligence. *Artif. Intell.*, 47(1-3):79–106, 1991.

[122] Carl Hewitt and Peter de Jong. Open systems. In *On Conceptual Modelling (Intervale)*, pages 147–164, 1982.

[123] Carl Hewitt and Peter de Jong. Analyzing the roles of descriptions and actions in open systems. In *AAAI*, pages 162–167, 1983.

[124] H. Hamner Hill. A functional taxonomy of normative conflict. *Law and Philosophy*, 6(2):227–247, August 1987.

[125] Risto Hilpinen. *Normative Conflicts and Legal Reasoning*, pages 191–208. Dordrecht: D. Reidel, 1985.

[126] Yigal Hoffner, Simon Field, Paul W. P. J. Grefen, and Heiko Ludwig. Contract-driven creation and operation of virtual enterprises. *Computer Networks*, 37(2):111–136, 2001.

[127] John F. Horty. Moral dilemmas and nonmonotonic logic. *Journal of Philosophical Logic*, 23(1):35–65, 1994.

[128] John F. Horty. Reasoning with moral conflicts. *Nous*, 37(4):557–605, 2003.

[129] Markus C. Huebscher and Julie A. McCann. A survey of autonomic computing - degrees, models, and applications. *ACM Comput. Surv.*, 40(3), 2008.

[130] J. Jaffar and M. J. Maher. Constraint logic programming: A survey. *Journal of Logic Programming*, 19-20:503–582, 1994.

[131] Mark Jago. Modelling assumption-based reasoning using contexts. In *Workshop on Context Representation and Reasoning (CRR'05)*, 2005.

[132] Paul Taylor Jean-Yves Girard and Yves Lafont. *Proofs and types*. Cambridge University Press, New York, NY, USA, 1989.

[133] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(3):223–250, 1993.

[134] Andrew J.I. Jones and Marek J. Sergot. Deontic logic in the representation of law: Towards a methodology. *Artificial Intelligence and Law*, 1(1):45–64, 1992.

[135] Andrew J.I. Jones and Marek J. Sergot. On the characterisation of law and computer systems: the normative systems perspective. In John-Jules Ch. Meyer and Roel J. Wieringa, editors, *Deontic Logic in Computer Science:*

Normative System Specification, chapter 12, pages 275–307. John Wiley & Sons, Chichester, England, 1993.

[136] Andrew J.I. Jones and Marek J. Sergot. A formal characterisation of institutionalised power. *Journal of the IGPL*, 4(3):427–443, 1996.

[137] Michael Kaminski. A comparative study of open default theories. *Artif. Intell.*, 77(2):285–319, 1995.

[138] Michael Kaminski, Johann A. Makowsky, and Michael L. Tiomkin. Extensions for open default theories via the domain closure assumption. *J. Log. Comput.*, 8(2):169–187, 1998.

[139] Hans Kelsen. *General Theory of Law and State*. Harvard University Press, 1945.

[140] Hans Kelsen. *Pure Theory of Law*. Berkeley: University of California Press., 2nd edition, 1960.

[141] Jeffrey O. Kephart and David M. Chess. The vision of autonomic computing. *IEEE Computer*, 36(1):41–50, 2003.

[142] Steven O. Kimbrough. Formal language for business communication: sketch of a basic theory. *Int. J. Electron. Commerce*, 3(2):23–44, 1998.

[143] John Knottenbelt and Keith Clark. Contract-related agents. In Francesca Toni and Paolo Torroni, editors, *CLIMA VI*, volume 3900 of *Lecture Notes in Computer Science*, pages 226–242. Springer, 2005.

[144] Jórg Kohlas and Paul-Andrι Monney. Probabilistic assumption-based reasoning. In David Heckerman and E. H. Mamdani, editors, *UAI*, pages 485–491. Morgan Kaufmann, 1993.

[145] Martin J. Kollingbaum and Timothy J. Norman. Strategies for resolving norm conflict in practical reasoning. In *ECAI Workshop CEAS*, 2004.

