Defeasible Contextual Reasoning in Ambient Intelligence

Antonis Bikakis

Computer Science Department, University of Crete

Institute of Computer Science, Foundation for Research & Technology - Hellas

A thesis submitted for the degree of

PhilosophiæDoctor (PhD)

Heraklion, Greece

June 2009

©Antonis Bikakis, 2009

UNIVERSITY OF CRETE
DEPARTMENT OF COMPUTER SCIENCE

Contextual Defeasible Reasoning in Ambient Intelligence

Dissertation submitted by
Antonis Bikakis
in partial fulfillment of the requirements for
the PhD degree in Computer Science

Author:

____
Antonis Bikakis, University of Crete

Examination Committee:

____
Grigoris Antoniou, Professor, University of Crete, Supervisor

____
Dimitris Plexousakis, Professor, University of Crete

____
Anastasia Analyti, Principal Researcher, Institute of Computer Science, FO.R.T.H.

____
Constantine Stephanidis, Professor, University of Crete

____
Nick Bassiliades, Assistant Professor, Aristotle University of Thessaloniki

____
Yannis Dimopoulos, Associate Professor, University of Cyprus

____
Pavlos Peppas, Associate Professor, University of Patras

Approved by:

____
Panagiotis Trahanias
Chairman of Graduate Studies

Heraklion, July 2009

Abstract

Ambient Intelligence environments consist of various devices that collect, process, change and share the available context information. The imperfect nature of context, the open and dynamic nature of ambient environments, and the special characteristics of the involved devices have introduced new research challenges in the field of Distributed Artificial Intelligence, which have not yet been successfully addressed by current Ambient Intelligence systems.

This thesis proposes a solution based on the Multi-Context Systems paradigm, in which local context knowledge of ambient agents is encoded in rule theories (contexts), and information flow between agents is achieved through mapping rules that associate concepts used by different contexts. To handle imperfect context, we extend Multi-Context Systems with non-monotonic features, such as local defeasible theories, defeasible mapping rules, and a preference ordering over the system contexts. On top of this model, we have developed an argumentation framework that exploits context and preference information to resolve potential conflicts caused by the interaction of ambient agents through their mappings. We also provide an operational model in the form of a distributed algorithm for query evaluation, which is sound and complete with respect to the argumentation framework, as well as three alternative versions of the algorithm, each of which implements a different strategy for conflict resolution. The four strategies, which mainly differ in the type and extent of context and preference information that is used to resolve potential conflicts, have been evaluated in a simulated peer-to-peer system and implemented in Logic Programming in four different logic meta-programs.

Supervisor:

Grigoris Antoniou

Professor of Computer Science

University of Crete

Περίληψη

Τα περιβάλλοντα Διάχυτης Νοημοσύνης αποτελούνται από διάφορες συσκευές οι οποίες συλλέγουν, επεξεργάζονται, αλλάζουν και διαμοιράζονται την διαθέσιμη πληροφορία περιβάλλοντος. Η ατελής γνώση για την πληροφορία περιβάλλοντος, η ανοικτή και δυναμική φύση των περιβαλλόντων Διάχυτης Νοημοσύνης, και τα ειδικά χαρακτηριστικά των εμπλεκόμενων συσκευών έχουν εισάγει νέα ερευνητικά προβλήματα στο πεδίο της Κατανεμημένης Τεχνητής Νοημοσύνης, τα οποία δεν έχουν αντιμετωπιστεί επαρκώς από τα υπάρχοντα συστήματα Διάχυτης Νοημοσύνης.

Η διατριβή αυτή προτείνει μία λύση που βασιζεται στο παράδειγμα των Συστημάτων Πολλαπλών Περιβαλλόντων (Multi-Context Systems), στα οποία η τοπική γνώση των εμπλεκόμενων πρακτόρων αναπαρίσταται ως θεωρίες κανόνων (contexts), και η ροή πληροφορίας μεταξύ των πρακτόρων επιτυγχάνεται με χρήση κανόνων συσχέτισης. Για το χειρισμό της ατελούς γνώσης περιβάλλοντος, επεκτείνουμε τα Συστήματα Πολλαπλών Περιβαλλόντων με χαρακτηριστικά μη-μονότονων συλλογιστικών, όπως τοπικές αναιρέσιμες θεωρίες, αναιρέσιμους κανόνες συσχέτισης και μία σχέση προτίμησης που εφαρμόζεται πάνω στα πολλαπλά περιβάλλοντα. Πάνω από το μοντέλο αυτό, έχουμε αναπτύξει ένα σύστημα συλλογιστικής με χρήση επιχειρημάτων (arguments) που αξιοποιεί πληροφορία περιβάλλοντος και προτιμήσεων για την επίλυση ασυνεπειών που ενδέχεται να προκύψουν κατά τη διάδραση των λογικών πρακτόρων μέσω των κανόνων συσχέτισης. Επίσης, παρέχουμε ένα λειτουργικό μοντέλο στη μορφή ενός κατανεμημένου αλγορίθμου για αποτίμηση επερωτήσεων, ο οποίος είναι ισοδύναμός ως προς τα αποτελέσματά του με το σύστημα επιχειρημάτων, καθώς και τρεις διαφορετικές εκδοχές του αλγορίθμου, καθεμία από τις οποίες υλοποιεί μία διαφορετική στρατηγική για την επίλυση ασυνεπειών. Οι τέσσερεις στρατηγικές, οι οποίες κυρίως διαφέρουν στον τύπο και

την έκταση της πληροφορίας που αξιοποιείται για την επίλυση ασυνεπειών, έχουν αξιολογηθεί σε ένα προσομειωμένο διομότιμο σύστημα, και έχουν υλοποιηθεί σε Λογικό Προγραμματισμό, σε τέσσερα διαφορετικά λογικά μεταπρογράμματα.

Επόπτης Καθηγητής:
Γρηγόρης Αντωνίου
Καθηγητής Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

To my parents, Pantelis and Popi

Acknowledgements

First of all, I would like to thank and express my deep gratitude to my supervisor Professor Grigoris Antoniou for his excellent guidance, supervision and support. I am deeply indebted to him, for his excellent advice, the numerous stimulating discussions that we had, his consistent trust on my abilities, and his inspiring ideas. Throughout all the years since 2003, when he started supervising my master thesis, he has always been a reference of enthusiasm and dedication.

I would also like to thank and express my gratitude to the other members of my doctoral committee, Professor Dimitris Plexousakis and Principal Researcher Anastasia Analyti for their guidance, helpful remarks, and stimulating discussions. During these years, we had an excellent collaboration, and their participation in my doctoral committee had a significant contribution to the successful completion of my PhD.

Many thanks also to the other four members of the committee of my phd defense, Professor Constantine Stephanides, and Associate Professors Nick Bassiliades, Yannis Dimopoulos and Pavlos Peppas, for their helpful remarks and their participation to the committee.

Moreover, I would like to acknowledge the support of the Institute of Computer Science (ICS-FORTH) both financially and in facilities, as well as the University of Crete for granting me the Maria Manasaki legacy's fellowship. Several people in FORTH-ICS and the University of Crete helped me in different ways throughout this PhD, and I feel deeply grateful to them. A special acknowledgement to my friend and colleague at FORTH-ICS, PhD candidate Theodore Patkos, for the inspiring discussions we had, for his constant encouragement throughout all these long years, and for the excellent collaboration we had in several research projects. I would also like to acknowledge graduate students Panayiotis Hassapis and Constantinos Papatheodorou for their help on the implementation and evaluation of the proposed algorithms.

I would like to thank Dr. Guido Governatori for his interesting ideas, advice and inspiring conversations, which greatly helped me at the beginning of my PhD.

I would also like to acknowledge my brother, Vagelis, and all my friends, for their support, patience, encouragement and joyful company. Without them, it would be impossible to get over some hard, stressful times during the PhD. I promise to reciprocate their love and support and be there whenever they need me.

Last, but not least, special thanks to my parents Pantelis and Popi for their unconditional and constant support and encouragement. Devoting the PhD thesis to them and making them feel proud for me are the least that I can do to reciprocate their invaluable love, help and support.

Contents

List of Figures

LIST OF FIGURES

List of Tables

Chapter 1

Introduction

Computing is moving towards pervasive, ubiquitous environments in which devices, software agents and services are all expected to seamlessly integrate and cooperate in support of human objectives, anticipating needs and delivering services in an anywhere, any-time and for-all fashion [119]. *Pervasive Computing* and *Ambient Intelligence* are considered to be key issues in the further development and use of Information and Communication technologies, as evidenced, for example, by the IST Advisory Group [70].

Ambient Intelligence systems aim at providing the right information to the right users, at the right time, in the right place, and on the right device. In order to achieve this, a system must have a thorough knowledge and, as one may say, "*understanding*" of its environment, the people and devices that exist in it, their interests and capabilities, and the tasks and activities that are being undertaken. All this information falls under the notions of *context*.

In the literature one can find various definitions of context. If we restrict the search in the area of human-computer interaction, the most prominent definitions are the following. In the work that first introduces the term "context-aware" [103], context is referred to as "location, identities of nearby people and objects, and changes to those objects". Trying to give a broader definition, Dey et al. [1] describe context as "any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and application, including the user and applications themselves". Gray and Salber [62] have built on the above definition to derive a definition about *sensed context*:

"Sensed context are properties that characterize a phenomenon, are sensed and that are potentially relevant to the tasks supported by an application and/or the means by which those tasks are performed". In an effort to specify the different dimensions of context, Ryan et al. [100] suggest context types of location, environment, identity and time. Dey et al. [1] suggest a similar categorization, but have replaced "environment" with "activity", in order to describe what is occurring in a situation. The four primary pieces of context indicate the types of information necessary for characterizing a situation and their use as indices provide a way for the context to be used and organized.

1.1 Reasoning about Context in Ambient Intelligence

The intelligence of a mechanic system is mainly determined by its reasoning capabilities. The aim of reasoning about context in Ambient Intelligence is to exploit the true meaning of raw context data; to process, combine and ultimately translate the low-level data that is stored in the registers of the sensors into valuable information, based on which the system can determine the state of its context, and react appropriately to certain context changes. The uncertainty and imperfection of context information, and the special characteristics of the entities that operate in Ambient Intelligence environments introduce, however, several challenges in this task.

Henricksen and Indulska in [69] characterize four types of imperfect context information: *unknown*, *ambiguous*, *imprecise*, and *erroneous*. Sensor or connectivity failures (which are inevitable in wireless connections) result in situations, that not all context data is available at any time. When data about a context property comes from multiple sources, then context may become ambiguous. Imprecision is common in sensor-derived information, while erroneous context arises as a result of human or hardware errors.

The entities that operate in an Ambient Intelligence environment are expected to have different goals, experiences and perceptive capabilities. They may use distinct vocabularies; they may even have different levels of sociality. Due to the highly dynamic and open nature of the environment (various entities join and leave the environment at random times) and the unreliable and restricted by the range of the transmitters wireless communications, ambient agents do not typically know a priori all other entities that are present at a specific time instance nor can they communicate directly with all of them.

Overall, the role of reasoning about context in Ambient Intelligence systems includes:

- detecting possible errors in the available context information;
- handling missing values;
- evaluating the quality and the validity of the sensed data;
- transforming the low level raw context data into higher level meaningful information so that it can later be used in the application layer;
- making decisions about the system behavior when certain changes are detected in the system's context.

Considering these requirements and the special characteristics of context and ambient entities, the three main challenges of knowledge management in Ambient Intelligence are to enable:

1. Reasoning with the highly dynamic and imperfect context.
2. Managing the potentially huge piece of context data, in a real-time fashion, considering the restricted computational capabilities of some mobile devices, and the constraints imposed by wireless communications.
3. Collective intelligence, by supporting heterogeneous information sharing, and distributed reasoning with all the available context information.

1.2 Reasoning Limitations of current Ambient Intelligence Systems

Most current Ambient Intelligence systems have failed to efficiently handle all challenges of knowledge management that are discussed above. Their main limitations result from the absence of a reasoning model that may inherently deal with the uncertainty and ambiguity of context knowledge, and from their centralized architecture, which requires the existence of dedicated central entities that collect all the available context information and perform the appropriate reasoning tasks. Below, these two limitations are discussed in more detail.