[146] Robert A. Kowalski. A logic-based model for conflict resolution, 2003. http://www.doc.ic.ac.uk/ rak/.

[147] Robert A. Kowalski and Fariba Sadri. The situation calculus and event calculus compared. In *Maurice Bruynooghe (Ed.): Logic Programming, Proceedings of the 1994 International Symposium, November 13-17, MIT Press*, pages 539–553, 1994.

[148] Robert A. Kowalski and Fariba Sadri. Reconciling the event calculus with the situation calculus. *J. Log. Program.*, 31(1-3):39–58, 1997.

[149] Robert A. Kowalski and Marek J. Sergot. A logic-based calculus of events. *New Generation Comput.*, 4(1):67–95, 1986.

[150] P. Radha Krishna, Kamalakar Karlapalem, and Dickson K. W. Chiu. An er^ec framework for e-contract modeling, enactment and monitoring. *Data Knowl. Eng.*, 51(1):31–58, 2004.

[151] Joohyung Lee and Vladimir Lifschitz. Describing additive fluents in action language c+. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 1079–1084. Morgan Kaufmann, 2003.

[152] Ronald M. Lee. Bureaucracies as deontic systems. *ACM Trans. Inf. Syst.*, 6(2):87–108, 1988.

[153] Ioan Alfred Letia and Adrian Groza. Running contracts with defeasible commitment. In Moonis Ali and Richard Dapoigny, editors, *IEA/AIE*, volume 4031 of *Lecture Notes in Computer Science*, pages 91–100. Springer, 2006.

[154] V. Lifschitz. Success of default logic. In *Hector Levesque and Fiora Pirri editors, Logical Foundations for Cognitive Agents: Contributions in Honour of Ray Reiter*, pages 208–212. Springer Verlag, 1999.

[155] Vladimir Lifschitz. On open defaults. In J. Lloyd, editor, *Proceedings of the symposium on computational logic*, pages 80–95. Berlin: Springer-Verlag, 1990.

[156] Fangzhen Lin and Yoav Shoham. Argument systems: A uniform basis for nonmonotonic reasoning. In *KR*, pages 245–255, 1989.

[157] Lars Lindahl. *Position and Change*. Dordrecht: D. Reidel, 1977.

[158] Donald W. Loveland. A linear format for resolution. *Lecture Notes in Mathematics 125*, *Springer-Verlag*, *Berlin*, pages 147 – 162, 1970.

[159] Yann Loyer and Umberto Straccia. Any-world assumptions in logic programming. *Theor. Comput. Sci.*, 342(2-3):351–381, 2005.

[160] Michael Luck and Mark d'Inverno. A formal framework for agency and autonomy. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95)*, pages 254–260, San Francisco, California, USA, June 1995.

[161] Michael Luck and Mark d'Inverno. Autonomy: A nice idea in theory. 1986:351–353, 2000.

[162] Heiko Ludwig and Markus Stolze. Simple obligation and right model (SORM) - for the runtime management of electronic service contracts. *Second International Workshop on Web Services, E-Business and the Semantic Web, WES 2003*, pages 62–76, 2003.

[163] Witold Lukaszewicz. Considerations on default logic: an alternative approach. *Computational Intelligence*, 4:1–16, 1988.

[164] Emil Lupu and Morris Sloman. Conflicts in policy-based distributed systems management. *IEEE Trans. Software Eng.*, 25(6):852–869, 1999.

[165] David Makinson. On the formal representation of rights relations. *Journal of Philosophical Logic*, 15(4):403–425, 1986.

[166] Rafaıl Hernαndez Marvn and Giovanni Sartor. Time and norms: a formalisation in the event-calculus. In *7th International Conference on Artificial Intelligence and Law*, pages 90–99, New York, NY, USA, 1999. ACM Press.

[167] Olivera Marjanovic and Zoran Milosevic. Towards formal modeling of e-contracts. In *EDOC*, pages 59–68. IEEE Computer Society, 2001.

[168] John Anderson Mark Evans and Geoff Crysdale. Achieving flexible autonomy in multi-agent systems using constraints. *Applied Artificial Intelligence*, 6(1):101–126, 1992.