1.2.1 Reasoning with the Imperfect Context

The reasoning approaches followed so far in Ambient Intelligence systems either neglect to address the problems caused by the imperfect nature of context, or handle them by using heuristics and building additional reasoning mechanisms on top of logic models that cannot inherently deal with the problems of uncertainty, ambiguity and inconsistency. Representative examples of the first category are:

(a) the Description Logic-based approaches followed in [106], where both user profiles and semantic services are modeled as DL predicates, and a set of DL rules is used for semantic matching between users and services; and [114], where three DL reasoners (RACER [66], its commercial successor RacerPro [65], and Pellet [90]) are tested in a real-case application scenario from the smart home domain. The DL reasoning approaches have two significant advantages. They integrate well with the ontology model, which is widely used for the representation of context; and most of them have relatively low computational complexity, which enables them to deal well with situations of rapidly changing context. However, their limited reasoning capabilities are a trade-off that cannot be neglected. They cannot deal with missing or ambiguous information, which is a common case in Ambient Intelligence environments, nor do they provide support for decision making procedures. Thus, although they can be applied in cases that the needs are restricted to retrieving information from the context knowledge base, checking if the available context data is consistent, or deriving implicit higher level ontological knowledge from raw context property values, they cannot serve as a standalone solution for the needs of ambient context-aware applications.

(b) rule-based reasoning approaches, which include the FOL-based approaches followed in [117; 118], where a set of user-defined rules is used for the deduction of higher level context information from an OWL context knowledge base; and the Logic Programming-based approaches followed in [3; 113]. The latter use rules that follow the pattern: if *context attributes* $C_1 ... C_n$ *then context attribute* C_m, which corresponds to a Horn clause, where predicates in the head and in the body are represented by classes and properties defined in the context and application-specific ontologies. The FOL and LP based approaches provide formal models for context reasoning. Moreover, rules are easy to understand and widespread used. However, both approaches share a common deficiency; they cannot handle the highly dynamic, ambiguous and imperfect context information, and suit better in cases, where there is certainty about the quality of the available data.

The second category includes the FOL-based context frameworks of Gaia [97] and SOCAM (Middleware for Context-Aware Mobile Services [63]). Both frameworks use first-order predicates for the representation of context information. In Gaia, in order to resolve conflicts that occur when multiple rules are activated in the same time, they have developed a priority based mechanism, allowing only one rule to fire at each time. In SOCAM, to resolve possible conflicts, they have defined sets of rules on the classification and quality information of the context data, considering that different types of context have different levels of confidence, reliability and quality. The development of such priority mechanisms indeed offers some solutions for the problems of uncertainty and ambiguity of context, adding however additional complexity to the reasoning tasks. Moreover, these solutions are rather restricted to meet the needs of the specific systems / applications. For the general needs of Ambient Intelligence systems, a more general and formal approach that can inherently deal with missing, uncertain, inaccurate and ambiguous information is certainly required.

1.2.2 Reasoning with the Distributed Context

So far, most Ambient Computing frameworks have been based on fully centralized architectures for managing context. The common approach followed in such systems dictates the existence of a central entity, which is responsible for collecting the available context data from all sensors and ambient agents operating in the same environment, and for all the required reasoning tasks, which may include transforming the imported context data in a common format, deducing higher-level context information form the raw context data, and taking context-dependent decisions for the behavior of the system. Representative examples of such systems are: (a) the *Context Broker Architecture* (CoBrA, [41]); (b) the *Context Awareness Framework* presented in [47]; the Contextual Guiding Platform [92]; the *Gaia* Context-Aware Framework [97]; the *eWallet* architecture of *myCampus* project [50]; the Semantic Context-Aware Access Control Framework presented in [113]; the *AmbieSense* Context Management Framework [76]; and the *ec(h)o* audio museum guide [68].

The need for more decentralized approaches in Ambient Intelligence systems has recently led several research teams to deploy methods and techniques from Distributed

Artificial Intelligence. One such approach is followed in *sTuples* [75]. This framework extends Tuple Spaces [52] using Semantic Web technologies to represent and retrieve tuples from a Tuple Space. The Tuple Space model uses a logically shared memory, where producers add tuples to a common space, while consumers read or extract tuples from the space using a search template. Similar approaches, which combine Semantic Web technologies and shared memory models to support asynchronous communications in ambient environments, have been adopted in *Semantic Spaces* [78], and in the context management framework presented in [77]. The latter follows a *blackboard*-based approach. A mobile terminal system uses a central context manager, which stores context information from any available source. Clients can directly query the manager to gain context information, subscribe to various context change notification services, or use higher level contexts transparently. In the latter case, the context manager assigns the reasoning tasks to dedicated recognition services.

The OWL-SF framework [87] follows a more decentralized approach. It combines the OMG's *Super Distributed Objects* (SDO) technology and the OWL language to allow the distribution of semantically annotated services for the needs of ambient context-aware systems. SDOs are logical representations of hardware and software entities that are used to enable distributed interoperability. The proposed framework integrates two basic building blocks, OWL-SDOs and Deduction Servers. The OWL-SDOs are semantic extensions of SDOs; they use the OWL language to describe their status, services and communication interface. Deduction servers are specific OWL-SDOs that provide reasoning services. They contain a deduction engine coordinating reasoning tasks, an RDF inference layer providing rule reasoning support and an OWL-DL reasoner. Besides providing reasoning support, they are responsible for collecting the status of SDOs published using the OWL format, and for building an integrated OWL description accessible to reasoning.

The main feature that distinguishes the latter approach is the lack of one central reasoning or control entity; it is fully decentralized. Collecting the reasoning tasks in a central entity certainly has many advantages; it can achieve better control, and better coordination between the various entities that have access to the central entity. Blackboard-based and shared-memory models have been thoroughly studied and used in many different types of distributed systems and have proved to work well in practice. The requirements are, though, much different in this setting. Context may not be restricted to a small room, office or apartment; cases of broader areas must also be considered. The communication with a central entity is not always guaranteed, and wireless communications are typically unreliable and restricted by the range of the transmitters. Thus, a fully distributed scheme is a necessity. The OWL-SF framework is a step towards this direction, but certainly not the last one. In order to deal with the challenging issues that arise in Ambient Intelligence environments, some of the assumptions that they make must be relaxed. For example, different entities should not be expected to use the same representation and reasoning models, and the existence of dedicated reasoning machines cannot always be assumed.

The reasoning approaches, along with the aim of the systems referenced in this section are summarized in Table 1.1.

1.3 Motivating Scenarios from the Ambient Intelligence Domain

Below, there is a description of three use case scenarios from the Ambient Intelligence domain. The aim of these scenarios is to highlight the challenges of contextual reasoning in Ambient Intelligence.

1.3.1 Context-Aware Mobile Phone in an Ambient Classroom

The first scenario involves a context-aware mobile phone that has been configured by Dr. Amber to make decisions about whether it should ring (in case of incoming calls) based on his preferences and context. Dr. Amber has the following preferences: His phone should ring in case of an incoming call, unless it is in silent mode or he is giving a lecture.

Consider the case that Dr. Amber is currently located in the 'RA201' university classroom. It is class time and he has just finished with a course lecture, but he still remains in the classroom reading his emails on his laptop. The mobile phone receives an incoming call, while it is in normal mode.

The phone cannot decide whether it should ring based only on its local context knowledge, which includes knowledge about incoming calls and the mode of the phone, as it is not aware of other important context parameters (e.g. Dr. Amber's current

Table 1.1: Reasoning Approaches followed by Current Ambient Intelligence Systems

System	Reasoning	Architecture	Aim
Semantic Mobile Environment [106]	DL	distributed (agent-based)	Semantic Service matchmaking
Context-Aware Door Lock [114]	DL	centralized	automatic door-lock
CONON Prototype [118]	DL+FOL	centralized	context-aware services
Semantic Space [117]	FOL	centralized	smart space mobile services
CARE [3]	DL+LP	centralized (middleware)	service adaptation
Context-Aware Access Control Framework [113]	DL+LP	centralized	policy evaluation
Gaia [97]	FOL	centralized	context-aware services
SOCAM [63]	FOL	centralized (middleware)	middleware for mobile services
CoBrA [41]	FOL	centralized (middleware)	middleware for mobile services
Context Awareness Framework [47]	RDQL	centralized	service prioritization
CG Platform [92]	RQL	centralized	location-based services
eWallet [50]	FOL	centralized (agent-based)	context-aware services
AmbieSense [76]	CBR	centralized	context management
ec(h)o system [68]	FOL	centralized	audio museum guide
sTuples [75]	DL	decentralized shared memory	mobile services
Semantic Spaces [78]	DL	decentralized shared memory	information sharing
Context Management Framework [77]	Bayesian	decentralized (blackboard)	information sharing notification services
OWL-SF [87]	DL	distributed (SDOs)	distributed services

activity). Therefore, it attempts to contact through the wireless network of the university other ambient agents that are located nearby, import from them further context information, and use this information to reach a decision.

In order to determine whether Dr. Amber is currently giving a lecture, the mobile phone uses two rules. The first rule states that if at this time there is a scheduled lecture, and Dr. Amber is located in a university classroom, then he is possibly giving a lecture. Information about scheduled events is imported from Dr. Amber's laptop, while information about his current location is imported from the wireless network localization service. The second rule states that if there is no class activity in the classroom, then Dr. Amber is rather not giving a lecture. Information about the state of the classroom is imported from the classroom manager (a stationary computer installed in the 'RA201' classroom).

Dr. Amber's personal laptop contains his personal calendar. Using this information, it can infer that at present there is a scheduled class event. The localization service possesses knowledge about Dr. Amber's current position (actually about the position of his mobile phone). In this case, it provides information that Dr. Amber is currently located in 'RA201'. The classroom manager possesses knowledge about the state of the classroom. Specifically, it provides information that the classroom projector is off, and imports information about the presence of people in the classroom from an external person detection service. In this case, the service detects only one person (Dr. Amber) in the classroom. Based on this information, the classroom manager infers that there is no class activity.

The overall information flow in the scenario is depicted in Figure 1.1. Eventually, the mobile phone will receive ambiguous context information from the various ambient agents operating in the classroom. Information imported from Dr. Amber's laptop and the localization service leads to the conclusion that Dr. Amber is currently giving a lecture. On the other hand, information imported from the classroom manager leads to the contradictory conclusion that Dr. Amber is not giving a lecture. To resolve this conflict, the mobile phone must be able to evaluate the information it receives from the various sources. For example, in case it is aware that the information derived from the classroom manager is more accurate than the information imported from Dr. Amber's laptop, it will determine that Dr. Amber is not currently giving a lecture, and therefore will reach the 'ring' decision.

Figure 1.1: Context Information Flow in the Scenario

1.3.2 Ambient Intelligence Home Care System

The second scenario takes place in an apartment hosting an old man, Mr. Jones. Mr. Jones, a 80 year old widower, is living alone in his apartment, while his son resides in close proximity. A nurse visits Mr. Jones 8 to 10 hours daily, while his son also visits him for some hours every couple of days. Mr Jones' apartment is equipped with an Ambient Intelligence Home Care System, which consists of:

- A position tracking system, which localizes Mr. Jones in the apartment.
- An activity tracking system, which monitors the activities carried out by Mr. Jones; activity can take values such as *sitting*, *walking*, *lying*, etc.
- A data monitoring system, in the form of a bracelet, which collects Mr. Jones' basic vital information, such as pulse, skin temperature and skin humidity.
- A person detection system, which is able to recognize Mr. Jones, his son and the nurse.
- An emergency monitoring system, identifying emergency situations. This system has a wired connection with the position tracking system, the activity tracking system, the person detection system and the emergency telephony system, and a wireless connection with the data monitoring bracelet.

- An emergency telephony system, which makes emergency calls to Mr. Jones' son in case of emergency.

Assume that neither the nurse nor Mr. Jones' son are located in the apartment, and Mr Jones is walking through a hall of the apartment to his bedroom. He suddenly stumbles, falls down and loses his consciousness, while the data monitoring bracelet that he wears in his wrist is damaged, transmitting erroneous data to the emergency monitoring system.