[169] Cheryl E. Martin, Robert H. Macfadzean, and K. Suzanne Barber. Supporting dynamic adaptive autonomy for agent-based systems. In *Proceedings of 1996 Artificial Intelligence and Manufacturing Research Planning Workshop*, pages 112–120. AAAI Press, 1996.

[170] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[171] John McCarthy. Programs with common sense. In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, pages 75–91, London, 1959. Her Majesty's Stationary Office.

[172] John McCarthy. Situations, actions as causal laws. *Memo 2, Stanford University, Artificial Intelligence Project, Stanford, California*, 1963.

[173] John McCarthy. Circumscription - a form of non-monotonic reasoning. *Artif. Intell.*, 13(1-2):27–39, 1980.

[174] John McCarthy. Some expert systems need common sense. In *Proc. of a symposium on Computer culture: the scientific, intellectual, and social impact of the computer*, pages 129–137, New York, NY, USA, 1984. New York Academy of Sciences.

[175] J. J. Ch. Meyer. A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, 29(1):109–136, 1988.

[176] J.-J. Ch. Meyer and R. J. Wieringa. Deontic logic: a concise overview. pages 3–16, 1993.

[177] J.-J. Ch. Meyer, R. J. Wieringa, and F. P. M. Dignum. The role of deontic logic in the specification of information systems. pages 71–115, 1998.

[178] Artur Mikitiuk and Miroslaw Truszczynski. Constrained and rational default logics. In *IJCAI*, pages 1509–1517, 1995.

[179] Nikola Milanovic and Miroslaw Malek. Current solutions for web service composition. *IEEE Internet Computing*, 8(6):51–59, 2004.

[180] Rob Miller and Murray Shanahan. The event calculus in classical logic - alternative axiomatisations. *Electron. Trans. Artif. Intell.*, 3(A):77–105, 1999.

[181] Jonathan D. Moffett and Morris S. Sloman. Policy conflict analysis in distributed system management. *Journal of Organizational Computing*, 1993.

[182] Robert C. Moore. Semantical considerations on nonmonotonic logic. *Artif. Intell.*, 25(1):75–94, 1985.

[183] Erik T. Mueller. *Commonsense Reasoning*. Morgan Kaufmann Publishers, 2006.

[184] Stephen Munzer. Validity of legal conflicts. *Yale Law Journal*, 82:1140–1174, 1973.

[185] Dana S. Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman. Shop2: An htn planning system. *J. Artif. Intell. Res. (JAIR)*, 20:379–404, 2003.

[186] Nils J. Nilsson. *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[187] Donald Nute. Defeasible logic. In Dov Gabbay, Christopher J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Nonmonotonic Reasoning and Uncertain Reasoning*, volume 3, pages 353–395. Oxford University Press, 1994.

[188] Gordon J. Pace, Cristian Prisacariu, and Gerardo Schneider. Model checking contracts - a case study. In Kedar S. Namjoshi, Tomohiro Yoneda, Teruo Higashino, and Yoshio Okamura, editors, *ATVA*, volume 4762 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2007.

[189] Adrian Paschke and Martin Bichler. SLA representation, management and enforcement. In *EEE*, pages 158–163. IEEE Computer Society, 2005.

[190] Adrian Paschke and Martin Bichler. Knowledge representation concepts for automated sla management, int. *Journal of Decision Support Systems*, 2008. In Press.

[191] Adrian Paschke, Martin Bichler, and Jens Dietrich. ContractLog: An approach to rule based monitoring and execution of service level agreements. In *1st International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 209–217, 2005.

[192] Damien Pellier and Humbert Fiorino. Assumption-based planning. In *AISTA'04: International Conference on Advances in Intelligence Systems Theory and Applications*, pages 367–376, Luxembourg-Kirchberg, Luxembourg, November 2004.

[193] Damien Pellier and Humbert Fiorino. Multi-agent assumption-based planning. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 1717–1718. Professional Book Center, 2005.

[194] D. Poole, R. Goebel, and R. Aleliunas. *Theorist: a logical reasoning system for defaults and diagnosis*, pages 331–352. Knowledge Frontier: Essays in the Representation of Knowledge, Springer Verlag, New York, 1987.

[195] David Poole. A logical framework for default reasoning. *Artif. Intell.*, 36(1):27–47, 1988.

[196] David Poole. *Who chooses the assumptions?* Abductive Reasoning, Cambridge: MIT Press., 1996.

[197] David Poole, Alan Mackworth, and Randy Goebel. *Computational intelligence: a logical approach*. Oxford University Press, Oxford, UK, 1997.

[198] Henry Prakken. An argumentation framework in default logic. *Ann. Math. Artif. Intell.*, 9(1-2):93–132, 1993.

[199] Henry Prakken and Giovanni Sartor. The three faces of defeasibility in the law. *Ratio Juris*, 17:118–139, 2004.

[200] Henry Prakken and Marek J. Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1):91–115, 1996.

[201] Alessandro Provetti. Hypothetical reasoning about actions: From situation calculus to event calculus. *Computational Intelligence*, 12:478–498, 1996.

[202] Anand S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In Rudy van Hoe, editor, *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[203] Jinghai Rao and Xiaomeng Su. A survey of automated web service composition methods. In Jorge Cardoso and Amit P. Sheth, editors, *SWSWPC*, volume 3387 of *Lecture Notes in Computer Science*, pages 43–54. Springer, 2004.

[204] D. Reeves, B. Grosof, M. Wellman, and H. Chan. Towards a declarative language for negotiating executable contracts. In *Proc. AAAI-99 Workshop on Artificial Intelligence in Electronic Commerce (AIEC-99).*

[205] Han Reichgelt and Nigel Shadbolt. Planning as theory extension. In *Proceedings of the Seventh Conference on Artificial Intelligence and Simulation of Behaviour (AISB89)*, pages 191–199, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[206] Han Reichgelt and Nigel Shadbolt. A specification tool for planning systems. In *European Conference on Artificial Intelligence (ECAI90)*, pages 541–546, 1990.

[207] Raymond Reiter. On closed world data bases. In *Logic and Data Bases*, pages 55–76, 1977.

[208] Raymond Reiter. A logic for default reasoning. *Artif. Intell.*, 13(1-2):81–132, 1980.

[209] Raymond Reiter. Proving properties of states in the situation calculus. volume 64, pages 337–351, 1993.

[210] Raymond Reiter and Johan de Kleer. Foundations of assumption-based truth maintenance systems: Preliminary report. In *Proceedings AAAI-87 Sixth National Conference on Artificial Intelligence, July 13-17, Seattle, WA. AAAI Press*, pages 183–189, 1987.

[211] Mohsen Rouached, Olivier Perrin, and Claude Godart. A contract-based approach for monitoring collaborative web services using commitments in the event calculus. In Anne H. H. Ngu, Masaru Kitsuregawa, Erich J. Neuhold, Jen-Yao Chung, and Quan Z. Sheng, editors, *6th International Conference on Web Information Systems Engineering*, volume 3806 of *Lecture Notes in Computer Science*, pages 426–434. Springer, 2005.

[212] Mohsen Rouached, Olivier Perrin, and Claude Godart. A contract layered architecture for regulating cross-organisational business processes. In Wil M. P. van der Aalst, Boualem Benatallah, Fabio Casati, and Francisco Curbera, editors, *Business Process Management*, volume 3649, pages 410–415, 2005.

[213] Lamber M. M. Royakkers. *Extending Deontic Logic for the Formalisation of Legal Rules*. Kluwer Academic Publishers, 1998.

[214] Dick W. P. Ruiter. A basic classification of legal institutions. *Ratio Juris*, 10(4):357–371, 1977.

[215] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2003.

[216] M. Sallı and A. Boulmakoul. Integrated contract management, 2002.

[217] Mathias Sallı. Electronic contract framework: An overview. *AAAI-2002 Workshop on AgentTechnologies for B2B Electronic Commerce*, 2002.