The emergency telephony system is configured to determine about whether it should make an emergency call to his son using the following rule: 'If an emergency situation is detected, and neither the nurse nor Mr. Jones' son are located in the house, then make an emergency call'. The detection of emergency situations is a responsibility of the emergency monitoring system, while the person detection system is responsible for detecting Mr. Jones, his son and the nurse in the house.

The emergency monitoring system uses the following rules for determining emergency situations: (a) Any abnormal situation is an emergency situation; (b) If Mr. Jones' temperature, skin humidity and pulse have normal values then there is no case of emergency situation; (c) In case Mr. Jones is lying in a place different than his bed, then this is an abnormal situation. Information about Mr. Jones' physical situation is imported from the data monitoring bracelet. In the specific case described above, the bracelet is damaged and keeps erroneously transmitting normal values about Mr. Jones' temperature, skin humidity and pulse. Knowledge about Mr. Jones' current activity (lying) is possessed by the activity tracking system, while the position tracking system is aware of Mr. Jones' current position (hall).

Using information imported from the data monitoring bracelet, and rules a and b, the emergency monitoring system may conclude that this is not an emergency situation. However, based on the information imported from the activity and position tracking systems and rules a and c, the emergency monitoring system reaches the contradictory conclusion that this is an emergency situation. As the data monitoring bracelet is considered more prone to damage than the activity and position tracking systems, and the wired connections between the emergency monitoring system with the activity and position tracking systems are more reliable than the wireless connection with the data monitoring bracelet, the emergency monitoring system determines that this is an emergency situation, and the telephony system reaches the 'emergency call' decision.

1.3.3 Mushroom Hunting in an Ambient Natural Park

The third scenario takes place in an Ambient Intelligence environment of mushroom hunters, who collect mushrooms in a natural park in North America. The hunters carry mobile devices, which they use to communicate with each other through a wireless network, in order to share their knowledge on edible and non-edible mushrooms.

People interested in picking mushrooms typically do not know every species and family of mushrooms in detail. They know that a deadly mushroom can be very similar to an edible one, e.g., the "*amanita phalloides*" (deadly) and the "*amanita caesarea*" (edible and one of the best mushrooms) that look very much alike. In general, a mushroom hunter has to respect certain rules imposed by the natural park legislation such as the limited quantity of mushrooms that can be picked. Due to the limitation on the allowed quantity, there is the need of establishing the specie of an unknown mushroom during the picking itinerary instead of bringing the picked ones to an expert and discovering, after some days, that the picking has been useless due to the high number of non-edible picked mushrooms. Furthermore, the picking has not been simply useless but it has also vainly cheated the ecosystem of a part. Moreover, even in the case of an irrelevant quantity of non-edible picked mushrooms, it might happen that a small chunk of a deadly mushroom (e.g., "*amanita phalloides*" also known as *The Death Cap*) mixes with edible ones and accidentally eaten. By keeping in mind the above discussed motivations, let us consider the scenario in which a mushroom hunter, Adam, finds an interesting mushroom but it is unclear if it is edible.

Suppose that the mushroom in question has the following characteristics: It has a stem base featuring a fairly prominent sack that encloses the bottom of the stem (*volva*), and a pale brownish cap with patches, while the margin of the cup is prominently lined, and the mushroom does not have a ring (*annulus*).

Adam has some knowledge on the description of specific species, such as the *Destroying Angel*, the *Death Cap* and the *Caesar's Mushroom*. He also knows that the first two of them are poisonous, while the third one is not. However, the description of the mushroom in question does not fit with any of these species, so Adam cannot determine about whether this mushroom is poisonous. He decides to exploit the knowledge of other mushroom hunters in the Ambient Natural Park, and uses the wireless network to contact other hunters that are located nearby. His wireless device establishes connection with the devices of three other hunters.

The first one of the three other hunters, Bob, uses a generic rule, which states that mushrooms with a volva are non-edible. The second hunter, Chris, has knowledge of some specific species that are not toxic, including *springtime amanita*, but does not know their distinct characteristics. The third hunter, Dan, on the other hand, also uses a very generic rule, which states that amanitas are typically dangerous. Using the wireless network, Chris establishes a connection with another hunter, Eric, who knows how *amanita velosa* (a formal name for *springtime amanita*) looks like, and the description of this specific specie fits exactly the description of the mushroom in question.

In this scenario Adam has three options: Using the knowledge of Bob, he will reach the conclusion that the mushroom is poisonous, and therefore he should not pick it. Using the knowledge of Dan, he will reach the same decision. The third option is to use the combined knowledge of Chris and Eric. In the latter case, he will reach a different decision; he will determine that the mushroom is not dangerous, and therefore he may pick it. Being aware that Chris and Eric possess more specialized knowledge than Bob and Dan, he will determine to give priority to the third option determining that the mushroom in question is not poisonous.

1.3.4 Common Characteristics of the Three Scenarios

The three scenarios described above share some common characteristics with respect to the distribution of context knowledge, the nature of this knowledge, and the relations that exist between the various involved ambient agents. Specifically, the following assumptions have been implicitly made:

- In each case there is an available means of communication through which an ambient agent can communicate and exchange context information with a subset of the other available ambient agents.
- Each ambient agent is aware of the type of knowledge that each of the other agents that it can communicate with possesses, and has specified how part of this knowledge relates to its local knowledge.

- Each ambient agent is aware of the quality of context information that it imports from other ambient agents.
- Each ambient agent has some computing and reasoning capabilities that it may use to make certain decisions based on its local and imported context information.
- Each ambient agent is willing to disclose and share part of its local knowledge.

The challenges of reasoning with the available context information and making correct context-dependent decisions in the described scenarios include:

- Local context knowledge may be incomplete, meaning that none of the agents involved in the scenarios described above has immediate access to all the available context information.
- Context knowledge may be ambiguous; in all the three scenarios, there is one case that an ambient agent receives mutually inconsistent information from two or more other agents.
- Context knowledge may be imprecise; e.g. in the first scenario the knowledge about Dr. Amber's schedule possessed by his laptop is not accurate.
- Context knowledge may be erroneous; e.g. in the second scenario, the values for Dr. Jones' temperature, skin humidity and pulse that are transmitted by the bracelet are not valid.
- Each agent may use its own vocabulary to describe its context; e.g. in the third scenario two hunters may use a different name for the same specie of amanita.
- The computational capabilities of most of the devices that are involved in the three scenarios are restricted, so the overhead imposed by the reasoning tasks must not be too heavy.
- The communication load must not also be too heavy, so that the system can quickly make a decision, taking into account all the available context information that is distributed between the ambient agents. Communication load refers not only to the required number of messages exchanged between the involved devices, but also to the size of these messages.

1.4 Thesis Contribution

This thesis presents a fully distributed approach for reasoning about context in Ambient Intelligence environments. The involved entities are modeled as autonomous logic-based agents, the knowledge possessed by an ambient entity as a local context rule theory, and the associations between the context knowledge possessed by different ambient agents as mappings between their respective context theories. To support cases of missing or inaccurate local context knowledge, contexts are represented as local theories of Defeasible Logic; and to handle inconsistencies caused by the interaction of mutually inconsistent contexts, the *Multi-Context Systems* (*MCS*) model is extended with defeasible mapping rules, and with a preference order on the system contexts.

Reasoning with the distributed context information is based on an argument-based approach. The arguments have a local range, in the sense that each one uses knowledge of a single context, but are interrelated through mapping rules. Conflicts that arise from the interaction of mutually inconsistent contexts are captured through *attacking* arguments, and conflict resolution is achieved by *ranking* arguments according to the preference order on the system contexts.

Finally, an operational model in the form of a distributed algorithm for query evaluation is provided. The algorithm meets the requirement for low computational load, while it is sound and complete with respect to the argumentation framework. Additionally to the algorithm that implements the proposed argumentation framework, three alternative versions are also described. Each of them implements a different strategy for conflict resolution with respect to the type and extend of context and preference information used to resolve the potential conflicts.

1.5 Outline

1.5.1 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2 provides background information on contextual reasoning and on argumentation systems. Specifically, it describes the intuition behind Multi-Context Systems, discusses relevant works on non-monotonic MCS and peer data management systems, and presents the most prominent works on non-monotonic and preference-based argumentation systems.

Chapter 3 describes the argumentation framework that enables reasoning with the imperfect and distributed context information. Firstly, it presents the MCS-based representation model, and then provides its semantic characterization using arguments.

Chapter 4 provides the operational model in the form of a distributed algorithm for query evaluation, and studies its formal properties regarding its termination, complexity, soundness and completeness with respect to the argumentation framework. It also describes a process, which unifies the distributed theories in an equivalent global defeasible theory.

Chapter 5 presents three alternative strategies for conflict resolution, discusses their differences, and describes how they are implemented in three different versions of a distributed algorithm for query evaluation.

Chapter 6 presents the process and the results of an evaluation of the four strategies in a simulated P2P system. It also describes the implementation of the algorithms in Logic Programming.

Chapter 7 summarizes the main points of the thesis and discusses potential extensions of this work.

1.5.2 Relevant Publications

Most parts of this study have already been published in the following conference and journal articles

- [29] presents the results of a survey that we conducted on semantics-based approaches for contextual reasoning in Ambient Intelligence (the main results of this survey appear in Section 1.2 of this chapter)
- [25] describes the main intuitions behind our approach
- [23] presents how our approach can be applied to the three different use case scenarios that we discuss in Section 1.3 of this chapter
- [27] describes the argumentation framework, which we present in this thesis in Chapter 3

- [24] and [26] present the representation model and the distributed algorithm query evaluation, which we describe in Chapter 4
- [22] presents the main features of the four alternative strategies for conflict resolution, which appear in Chapter 5 of this thesis, while [28] presents and discusses the results of the evaluation of the four strategies (Chapter 6)
- [19] contains several different parts of this study, including the general ideas, the representation model, the distributed algorithms query evaluation and the application of our approach in the Ambient Intelligence domain, while it also provides a proof theoretic formalization

The following two articles are currently under review:

- [21] is an extended version of the LPNMR paper, which additionally provides some additional properties of the argumentation framework and of the algorithms for distributed query evaluation and proves the soundness and completeness of the algorithms with respect to the argumentation framework
- [20] provides a more detailed description of the four different strategies for conflict resolution, their implementation and a discussion on the results of the evaluation

Chapter 2

Background

This chapter provides background information on Contextual Reasoning and on Argumentation Systems. Specifically, it provides general information about formalizations of context in AI, and describes the intuitions behind Multi-Context Systems, which is the underlying model of our approach. It also presents non-monotonic extensions of Multi-Context Systems and Peer Data Management Systems, which can be viewed as special cases of Multi-Context Systems, and discusses their main limitations with respect to Ambient Intelligence. The second section presents the most prominent works on non-monotonic Argumentation Systems, and discusses recently proposed approaches on the integration of preferences in Argumentation Systems.

2.1 Contextual Reasoning in Artificial Intelligence

The notions of *context* and *contextual reasoning* were first introduced in Artificial Intelligence by McCarthy in [84], as an approach for the problem of *generality*. In the same paper, he argued that the combination of non-monotonic reasoning and contextual reasoning would constitute an adequate solution to this problem. Since then, two main formalizations have been proposed to formalize context: the propositional logic of context (*PLC* [37; 85]), and the Multi-Context Systems introduced in [57], which later became associated with the Local Model Semantics proposed in [54]. Multi-Context Systems has been argued to be most adequate with respect to the three properties of contextual reasoning as these were formulated by [17] (*partiality*, *approximation*, *proximity*) and has been shown to be technically more general than PLC [104].

2.1.1 Applications of Context and Contextual Reasoning

Context reasoning mechanisms have already been used in practice in various distributed reasoning applications, and are expected to play a significant role in the development of the next generation Artificial Intelligence Applications. A first prominent example was the design of the CYC common sense knowledge base as a collection of interrelated partial "*microtheories*" [64; 79]. CYC's enormous knowledge base of assertions, which may represent a particular set of surrounding circumstances, relevant facts, IF-THEN rules, and background assumptions, is divided into many different contexts; every fact or rule is in some particular context. Whenever Cyc is asked a question, or has to do some reasoning, the task is always done in some particular context. In this way, Cyc's "*microtheories*" provide a mechanism for focusing on relevant information during problem solving.

The most representative example of distributed interrelated knowledge bases is, however, the *Semantic Web* [11; 112]. The Semantic Web is an extension of the current Web, in which information is given well-defined meaning, better enabling computers and people to work in cooperation. For the semantic annotation of web information, Semantic Web uses online *ontologies* - formal descriptions of particular domains, which can potentially be published and edited by anyone willing to do so. As a result, ontologies are highly scattered and heterogeneous. To deal with this problem, recent studies have focused on the development of languages that allow for the expression of *contextualized ontologies*, with the Distributed Description Logics [33] and C-OWL [35] being the most prominent examples.

The use of Multi-Context Systems as a means of specifying and implementing agent architectures has been proposed in [91] and [102]. Both studies propose *breaking* the logical description of an agent into a set of contexts, each of which represents a different component of the architecture, and the interactions between these components are specified by means of bridge rules between the contexts. In [91], they follow this approach to simplify the construction of a belief/desire/intention (BDI) agent, while in [102] they extend it to handle more efficiently implementation issues such as grouping together contexts in modules, and enabling inter-context synchronization.

Other fields that the notions of context and contextual reasoning have been successfully applied include: multi-agent systems [15; 42], modeling dialog, argumentation and information integration in electronic commerce applications [91], commonsense reasoning [34], reasoning about beliefs [16; 46; 53; 55; 56] and reasoning with viewpoints [12].

2.1.2 Multi-Context Systems

A simple illustration of the intuitions underlying Multi Context Systems is provided by the so-called '*magic box*' example:

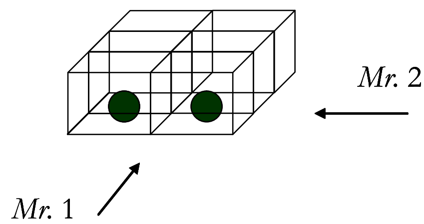

Figure 2.1: A magic box.

As depicted in Figure 2.1 above, Mr.1 and Mr.2 look at a box, which is called "*magic*" because neither of the observers can make out its depth. Both Mr.1 and Mr.2 maintain a representation of what they believe to be true about the box. Mr.1's beliefs may regard concepts that are completely meaningless for Mr.2, and vice versa. For example, Mr.2 could believe the central section of the box to contain a ball. From Mr.1's viewpoint however, the box does not have a central section, so any statement about whether it contains a ball or not is meaningless for him. Mr.1 and Mr.2 may also have concepts in common, but in any case their respective interpretations of those concepts are independent. For example, "the right section of the box contains a ball" is a meaningful statement for both Mr.1 and Mr.2. But it is perfectly conceivably that Mr.1 believes the right section of the box to be empty, while Mr.2 believes it to contain a ball, and vice versa. The bottom line is that both observers have their own local language in which they express their beliefs.

Another important notion is that the observers may have (partial) access to each other's beliefs about the box. For example, Mr.1 may have access to the fact that Mr.2 believes the box to contain a ball. Mr.1 may interpret this fact in terms of his own language, and adapt his beliefs accordingly. We think of this mechanism as an information flow among different observers.

The intuitions underlying Multi-Context Systems can be summarized in the following points.

- A *context* is a subset of an individual global state, or - more formally - a partial and approximate theory of the world from some individual's perspective [55].
- Reasoning with multiple contexts is a combination of:
  - *local reasoning*, which is performed using knowledge of a single context. Local conclusions in a given context derive from a set of axioms and inference rules that model local context knowledge, and which constitute a small subset of the entire knowledge.
  - *distributed reasoning*, which also that takes into account the possible relations between local contexts. These relations result from the fact that different contexts actually constitute different representations of the same world. They are modeled as inference rules (known as *mapping* or *bridge* rules) with premises and consequences in different contexts, and enable information flow between related contexts.
- The relationship between different contexts can be described only to a partial extent. In other words, no context can be 'fully' translated into another, as each of them may encode assumptions that are not fully explicit.

As a result from the characteristics discussed above, different contexts are expected to use different languages and inference systems, and although each context may be locally consistent, global consistency cannot be required or guaranteed. The most critical issues of reasoning in Multi-Context Systems are; (a) the *heterogeneity* of local context theories with respect to the language and inference system that they use; and (b) the potential conflicts that may arise from the interaction of contexts through their mappings.

2.1.3 Non-monotonic Contextual Reasoning

Multi-Context Systems have been the basis of two recent studies that were the first to deploy non-monotonic features in contextual reasoning:

- the non-monotonic rule-based MCS framework [98], which supports default negation in the mapping rules allowing to reason based on the absence of context information

- the multi-context variant of Default Logic (ConDL [36]), which models bridge relations between different contexts as *default rules*.

Both approaches support many of the characteristics of contextual reasoning in Ambient Intelligence environments that we discuss in Chapter 1. Specifically, additionally to the three fundamental dimensions of contextual reasoning (partiality, approximation and perspective) that the generic MCS model inherently supports, both approaches support reasoning with incomplete local information using default negation in the body of the mapping rules. Furthermore, Contextual Default Logic handles ambiguous context using default mapping rules. The case that context A imports conflicting context information from contexts B and C through A's mapping rules, is modeled using the different extensions of the theory that includes A's local theory, A's mapping (default) rules, the local theories and (possibly) the mappings of contexts B and C, and (possibly) other context theories that B and C are connected to through their mapping rules. Finally, comparing to [98], the ConDL approach has the additional advantage that is closer to implementation due to the well-studied relation between Default Logic and Logic Programming.

However, there are still some issues that Contextual Default Logic cannot efficiently handle. Specifically, it does not provide ways to model the quality of the imported context information, nor the preference between two different information sources. In other words, it does not include any notion of priority, not allowing to resolve potential conflicts that may arise while importing context information from two different sources. Furthermore, computing extensions or checking if a formula is in one or in all extensions of a Default theory has been showed to be a complex computational problem [59; 107], and would add a too heavy computational overhead to the devices operating in Ambient Intelligence environments, which typically have limited computational capabilities.

2.1.4 Peer Data Management Systems

Peer data managements systems can be viewed as special cases of Multi-Context Systems, as they consist of autonomous logic-based entities (*peers*) that exchange local information using *bridge rules*. A key issue in formalizing data-oriented Peer-to-Peer systems is the semantic characterization of *mappings* (bridge rules). One approach, followed in [18; 67], is the first-order logic interpretation of Peer-to-Peer systems. [39] identified several drawbacks with this approach, regarding modularity, generality and decidability, and proposed new semantics based on epistemic logic. A common problem of both approaches is that they do not model and thus cannot handle inconsistency. Franconi *et al.* in [48] extended the autoepistemic semantics to formalize local inconsistency. The latter approach guarantees that a locally inconsistent database base will not render the entire knowledge base inconsistent. A broader extension, proposed by Calvanese *et al.* in [38], is based on non-monotonic epistemic logic, and enables isolating local inconsistency, while also handling peers that may provide mutually inconsistent data. It guarantees that in case of importing knowledge that would render the local knowledge inconsistent, the local peer knowledge base remains consistent by discarding a minimal amount of the data retrieved from the other peers. The propositional Peer-to-Peer Inference System proposed by Chatalic *et al.* in [40] extends the distributed reasoning methods of [2] to deal with conflicts caused by mutually inconsistent information sources, by detecting them and reasoning without them. Finally, based on the latter study, [31] proposes algorithms for inconsistency resolution in Peer-to-Peer Query Answering exploiting a preference relation on the peers.

The three latter approaches ([38], [40] and [31]), have some common characteristics that meet many of the requirements of Ambient Intelligence environments discussed in Chapter 1. Specifically, they support information flow between different agents through mapping rules, enable reasoning with incomplete local information, and handle (each one in its own way) agents that provide mutually inconsistent information. However, regarding their deployment in Ambient Intelligence, they have the following limitations:

- The approach of [38] assumes that all peers share a common alphabet of constants, which is not always realistic in ambient environments.
- The approaches of [38] and [40] do not include the notion of preference between system peers, which could be used to resolve potential conflicts caused by mutually inconsistent information sources.
- The method followed by [31] assumes a global preference relation on the systems peers, which is shared and used by all peers. This feature is in contrast with

the dimension of perspective, which allows each agent to use its own preference relation based on its own viewpoint.

- The distributed algorithms used in [40] and [31] assume that the inconsistencies caused by the mappings of a newly joined peer must be computed at the time the mappings are created, and not at reasoning time. This has two implications: (a) It may produce an additional possibly unnecessary computational overhead to a peer, considering that it may never have to use this information; (b) This information may become stale, in the sense that some of the mappings that cause the inconsistencies may have been defined by a peer which has left the system at the time of query evaluation.

- The studies of [40] and [31] do not deal with cases of local inconsistency, which are realistic in ambient environments.

- None of the approaches include the notion of privacy. All peers are expected to cooperate and disclose the same type of information during distributed query evaluation, and use the same strategy for conflict resolution.

2.2 Argumentation Systems

Argumentation systems constitute a way to formalize non-monotonic reasoning, viz. as the construction and comparison of arguments for and against certain conclusions. In these systems, the basic notion is not that of a defeasible conditional but that of a defeasible argument. The idea is that the construction of arguments is monotonic, i.e., an argument stays an argument if more premises are added. Non-monotonicity, or defeasibility, is not explained in terms of the interpretation of a defeasible conditional, but in terms of the interactions between conflicting arguments: in argumentation systems non-monotonicity arises from the fact that new premises may give rise to stronger counter-arguments, which defeat the original argument.

Argumentation systems can be applied to any form of reasoning with contradictory information, whether the contradictions have to do with rules and exceptions or not. For instance, the contradictions may arise from reasoning with several sources of information, or they may be caused by disagreement about beliefs or about moral, ethical or political claims. Moreover, it is important that several argumentation systems allow the construction and attack of arguments that are traditionally called *ampliative*, such as inductive, analogical and abductive arguments; these reasoning forms fall outside the scope of most other non-monotonic logics.

Most argumentation systems have been developed in artificial intelligence research on non-monotonic reasoning, although Pollock's work [93], which was the first logical formalization of defeasible argumentation, was initially applied to the philosophy of knowledge and justification (epistemology). The first artificial intelligence paper on argumentation systems was [99]. Argumentation systems have been applied to domains such as legal reasoning, medical reasoning and negotiation. Below, we present an abstract approach to defeasible argumentation, developed in several articles by Bondarenko, Dung, Toni and Kowalski. We also give a brief description of some other interesting approaches.

2.2.1 The Abstract Approach of Bondarenko, Dung, Kowalski and Toni

Historically, this work came after the development by others of a number of argumentation systems (to be discussed below). The major innovation of the BDKT approach is that it provides a framework and vocabulary for investigating the general features of these other systems, and also of non-monotonic logics that are not argument-based.

The latest and most comprehensive account of the BDKT approach is Bondarenko *et al.* [32]. In this account, the basic notion is that of a set of *assumptions*. In their approach the premises come in two kinds: *ordinary* premises, comprising a *theory*, and *assumptions*, which are formulas (of whatever form) that are designated (on whatever ground) as having default status. Bondarenko *et al.* regard non-monotonic reasoning as adding sets of assumptions to theories formulated in an underlying monotonic logic, provided that the contrary of the assumptions cannot be shown. What in their view makes the theory argumentation-theoretic is that this provision is formalized in terms of sets of assumptions attacking each other. In other words, according to [32], an argument is a set of assumptions. This approach has especially proven successful in capturing existing non-monotonic logics.

Another version of the BDKT approach, presented by Dung [45], completely abstracts from both the internal structure of an argument and the origin of the set of arguments; all that is assumed is the existence of a set of arguments, ordered by a binary relation of *defeat*. Dung then defines various notions of so-called argument extensions, which are intended to capture various types of defeasible consequence. These notions are declarative, just declaring sets of arguments as having a certain status. Finally, Dung shows that many existing non-monotonic logics can be reformulated as instances of the abstract framework.