[218] Mathias Sallı. Electronic contract framework for contractual agents. In Robin Cohen and Bruce Spencer, editors, *Canadian Conference on AI*, volume 2338 of *Lecture Notes in Computer Science*, pages 349–353. Springer, 2002.

[219] F. Santos and J. Carmo. A deontic logic representation of contractual obligations. *Deontic logic in computer science: normative system specification*, pages 243–257, 1994.

[220] Filipe Santos and Josı Carmo. Indirect action, influence and responsibility. In Mark A. Brown and Josı Carmo, editors, *DEON*, Workshops in Computing, pages 194–215. Springer, 1996.

[221] Filipe A. A. Santos, Andrew J. I. Jones, and Josı Carmo. Action concepts for describing organised interaction. In *HICSS (5)*, pages 373–382, 1997.

[222] Giovanni Sartor. Normative conflicts in legal reasoning. *Artificial Intelligence and Law*, 1(2-3):209–235, 1992.

[223] Giovanni Sartor. A simple computational model for nonmonotonic and adversarial legal reasoning. In *ICAIL '93: Proceedings of the 4th international conference on Artificial intelligence and law*, pages 192–201, 1993.

[224] Paul Scerri, David V. Pynadath, and Milind Tambe. Why the elf acted autonomously: towards a theory of adjustable autonomy. In *AAMAS*, pages 857–864. ACM, 2002.

[225] Torsten Schaub. On constrained default theories. In *17th European Conference on Artificial Intelligence (ECAI'92)*, pages 304–308, 1992.

[226] Torsten Schaub. Variations of constrained default logic. In Michael Clarke, Rudolf Kruse, and Serafn Moral, editors, *ICSQARU*, volume 747 of *Lecture Notes in Computer Science*, pages 310–317. Springer, 1993.

[227] John R. Searle. *The Construction of Social Reality*. Free Press, N.Y., 1995.

[228] Murray Shanahan. The event calculus explained. In *Artificial Intelligence Today*, pages 409–430. 1999.

[229] Murray Shanahan. Review: M. Shanahan, Solving the frame problem. *Artif. Intell.*, 123(1-2):275, 2000.

[230] Munindar P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1):97–113, 1999.

[231] Ellis Solaiman, Carlos Molina-Jimınez, and Santosh K. Shrivastava. Model checking correctness properties of electronic contracts. In Maria E. Orlowska, Sanjiva Weerawarana, Mike P. Papazoglou, and Jian Yang, editors, *ICSOC*, volume 2910 of *Lecture Notes in Computer Science*, pages 303–318. Springer, 2003.

[232] Biplav Srivastava. Web service composition - current solutions and open problems. In *In: ICAPS 2003 Workshop on Planning for Web Services*, pages 28–35, 2003.

[233] Daniel Stamate. Assumption based multi-valued semantics for extended logic programs. In *ISMVL*, page 10. IEEE Computer Society, 2006.

[234] Luc Steels. When are robots intelligent autonomous agents? *Robotics and Autonomous Systems*, 15(1-2):3–9, 1995.

[235] Roy Sterritt, Manish Parashar, Huaglory Tianfield, and Rainer Unland. A concise introduction to autonomic computing. *Advanced Engineering Informatics*, 19(3):181–187, 2005.

[236] Katia P. Sycara. Agents and semantic services: A critical review. In Matthias Klusch, Michal Pechoucek, and Axel Polleres, editors, *CIA*, volume 5180 of *Lecture Notes in Computer Science*, page 35. Springer, 2008.

[237] Ikuo Tahara. Computing scenario from knowledge with preferentially ordered hypotheses. *Systems and Computers in Japan*, 35(4):19–26, 2004.

[238] Yao-Hua Tan and Walter Thoen. Modeling directed obligations and permissions in trade contracts. In *HICSS (5)*, pages 166–175, 1998.

[239] Yao-Hua Tan and Walter Thoen. Using event semantics for modeling contracts. In *HICSS*, page 170, 2002.

[240] Michael Thielscher. Introduction to the fluent calculus. *Electron. Trans. Artif. Intell.*, 2:179–192, 1998.