2.2.2 Other Approaches

Pollock

Another interesting approach is Pollock's argumentation system [93]. In this system, the underlying logical language is that of standard first-order logic, but the notion of an argument has some non-standard features. What still conforms to accounts of deductive logic is that arguments are sequences of propositions linked by inference rules (or better, by instantiated inference schemes). However, Pollock's formalism begins to deviate when we look at the kinds of inference schemes that can be used to build arguments.

Inheritance Systems

A forerunner of argumentation systems is work on so-called inheritance systems, especially of Horty *et al.* [71]. Inheritance systems determine whether an object of a certain kind has a certain property. Their language is very restricted. The network is a directed acyclic graph. Its initial nodes represent individuals and its other nodes stand for classes of individuals. There are two kinds of links $\rightarrow$ and $\nrightarrow$, depending on whether an individual does or does not belong to a certain class, or a class is or is not member of a certain class. Links from an individual to a class express class membership, and links between two classes express class inclusion. A path through the graph is an inheritance path iff its only negative link is the last one. Thus the following are examples of inheritance paths:

$$P_1 : Tweety \rightarrow Penguin \rightarrow Canfly$$

$$P_1 : Tweety \rightarrow Penguin \nrightarrow Canfly$$

Another basic notion is that of an assertion, which is of the form $x \rightarrow y$ or $x \nrightarrow y$ where x is an individual and y is a class. Such an assertion is *enabled* by an inheritance path if the path starts with x and ends with the same link to y as the assertion. Above, an assertion enabled by P_1 is $Tweety \rightarrow Canfly$, and an assertion enabled by P_2 is $Tweety \nrightarrow Canfly$. Two paths can be conflicting. They are compared on specifity, which is read off from the syntactic structure of the net, resulting in relations of *neutralization* and *preemption* between paths. Although Horty *et al.* present their system as a special-purpose formalism, it clearly has all the elements of an argumentation system. An inheritance path corresponds to an argument, and an assertion enabled by a path to a conclusion of an argument. Their notion of conflicting paths corresponds to rebutting attack. Furthermore, neutralization and preemption correspond to defeat, while a permitted path is the same as a justified argument.

Lin & Shoham

Before the BDKT approach, an earlier attempt to provide a unifying framework for non-monotonic logics was made by Lin & Shoham [80]. They show how any logic, whether monotonic or not, can be reformulated as a system for constructing arguments. However, in contrast with the other theories in this section, they are not concerned with comparing incompatible arguments, and so their framework cannot be used as a theory of defeat among arguments. The basic elements of Lin & Shoham's abstract framework are an unspecified logical language, only assumed to contain a negation symbol, and an also unspecified set of inference rules defined over the assumed language. Arguments can be constructed by chaining inference rules into trees. Inference rules are either monotonic or non-monotonic.

Vreeswijk

Like the BDKT approach and Lin & Shoham [80], Vreeswijk [116; 116] also aims to provide an abstract framework for defeasible argumentation. His framework builds on the one of Lin & Shoham, but contains the main elements that are missing in their system; namely, notions of conflict and defeat between arguments. As Lin & Shoham, Vreeswijk also assumes an unspecified logical language L, only assumed to contain the symbol $\perp$, denoting *falsum* or *contradiction*, and an unspecified set of monotonic and nonmonotonic inference rules (which Vreeswijk calls *strict* and *defeasible*). This also makes his system an abstract framework rather than a particular system.

Prakken & Sartor

Inspired by legal reasoning, Prakken Sartor [95; 96] have developed an argumentation system that combines the language (but not the rest) of default logic with the grounded semantics of the BDKT approach. Actually, Prakken & Sartor originally used the language of extended logic programming, but Prakken [94] generalized the system to default logic's language. The main contributions to defeasible argumentation are a study of the relation between rebutting and assumption attack, and a formalization of argumentation about the criteria for defeat.

Governatori et al.

In [61], Governatori *et al.* provide argumentation semantics for two defeasible logics: ambiguity blocking and ambiguity propagating Defeasible Logic [10]. In both cases they disregard the rule superiority relation without, however, affecting the generality of their approach, as it has been proved that this relation can be simulated by the other ingredients of the logic [8]. Specifically, they show that Dung's grounded semantics characterizes the ambiguity propagating Defeasible Logic, while to give a semantic characterization of the original (ambiguity blocking) Defeasible Logic they modify Dung's notion of acceptability. In this framework, they define arguments as (possible infinite) proof trees, and classify them into two categories: strict and defeasible arguments. They define the notions of *attack* and *undercut* for defeasible arguments only. Specifically, a defeasible argument is attacked by arguments with conflicting conclusion, and undercut when a proper subargument is attacked by a justified argument. They also define the notions of *acceptable*, *justified* and *rejected* arguments for the two logics, taking into account the attacks against them. The argumentation framework for ambiguity blocking Defeasible Logic of [61] can be considered as a special case of the argumentation theoretic semantics of the ambiguity blocking DL with superiority relation provided in [60]. A distinct characteristic of the latter approach is that in order to handle *team defeat*, which refers to the full Defeasible Logic with priorities [10], they define arguments as sets of proof trees, instead of single proof trees.

2.2.3 Preference-based Argumentation Systems

A central notion in argument-based reasoning is that of *acceptability* of arguments. In general, to determine whether an argument is acceptable, it must be compared to its counter-arguments; namely, those arguments that support opposite or conflicting conclusions. In preference-based argumentation systems, this comparison is enabled by a preference relation, which is either implicitly derived from elements of the underlying theory, or is explicitly defined on the set of arguments.

In the works of Simari and Loui [105], and Stolzenburg *et al.* [108], the preference relation takes into account the internal structure of arguments. Overall, arguments are compared in terms of *specifity.* Intuitively, the notion of specifity favors two aspects in an argument; it prefers an argument with greater information content or less use of defeasible information. In other words, an argument will be deemed better than another if it is more *precise* or *concise.* The object language used in [105] is a first-order language, which uses facts and defeasible rules, while in the case of [108], the underlying language is *Defeasible Logic Programming* [51], which uses both strict and defeasible rules. An extension of [105], which explicitly supports distributed entities in the argumentation process, and an argumentation-based Multi-Agent Systems architecture have been recently proposed by Thimm *et al.* in [109; 111]. In this framework, all agents share common strict knowledge, which is encoded in strict rules, while each of them also possesses subjective beliefs encoded in local defeasible theories. Each agent generates its own arguments on the basis of the global knowledge and the local belief bases and reacts on arguments of other agents with counterarguments. In [110], they extend this framework to support collaborations between agents, which enables combining the belief bases of different agents and producing more and better arguments. The distributed nature of these frameworks is, however, compromised by two assumptions that they make; namely, the existence of a meta-agent representing and acting on behalf of collaborative agents, and the disjointness of collaborations - one agent cannot participate in more than one collaborations.

The approaches followed by Prakken and Sartor [96], Governatori *et al.* [60], and Kakas and Moraitis [73] share the common feature that preferences among arguments are derived from a priority relation on the rules in the underlying theory. Specifically, in [96], the underlying language is that of Extended Logic Programming, which includes strict rules and defeasible rules, while preference information is provided in the form of an ordering on the defeasible rules. They study two different cases with respect to the nature of this ordering. In the first one, the ordering is fixed and indisputable (*strict priorities*), while in the second case priorities are themselves defeasibly derived as conclusions within the system. To support defeasible priorities, they allow stating rules and constructing arguments about priorities. The argumentation framework of Governatori and Maher [60] uses the language of Defeasible Logic. In this framework, the rule priority relation of Defeasible Logic is used to determine whether an argument is defeated by a counter-argument. Similarly with [96], the framework proposed by Kakas and Moraitis [73] also includes the notion of *dynamic* priorities. Specifically, the underlying monotonic logic includes a special type of rules that are used to give priority between competing rules in case of conflict. Based on these rules, they build arguments on priorities, and reason with them to give preference to specific arguments in the system. Another interesting feature of this work is the integration of abduction to handle cases of incomplete information.

An alternative approach, which is followed by Bench-Capon [14], and by Kaci and Torre [72], relates arguments to the *values* they promote. In the so called *Value Based Argumentation Frameworks*, the preference ordering of the arguments is derived from a preference ordering over their related values. In [14], each argument promotes only a single value, and an argument is preferred to another one if and only if the value promoted by the former argument is preferred to the value promoted by the latter one. Kaci and Torre [72] provide a generalization of this approach, in which arguments can promote multiple values and the comparison is conducted between sets of arguments.

The abstract preference-based argumentation frameworks proposed by Amgoud and her colleagues [4; 5; 6] assume that preferences among arguments are induced by a preference relation defined on the supports of the arguments, which is a partial pre-ordering. The preference relation on the supports is itself induced by a priority relation defined on the underlying belief base. In [7], Amgoud *et al.* introduce the notion of *contextual* preferences in the form of several pre-orderings on the belief base, to take into account preferences that depend upon a particular context. Conflicts between preferences may appear when these preferences are expressed in different contexts; e.g. an argument A may be preferred to another argument B in a context C_1 and the argument B may be preferred to A in a context C_2. To resolve this kind of conflicts, they use meta-preferences, in the form of total preordering on the set of contexts.

The abstract argumentation framework of Modgil [86] integrates meta-level argumentation about preferences in Dung's argumentation theory. The extended framework retains the basic conceptual machinery of Dung's theory, and extends the notion of *defeat* to account for arguments that claim preferences between arguments.

Finally, building on previous works on inconsistency handling in P2P Query Answering Systems [2; 31; 40], Binas and McIlraith propose an argumentation framework that integrates preference information about the system peers [30]. They assume that each peer has its own knowledge base encoded in propositional logic, and use a preference ordering on the system peers to resolve potential conflicts that may arise between arguments derived from different peers.

The argumentation framework described in the following chapter is an extension of the framework of Governatori *et al.* [61]. Both frameworks use the language of Defeasible Logic without superiority relation as their underlying language. To take account for the distribution of context information, and a preference ordering on the system contexts, our framework additionally uses the notions of *local* and *mapping* arguments, *argumentation lines*, and *rank* of arguments. In the proposed framework, preferences are derived both from the structure of arguments - arguments that use strict rules are considered stronger than those that use defeasible rules - and from the preference ordering - arguments that use information from more preferred contexts are stronger than those that use information from less preferred contexts. Among the remaining works that we discuss in this section, our approach shares common ideas with the argumentation framework with contextual preferences of Amgoud *et al.* [7], in the sense that both frameworks allow different contexts to define different preference orderings. The main differences are that in our case, these orderings are applied to the contexts themselves rather than directly to a set of arguments, and that each context uses a different knowledge base; and therefore a different set of arguments. Compared to the P2P Argumentation System of Binas and McIlraith [30], our framework differs in two primary ways: (a) it can additionally handle local inconsistencies using defeasible arguments; (b) it assumes that each context uses its own preference ordering, opposed to the global preference relation used in [30].

Chapter 3

Contextual Argumentation

This chapter describes a non-monotonic extension of Multi-Context Systems that integrates defeasible mapping rules and preference information to model imperfect context information. The second section provides a semantic characterization of the model using arguments. Specifically, it provides the necessary definitions, explains how the proposed argumentation system is used to derive the logical consequences of a Multi-Context System through examples, and studies its formal properties.

3.1 Rule-based Representation Model

We model a Multi-Context System C as a collection of distributed context defeasible theories C_i: A context (context defeasible theory) is defined as a tuple of the form (V_i, R_i, T_i), where V_i is the vocabulary used by C_i, R_i is a finite set of rules, and T_i is a preference relation on C.

V_i is a finite set of positive and negative literals. If q_i is a literal in V_i, $\sim q_i$ denotes the complementary literal, which belongs also to V_i. If q_i is a positive literal p then $\sim q_i$ is $\neg q_i$; and if q_i is $\neg p$, then $\sim q_i$ is p. We assume that each context uses a distinct vocabulary.

R_i consists of two sets of rules: the set of local rules and the set of mapping rules. The body of a local rule is a conjunction of *local* literals (literals that are contained in V_i), while its head contains a single local literal. There are two types of local rules:

- Strict rules, of the form

$$r_i^l : a_i^1, a_i^2, ...a_i^{n-1} \rightarrow a_i^n$$

They express sound local knowledge and are interpreted in the classical sense: whenever the literals in the body of the rule ($a_i^1, a_i^2, ...a_i^{n-1}$) are strict consequences of the local theory, then so is the conclusion of the rule (a_i^n). Strict rules with empty body denote factual knowledge.

- Defeasible rules, of the form

$$r_i^d : b_i^1, b_i^2, ...b_i^{n-1} \Rightarrow b_i^n$$

They are used to express uncertainty, in the sense that a defeasible rule (r_i^d) cannot be applied to support its conclusion (b_i^n) if there is adequate contrary evidence.

Mapping rules associate literals from the local vocabulary V_i (*local literals*) with literals from the vocabularies of other contexts (*foreign literals*). The body of each such rule is a conjunction of local and foreign literals, while its head contains a single local literal. A mapping rule is modeled as a defeasible rule of the form

$$r_i^m : a_i^1, a_j^2, ...a_k^{n-1} \Rightarrow a_i^n$$

r_i^m associates local literals of C_i (e.g. a_i^1) with local literals of C_j (a_j^2), C_k (a_k^{n-1}) and possibly other contexts. a_i^n is a local literal of the theory that has defined r_i^m (C_i).

Finally, each context C_i defines a total preference ordering T_i on C to express its confidence on the knowledge it imports from other contexts. This is of the form:

$$T_i = [C_k, C_l, ..., C_n]$$

According to T_i, C_k is preferred by C_i to C_l if C_k precedes C_l in T_i. The total preference order enables resolving all potential conflicts that may arise from the interaction of contexts through their mapping rules. An alternative choice, which is closer to the needs of Ambient Intelligence applications, is partial ordering. However, this would add complexity to the reasoning tasks, and would enable resolving certain conflicts only.

We should note here that we have deliberately chosen to use the simplest version of Defeasible Logic, and disregard *facts*, *defeaters* and the *superiority relation* between rules, which are used in fuller versions of Defeasible Logic [8], to keep the discussion and technicalities simple. Besides, the results of [8] have shown that these elements can be simulated by the other ingredients of the logic.

Example 1. The representation model described above is applied as follows to the motivating scenario described in Section 1.3.1 of Chapter 1. The local knowledge of the mobile phone (denoted as C_1) is encoded in the following strict local rules:

$r^l_{11} : \rightarrow incoming\_call_1$

$r^l_{12} : \rightarrow normal\_mode_1$

Two defeasible rules encode Dr. Amber's preferences:

$r^d_{13} : incoming\_call_1, \neg lecture_1 \Rightarrow ring_1$

$r^d_{14} : silent\_mode_1 \Rightarrow \neg ring_1$

Mapping rules r^m_{15} and r^m_{16} encode the associations of the local knowledge of the mobile phone with the context knowledge of Dr. Amber's laptop (C_2), the localization service (C_3), and the classroom manager (C_4).

$r^m_{15} : classtime_2, location\_RA201_3 \Rightarrow lecture_1$

$r^m_{16} : \neg class\_activity_4 \Rightarrow \neg lecture_1$

The local context knowledge of the laptop, the localization service, and the classroom manager is encoded in rules r^l_{21}, r^l_{31} and r^l_{41} respectively. The classroom manager infers whether there is active class activity, based on the state of the projector that is installed in the classroom, and the number of people detected in the classroom by a person detection service (C_5). To import this information from C_5, C_4 uses rule r^m_{42}. Rule r^l_{51} encodes the local knowledge of the person detection service.

$r^l_{21} : \rightarrow classtime_2$

$r^l_{31} : \rightarrow location\_RA201_3$

$r^l_{41} : \rightarrow projector(off)_4$

$r^m_{42} : projector(off)_4, detected(1)_5 \Rightarrow \neg class\_activity_4$

$r^l_{51} : \rightarrow detected(1)_5$

The mobile phone is configured to give highest priority to information imported by the classroom manager and lowest priority to the information imported by the person detection service. This is described in the preference order $T_1 = [C_4, C_3, C_2, C_5]$. $\square$

3.2 Argumentation Semantics

The argumentation framework described in this section extends the argumentation semantics of Defeasible Logic presented in [61] (which in turn is based on Dung's argumentation framework [44; 45]) with the notions of distribution of the available information, and preference among system contexts.

3.2.1 Definitions

The framework uses arguments of local range, in the sense that each one contains rules of a single context only. Arguments of different contexts are interrelated in the *Support Relation* (Definition 1) through mapping rules. The Support Relation contains triples that represent proof trees for literals in the system. Each proof tree is made of rules of the context that the literal in its root is defined by. In case a proof tree contains mapping rules, for the respective triple to be contained in the Support Relation, similar triples for the foreign literals in the proof tree must have already been obtained. We should also note that, for sake of simplicity, we assume that there are no loops in the local context theories, and thus proof trees are finite. However the global knowledge base may contain loops caused by mapping rules, which associate different context theories.

Definition 1 *Let* $C = \{C_i\}$ *be a (Defeasible) MCS. The* Support Relation *of* C *(*SR_C*) is the set of all triples of the form* (C_i, PT_{p_i}, p_i)*, where* $C_i \in C$*,* $p_i \in V_i$*, and* PT_{p_i} *is the proof tree for* p_i *based on the set of local and mapping rules of* C_i*.* PT_{p_i} *is a tree with nodes labeled by literals such that the root is labeled by* p_i*, and for every node with label* q*:*

1. *If* $q \in V_i$ *and* $a_1, ..., a_n$ *label the children of* q *then*
   - *If* $\forall a_i \in \{a_1, ..., a_n\}$*:* $a_i \in V_i$ *then there is a local rule* $r_i \in C_i$ *with body* $a_1, ..., a_n$ *and head* q
   - *If* $\exists a_j \in \{a_1, ..., a_n\}$ *such that* $a_j \notin V_i$ *then there is a mapping rule* $r_i \in C_i$ *with body* $a_1, ..., a_n$ *and head* q
2. *If* $q \in V_j \neq V_i$*, then this is a leaf node of the tree and there is a triple of the form* (C_j, PT_q, q) *in* SR_C
3. *The arcs in a proof tree are labeled by the rules used to obtain them.*

Arguments are defined as triples in the Support Relation.

Definition 2 *An* argument *A for a literal* p_i *is a triple* (C_i, PT_{p_i}, p_i) *in* SR_C.

Any literal labeling a node of PT_{p_i} is called *a conclusion* of A. However, when we refer to *the conclusion* of A, we refer to the literal labeling the root of PT_{p_i} (p_i). We write $r \in A$ to denote that rule r is used in the proof tree of A.

The definition of *subarguments* (Definition 3) is based on the notion of *subtrees*.

Definition 3 *A (proper) subargument of A is every argument with a proof tree that is (proper) subtree of the proof tree of A.*

Based on the origin of the literals contained in their proof tree, arguments are classified to *local* arguments (Definition 4) and *mapping* arguments (Definition 5). Based on the type of rules that they use, local arguments are classified to *strict* local arguments and *defeasible* local arguments.

Definition 4 *A local argument of context* C_i *for a literal in* V_i *is an argument with a proof tree that contains only local literals of* C_i. *If a local argument A contains only strict rules, then A is a strict local argument; otherwise it is a defeasible local argument.*

Definition 5 *A mapping argument of context* C_i *for a literal in* V_i *is an argument with a proof tree that contains at least one foreign literal of* C_i.

Definition 6 $Args_{C_i}$ *is the set of all arguments of context* C_i. $Args_C$ *is the set of all arguments in* $C = \{C_i\}$: $Args_C = \bigcup_i Args_{C_i}$

Example 2. Consider the following context theory C_1:

$r^l_{11} : a_1 \rightarrow x_1$
$r^m_{12} : a_2 \Rightarrow a_1$
$r^m_{13} : a_3, a_4 \Rightarrow \neg a_1$
$r^d_{14} : b_1 \Rightarrow x_1$
$r^d_{15} : \Rightarrow b_1$
$r^l_{16} : d_1 \rightarrow \neg b_1$
$r^l_{17} : \rightarrow d_1$

a_1, x_1, b_1 and d_1 are local literals of C_i (they belong to V_i), while a_2, a_3 and a_4 are local literals of C_2, C_3 and C_4 respectively, which belong to the same Multi-Context

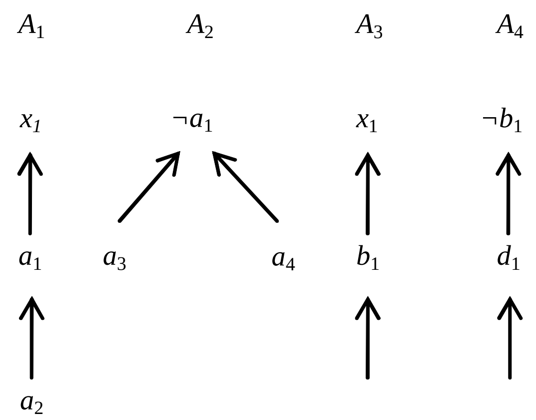

Figure 3.1: Arguments contained in $Args_{C_i}$ (example 2)

System C. Assuming that for a_2, a_3 and a_4 there are triples of the form (C_i, PT_{a_i}, a_i) in SR_C, $Args_{C_1}$ contains the arguments depicted in Figure 3.1.

The subarguments of A_1, A_3 and A_4 with conclusions a_1, b_1 and d_1 respectively are also arguments in $Args_{C_1}$. A_3 and A_4 are local arguments, since they contain only local literals of C_1. A_3 is a defeasible local argument, since it contains two defeasible rules (r^d_{14} and r^d_{15}), while A_4 is a strict local argument since both rules that it contains (r^l_{16} and r^l_{17}) are strict. On the other hand, A_1 and A_2 are mapping arguments of C_1 since they both contain foreign literals of C_1. □

The derivation of local logical consequences of a context C_i is based on its strict local arguments. Actually, the conclusions of all strict local arguments in $Args_{C_i}$ are actually logical consequences of C_i. The derivation of distributed logical consequences in C is based both on local and mapping arguments. In this case, we should also consider conflicts between competing rules (rules with complementary head literals), which are modeled in the framework as *attacks* between the arguments that contains the competing rules, and the preference orderings on the system contexts, which are used in our framework to *rank* mapping arguments.

The rank of a mapping argument (Definition 8) depends on the ranks of the literals contained in the argument (Definition 7).

Definition 7 *The rank of a literal p in context C_i ($R(p, C_i)$) equals 0 if $p \in V_i$. If $p \in V_j \neq V_i$, then $R(p, C_i)$ equals the rank of C_j in T_i (the preference order of C_i).*

Definition 8 *The rank of an argument* $A = (C_j, PT_{p_j}, p_j)$ *in* C_i *(*$R(A, C_i)$*) equals the maximum between the ranks in* C_i *of the literals contained in* PT_{p_j}*.*

It is obvious that for any three arguments A_1, A_2, A_3: If $R(A_1, C_i) \leq R(A_2, C_i)$ and $R(A_2, C_i) \leq R(A_3, C_i)$, then $R(A_1, C_i) \leq R(A_3, C_i)$; namely the preference relation $<$ on $Args_C$ is transitive. From the definitions above, it also becomes obvious that the ranks of local arguments equal 0, while the ranks of mapping arguments are greater than 0.

The definitions of *attack* and *defeat* that follow apply only for defeasible local and mapping arguments.

Definition 9 *An argument* A *attacks a defeasible local or mapping argument* B *at* p_i*, if* p_i *is a conclusion of* B*,* $\sim p_i$ *is a conclusion of* A*, and the subargument of* B *with conclusion* p_i *is not a strict local argument.*

Definition 10 *An argument* A *defeats a defeasible local or mapping argument* B *at* p_i*, if* A *attacks* B *at* p_i*, and for the subargument of* A*,* A'*, with conclusion* $\sim p_i$*, and the subargument of* B*,* B'*, with conclusion* p_i*, it holds that the rank of* B' *in* C_i *is not lower than the rank of* A' *in* C_i*:* $R(A', C_i) \leq R(B', C_i)$*.*

Example 2 (continued). Assuming that the preference ordering of C_1 is $T_1 = [C_4, C_2, C_3]$, A_1 attacks and defeats A_2 at $\neg a_1$, as $R(A'_1, C_1) = R(a_2, C_1) = 2$ (where A'_1 is the subargument of A_1 with conclusion a_1), while $R(A_2, C_1) = R(a_3, C_1) = 3$. A_2 attacks but does not defeat A_1 at a_1. For A_3 and A_4, it holds that A_4 attacks and defeats A_3 at b_1, since both are local arguments, and thus their ranks (and the ranks of their subarguments) equal 0. On the other hand, A_3 does not attack (and hence does not defeat) A_4, since A_4 is a strict local argument. $\square$

To link arguments through the mapping rules that they contain, we introduce in our framework the notion of *argumentation line*.