[241] Yathiraj B. Udupi and Munindar P. Singh. Contract enactment in virtual organizations: A commitment-based approach. In *AAAI*. AAAI Press, 2006.

[242] Wiebe van der Hoek, Wojciech Jamroga, and Michael Wooldridge. A logic for strategic reasoning. In Frank Dignum, Virginia Dignum, Sven Koenig, Sarit Kraus, Munindar P. Singh, and Michael Wooldridge, editors, *AAMAS*, pages 157–164. ACM, 2005.

[243] Bernd van Linder, Wiebe van der Hoek, and John-Jules Ch. Meyer. The dynamics of default reasoning. *Data Knowl. Eng.*, 21(3):317–346, 1997.

[244] Raluca Vartic and Ioan A. Letia. Rules for representing and handling contracts. In *2nd IEEE International Conference on Intelligent Computer Communication and Processing*, pages 65–72, 2007.

[245] Wamberto Weber Vasconcelos, Martin J. Kollingbaum, and Timothy J. Norman. Resolving conflict and inconsistency in norm-regulated virtual organizations. In Edmund H. Durfee, Makoto Yokoo, Michael N. Huhns, and Onn Shehory, editors, *AAMAS*, page 91. IFAAMAS, 2007.

[246] Harko Verhagen. Autonomy and reasoning for natural and artificial agents. In Matthias Nickles, Michael Rovatsos, and Gerhard Weiί, editors, *Agents and Computational Autonomy*, volume 2969 of *Lecture Notes in Computer Science*, pages 83–94. Springer, 2003.

[247] Harko Verhagen and Magnus Boman. Adjustable autonomy, norms and pronouncers. In *AAAI'99*. Spring, 1999.

[248] Lluis Vila and Hajime Yoshino. Time in automated legal reasoning. Technical Report ICS-TR-96-57, 1996.

[249] Georg H. von Wright. Deontic logic. *Mind, Oxford University Press*, 60:1–15, 1951.

[250] Georg H. von Wright. A new system of deontic logic. *Danish Yearbook of Philosophy*, 1:173–182, 1964.

[251] Dirk Walther, Wiebe van der Hoek, and Michael Wooldridge. Alternating-time temporal logic with explicit strategies. In *Theoretical Aspects of Rationality and Knowledge (TARK XI), Brussels, Belgium, June 2007*, 2007.

[252] R. J. Wieringa and J.-J. Ch. Meyer. Applications of deontic logic in computer science: a concise overview. pages 17–40, 1993.

[253] Patrick Henry Winston. *Artificial intelligence*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 1992.

[254] Michael Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley and Sons, Inc., 2002.

[255] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10:115–152, 1995.

[256] Michael Wooldridge and Wiebe van der Hoek. On obligations and normative ability: Towards a logical analysis of the social contract. *J. Applied Logic*, 3(3-4):396–420, 2005.

[257] Lai Xu. A framework for e-markets: Monitoring contract fulfillment. In Christoph Bussler, Dieter Fensel, Maria E. Orlowska, and Jian Yang, editors,

WES, volume 3095 of *Lecture Notes in Computer Science*, pages 51–61. Springer, 2003.

[258] Lai Xu. Monitorable electronic contract. In *CEC*, pages 92–99. IEEE Computer Society, 2003.

[259] Lai Xu and Manfred A. Jeusfeld. Pro-active monitoring of electronic contracts. In Johann Eder and Michele Missikoff, editors, *CAiSE*, volume 2681 of *Lecture Notes in Computer Science*, pages 584–600. Springer, 2003.

[260] Lai Xu and Manfred A. Jeusfeld. Detecting violators of multi-party contracts. In Robert Meersman and Zahir Tari, editors, *CoopIS/DOA/ODBASE (1)*, volume 3290 of *Lecture Notes in Computer Science*, pages 526–543. Springer, 2004.

[261] Pinar Yolum and Munindar P. Singh. Reasoning about commitments in the event calculus: An approach for specifying and executing protocols. *Ann. Math. Artif. Intell.*, 42(1-3):227–253, 2004.