Definition 11 *An* argumentation line A_L *for a literal* p_i *is a sequence of arguments in* $Args_C$*, constructed in steps as follows:*

- *In the first step add in* A_L *one argument for* p_i

- *In each next step, for each distinct literal q_j labeling a leaf node of the proof trees of the arguments added in the previous step, add one argument with conclusion q_j. The addition should obey the following restriction.*

- *An argument B for a literal q_j can be added in A_L only if there is no argument $D \neq B$ for q_j already in A_L.*

The argument for p_i added in the first step is called the *head argument* of A_L, and its conclusion p_i is also the conclusion of A_L. If the number of steps required to build an A_L is finite, then A_L is a finite argumentation line. Infinite argumentation lines imply loops in the global knowledge base. Arguments contained in infinite lines participate in *attacks* against counter-arguments but may not be used to support the conclusion of their argumentation lines.

The notion of *supported* argument (Definition 12) is meant to indicate when an argument has an active role in proving or preventing the derivation of a conclusion.

Definition 12 *An argument A is supported by a set of arguments S if:*

- *every proper subargument of A is in S and*

- *there is a finite argumentation line A_L with head A, such that every argument in $A_L - \{A\}$ is in S*

That an argument A is *undercut* by a set of arguments S (Definition 13) means that we can show that some premises of A cannot be proved if we accept the arguments in S.

Definition 13 *A defeasible local or mapping argument A is undercut by a set of arguments S if for every argumentation line A_L with head A: there is an argument B, such that B is supported by S, and B defeats a proper subargument of A or an argument in $A_L - \{A\}$.*

Example 3. Consider the Multi-Context System C depicted in Figure 3.2. The arguments in Figure 3.3 and their subarguments constitute the set of arguments $Args_C$. Specifically, $Args_{C_1} = \{A_1, B_1\}$, $Args_{C_2} = \{A_2, B_2\}$, $Args_{C_3} = \{B_3\}$, $Args_{C_4} = \{B_4\}$, $Args_{C_5} = \{A_5\}$, and $Args_{C_6} = \{A_6, B_6\}$. A_1, A_2 and A_5 can be used to form an argumentation line for x_1 (A_{L1}). B_1, B_3 and B_4 are used for the construction of an argumentation line for $\neg a_1$ (B_{L1}), while B_2 and B_6 are the arguments contained in

$\underline{C_1}$	$\underline{C_2}$	$\underline{C_3}$
$r^l_{11} : a_1 \rightarrow x_1$	$r^m_{21} : a_5 \Rightarrow a_2$	$r^l_{31} :\rightarrow a_3$
$r^m_{12} : a_2 \Rightarrow a_1$	$r^m_{22} : a_6 \Rightarrow \neg a_2$	
$r^m_{13} : a_3, a_4 \Rightarrow \neg a_1$		
$\underline{C_4}$	$\underline{C_5}$	$\underline{C_6}$
$r^l_{41} :\rightarrow a_4$	$r^l_{51} :\rightarrow a_5$	$r^d_{61} :\Rightarrow a_6$
		$r^l_{62} :\rightarrow \neg a_6$

Figure 3.2: A MCS of Six Context Theories (example 3)

the argumentation line for $\neg a_2$ (B_{L2}). Assuming that $S = \{A_5, A_6\}$, the following statements hold:

- A_2 is supported by S since there is an argumentation line A_{L2} with head A_2 (A_{L2} is the line obtained by removing A_1 from A_{L1}), A_2 has no proper subarguments, and A_5, which is the only argument contained in $A_{L2} - \{A_2\}$ is in S.

- B_2 is undercut by S since it is a mapping argument, and A_6, which is in S defeats B_6, which is contained in the only argumentation line with head B_2 (B_{L2}).

Assuming that $S = \{A_5, A_6, B_3, B_4, A_2\}$, $T_1 = [C_3, C_2, C_4, C_5, C_6]$ and $T_2 = [C_6, C_5, C_1, C_3, C_4]$ the following statements hold:

- B_1 and A'_1 (the subargument of A_1 with conclusion a_1) are supported by S.

- A'_1 is not undercut by S, since it has no proper subarguments and none of the arguments in A_{L1} is defeated by an argument supported by S (B_1 attacks but does not defeat A_1 $R(A_1, C_1) = 2 < R(B_1, C_1) = 3$, while B_2 defeats A_2, since $R(B_2, C_2) = 1 < R(A_2, C_2) = 2$, but B_2 is not supported by S). □

The definition of *acceptable* arguments that follows is based on the definitions given above. Intuitively, that an argument A is *acceptable* w.r.t. a set of arguments S means that if we accept the arguments in S as valid arguments, then we feel compelled to accept A as valid.

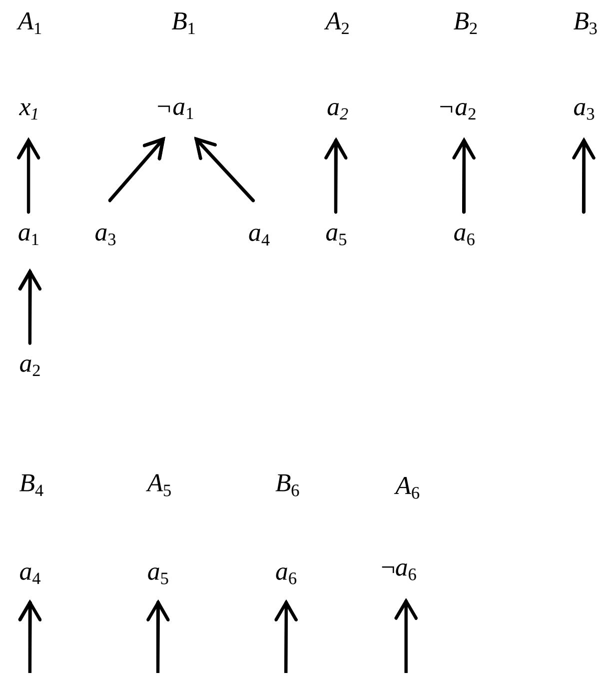

Figure 3.3: Arguments contained in $Args_C$ (example 3)

Definition 14 *An argument A is acceptable w.r.t a set of arguments S if:*

1. *A is a strict local argument; or*

2. (a) *A is supported by S and*

   (b) *every argument in $Args_C$ defeating A is undercut by S*

Based on the concept of acceptable arguments, we proceed to define justified arguments and justified literals.

Definition 15 *Let C be a MCS. J_i^C is defined as follows:*

- $J_0^C = \emptyset$;

- $J_{i+1}^C = \{A \in Args_C \mid A \text{ is acceptable w.r.t. } J_i^C\}$

The set of *justified arguments* in a MCS C is $JArgs^C = \bigcup_{i=1}^{\infty} J_i^C$. A literal p_i is *justified* in C if it is the conclusion of an argument in $JArgs^C$. That an argument A is justified means that it resists every reasonable refutation. That a literal p_i is justified, it actually means that it is a logical consequence of C.

Finally, we also introduce the notion of *rejected arguments* and *rejected literals* for the characterization of conclusions that do not derive from C. That an argument is rejected by sets of arguments S and T means that either it is supported by arguments in S, which can be thought of as the set of already rejected arguments, or it cannot overcome an attack from an argument supported by T, which can be thought of as the set of justified arguments.

Definition 16 *An argument A is rejected by sets of arguments S, T when:*

1. *A is not a strict local argument, and either*
2. *(a) a proper subargument of A is in S; or*
   (b) A is defeated by an argument supported by T; or
   (c) for every argumentation line A_L with head A there exists an argument $A' \in A_L - A$, such that either a subargument of A' is in S; or A' is defeated by an argument supported by T

Based on the definition of rejected arguments, R_i^C is defined as follows:

Definition 17 *Let C be a MCS, and $JArgs^C$ the set of justified arguments in C. J_i^C is defined as follows:*

- $R_0^C = \emptyset$;
- $R_{i+1}^C = \{A \in Args_C \mid A \text{ is rejected by } R_i^C,\ JArgs^C\}$

The set of *rejected arguments* in a MCS C is $RArgs^C = \bigcup_{i=1}^{\infty} R_i^C$. A literal p_i is *rejected* in C if there is no argument in $Args^C - RArgs^C$ with conclusion p_i. That a literal is rejected means that we are able to prove that it is not a logical consequence of the system theories.

Example 3 (continued). Based on the definitions given above, we can calculate the following for the arguments depicted in Figure 3.3.

$J_0^C = \{\}$;
$J_1^C = \{B_3, B_4, A_5, A_6\}$;
$J_2^C = \{B_3, B_4, A_5, A_6, A_2\}$;
$J_3^C = \{B_3, B_4, A_5, A_6, A_2, A_1'\}$;

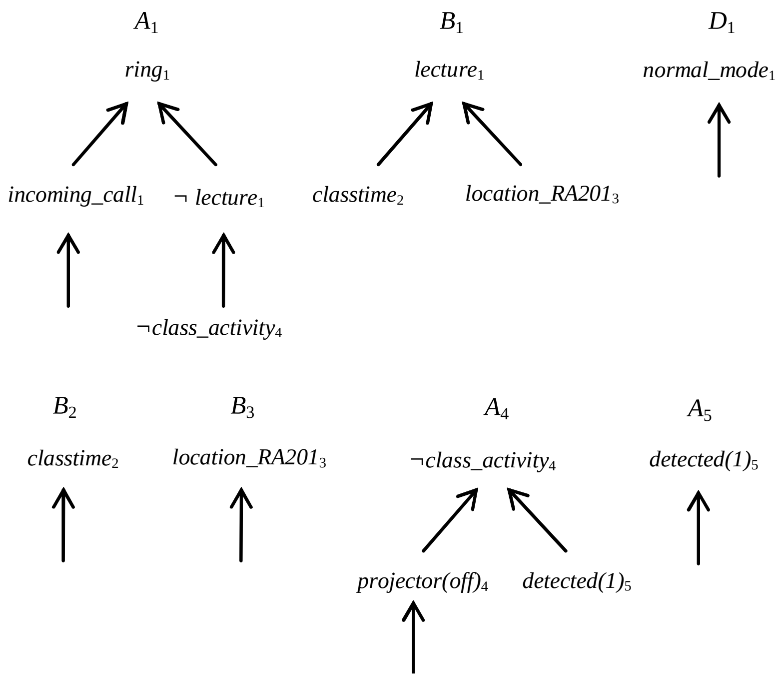

Figure 3.4: Arguments in the Ambient Intelligence Scenario (example 1)

$J_4^C = \{B_3, B_4, A_5, A_6, A_2, A'_1, A_1\} = JArgs^C$

$R_0^C(JArgs^C) = \{\}$;

$R_1^C(JArgs^C) = \{B_6, B_2, B_1\} = RArgs^C(JArgs^C)$

Based on the above, a_1, x_1, a_2, a_3, a_4, a_5 and $\neg a_6$ are justified literals, while $\neg a_1$, $\neg a_2$ and a_6 are rejected literals. □

Example 1 (continued). The arguments that are derived from the MCS C of Example 1 are the arguments depicted in Figure 3.4 and their subarguments.

A_1, B_1 and D_1 are in $Args_{C_1}$; the first two of them are mapping arguments, while D_1 is a strict local argument. B_2, B_3 and A_5 are strict local arguments of C_2, C_3 and C_5, respectively, while A_4 is a mapping argument of C_4.

J_0^C contains no arguments, while J_1^C contains the strict local arguments of the system; namely A'_1, D_1, B_2, B_3, A'_4 and A_5, where A'_1 is the subargument of A_1 with conclusion $incoming\_call_1$, and A'_4 is the subargument of A_4 with conclusion $projector(off)_4$.

J_2^C additionally contains A_4, since A_4 and A_5 form an argumentation line (A_{L4}) with head A_4, both A_5 and A'_4 (the only proper subargument of A_4) are in J_1^C, and there is no argument that attacks A_4.

J_3^C additionally contains A''_1, which is the subargument of A_1 with conclusion $\neg lecture_1$, since A_5, A_4 and A''_1 form an argumentation line with head A''_1, A_4 and A_5 are both in J_2^C, and the only argument attacking A''_1, B_1, does not defeat A''_1 as $R(A''_1, C_1) = 1$, while $R(B_1, C_1) = 3$ (according to $T_1 = [C_4, C_3, C_2, C_5]$).

Finally, J_4^C additionally contains A_1, as all proper subarguments of A_1 (A'_1 and A''_1) are in J_3^C, there is an argumentation line with head A_1, which consists of arguments A_1, A_4 and A_5, and both A_4 and A_5 are in J_3^C, while the only attacking argument, B_1, does not defeat A_1.

J_4^C actually constitutes the set of justified arguments in C ($JArgs^C = J_4^C$), as there is no other argument that can be added in the next steps of J_i^C. Hence, all the literals defined in the system except $lecture_1$ and $silent\_mode_1$ are justified.

On the other hand, $R_0^C(JArgs^C)$ contains no arguments, while $R_1^C(JArgs^C)$, which equals $RArgs^C(JArgs^C)$ contains only one argument, B_1. Hence, $lecture_1$ is a rejected literal, since the only argument with conclusion $lecture_1$ (B_1) is in $RArgs^C(JArgs^C)$. $silent\_mode_1$ is also a rejected literal, since there is no argument is $Args_C$ with this conclusion. □

Example 4 - Infinite Argumentation Lines. As we already stated in this Chapter, infinite argumentation lines cannot be used for the justification of a conclusion. However, unlike other approaches, we do not consider them as a fallacy of the system, as they can participate in attacks against counter-arguments. An infinite argumentation line is created when the conclusion of its head argument is also a conclusion of another argument in this line.

Consider a MCS of four contexts, which use the arguments depicted in Figure 3.5. C_1 uses A_1 and B_1, C_2 uses A_2, C_3 uses A_3, and C_4 uses B_4. A_1, B_1, A_2 and A_3 are mapping arguments, while B_4 is a local argument of C_4.

In this system, A_1 is the head argument of an infinite argumentation line $A_{L1} = \{A_1, A_2, A_3, A_1, A_2, ...\}$. On the other hand, B_1 is the head argument of the finite argumentation line B_{L1}=$\{B_1, B_4\}$. Assuming that the preference ordering of C_1 is $T_1 = [C_2, C_3, C_4]$, a_1 is not justified, as A_1 is the only argument with conclusion a_1, and

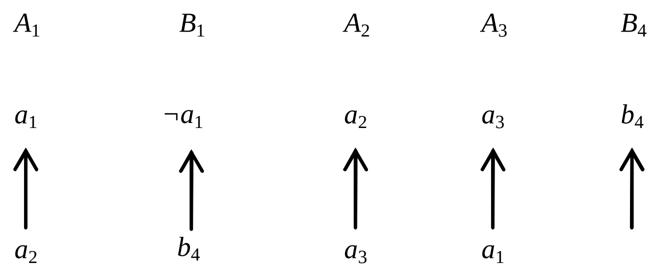

Figure 3.5: Infinite Argumentation Lines (example 4)

there is no finite argumentation line with head A_1. a_1 is not rejected, as B_1, which is the only argument that attacks A_1, has higher rank than A_1, and therefore does not defeat A_1. On the other hand, $\neg a_1$ is not justified, because A_1 defeats B_1 and is not undercut by $JArgs^C$, which contains only one argument, B_4. $\neg a_1$ is not also rejected because although A_1 defeats B_1, it is only supported by A_{L1}, which is an infinite argumentation line, and therefore it cannot be supported by $JArgs^C$.

Assuming that $T_1 = [C_4, C_2, C_3]$, B_1 defeats A_1, and therefore using B_{L1} we can prove that B_1 is in $JArgs^C$, and hence $\neg a_1$ is justified. On the other hand, a_1 is rejected in C, since B_1 is supported by $JArgs^C$ and defeats A_1. □

Example 5 - Self-Defeating Argumentation Lines. In this example we show how our framework deals with the so-called self-defeating argumentation lines. Consider a MCS of four contexts, which use the arguments depicted in Figure 3.6. C_1 uses A_1 and B_1, C_2 uses B_2, C_3 uses B_3, and C_4 uses A_4. A_1, B_1, B_2 and B_3 are mapping arguments, while A_4 is a strict local argument of C_4.

In this system, A_1 is the head argument of argumentation line A_{L1}=$\{A_1, A_4\}$, while B_1 is the head argument of argumentation line $B_{L1} = \{B_1, B_3, B_2, A_1, A_4\}$. Assuming that the preference ordering of C_1 is $T_1 = [C_2, C_3, C_4]$, a_1 is neither accepted nor rejected. It is not justified, since A_1 is defeated by B_1 (B_1 has lower rank than A_1), and although B_{L1} is undercut at argument A_1 by B_1, B_1 is not supported by $JArgs^C$, which contains only A_4, and therefore B_{L1} is not undercut by $JArgs^C$. a_1 is not rejected since, although B_1 defeats A_1, it is not supported by $JArgs^C$. On the other hand, $\neg a_1$ is also neither justified nor rejected. It is not justified, because A_1, which is one of the arguments in B_{L1}, is defeated by B_1, and B_1 is not undercut by $JArgs^C$, since none of

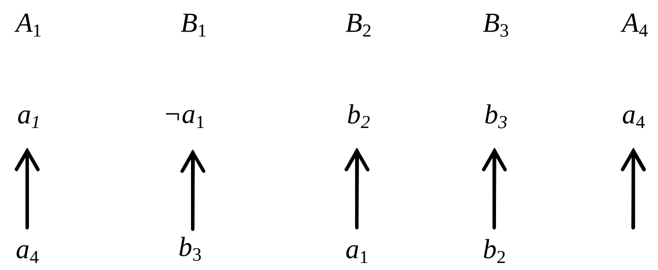

Figure 3.6: Self-Defeating Argumentation Lines (example 5)

the arguments in B_{L1} is defeated by an argument that is supported by $JArgs^C$. $\neg a_1$ is not rejected because none of the arguments in B_{L1} is defeated by an argument that is supported by $JArgs^C$.

Assuming that $T_1 = [C_4, C_2, C_3]$, A_1 is not defeated by B_1, and is supported by $JArgs^C$ (since A_4 is in $JArgs^C$); hence A_1 is in $JArgs^C$, and its conclusion, a_1 is a justified conclusion. On the other hand, since B_1 is defeated by A_1, which is supported by $JArgs^C$, B_1 is rejected by $JArgs^C$, and $\neg a_1$ is rejected. □

3.2.2 Properties of the Framework

Lemmata 1-3 (presented below) describe some formal properties of the framework. Their proofs are presented in Appendix A. Lemma 1 refers to the monotonicity in J_i^C and $R_i^C(T)$.

Lemma 1 *The sequences of sets of arguments J_i^C and $R_i^C(T)$ are monotonically increasing.*

Lemma 2 represents the fact that no argument is both "believed" and "disbelieved".

Lemma 2 *In a defeasible Multi-Context System C:*

- *No argument is both justified and rejected.*
- *No literal is both justified and rejected.*

If consistency is assumed in the local strict rules of a context theory (two complementary conclusions may not derive as strict local consequences of a context theory), then using the previous Lemma, it is easy to prove that the entire framework is consistent; this is described in the following Lemma.

Lemma 3 *If the set of justified arguments in C, $JArgs^C$, contains two arguments with conflicting conclusions, then both arguments are strict local arguments.*

Chapter 4

Distributed Query Evaluation

The previous chapter presented a semantic characterization of our approach for defeasible reasoning in Multi-Context Systems using arguments. This chapter provides an operational model in the form of a distributed algorithm for query evaluation. The first section describes the algorithm in detail and explains how it works through examples, while the second one studies its formal properties regarding its termination, soundness and completeness with respect to the argumentation framework, communication and computational load. The last section describes a standard procedure that unifies the distributed context theories in a single theory of Defeasible Logic, which produces the same results with the query evaluation algorithm, under the proof theory of [8].

4.1 Algorithm Description

$P2P\_DR$ is a distributed algorithm for query evaluation in Multi-Context Systems following the model described in Section 3.1. The specific reasoning problem that it deals with is: *Given a MCS C, and a query about literal p_i issued to context C_i, compute the truth value of p_i.* For an arbitrary literal p_i, $P2P\_DR$ returns one of the following values:

- *true*; indicating that p_i is justified in C
- *false*; indicating that p_i is rejected in C
- *undefined*; indicating p_i is neither justified nor rejected in C

P2P_DR proceeds in four main steps. In the first step (lines 1-8 in the pseudocode given below), *P2P_DR* determines whether p_i, or its negation $\sim p_i$ are consequences of the local strict rules of C_i, using *local_alg*, a local reasoning algorithm, which is described later in this section. If *local_alg* computes *true* as an answer for p_i or $\sim p_i$, *P2P_DR* returns *true*/*false* respectively as an answer for p_i and terminates.

In the second step (lines 9-12), *P2P_DR* calls *Support* (described later in this section) to determine whether there are *applicable* and *unblocked* rules with head p_i. We call *applicable* those rules that for all literals in their body *P2P_DR* has computed *true* as their truth value, while *unblocked* are the rules that for all literals in their body *P2P_DR* has computed either *true* or *undefined* as their truth value. *Support* also returns two data structures for p_i: (a) the set of foreign literals used in the most preferred (according to T_i) chain of applicable rules for p_i (SS_{p_i}); and (b) the set of foreign literals used in the most preferred chain of unblocked rules for p_i (BS_{p_i}). If there is no unblocked rule for p_i, the algorithm returns *false* as an answer and terminates.

In the third step (lines 13-14), *P2P_DR* calls *Support* to compute the respective constructs for $\sim p_i$ ($SS_{\sim p_i}$, $BS_{\sim p_i}$).

In the last step (lines 15-24), *P2P_DR* uses the constructs computed in the previous steps and the preference order defined by C_i (T_i), to determine the truth value of p_i. In case there is no unblocked rule for $\sim p_i$ ($unb_{p_i} = false$), or SS_{p_i} is computed by *Stronger* (described later in this section) to be *stronger* than $BS_{\sim p_i}$, *P2P_DR* returns *true* as an answer for p_i. That SS_{p_i} is *stronger* than $BS_{\sim p_i}$ means that the chains of applicable rules for p_i involve information from contexts that are preferred by C_i to the contexts that are involved in the chain of unblocked rules for $\sim p_i$. In case there is at least one applicable rule for $\sim p_i$, and BS_{p_i} is *not stronger* than $SS_{\sim p_i}$, *P2P_DR* returns *false* as an answer for p_i. In any other case, the algorithm returns *undefined*.

The context that is called to evaluate the query for p_i (C_i) returns through Ans_{p_i} the truth value of the literal it is queried about. SS_{p_i} and BS_{p_i} are returned to the querying context (C_0) only if the two contexts (the querying and the queried one) are actually the same context. Otherwise, the empty set is assigned to both SS_{p_i} and BS_{p_i} and returned to C_0. In this way, the size of the messages exchanged between different contexts is kept small. $Hist_{p_i}$ is a structure used by *Support* to detect loops in the global knowledge base. The input parameters of *P2P_DR* are:

- p_i: the queried literal
- C_0: the context that issues the query
- C_i: the context that defines p_i
- $Hist_{p_i}$: the list of pending queries $([p_1, ..., p_i])$
- T_i: the preference ordering of C_i

The output parameters of the algorithm are:

- SS_{p_i}: a set of foreign literals of C_i denoting the Supportive Set of p_i
- BS_{p_i}: a set of foreign literals of C_i denoting the Blocking Set of p_i
- Ans_{p_i}: the answer returned for p_i

Below, we provide the pseudocode of $P2P\_DR$.

P2P_DR$(p_i, C_0, C_i, Hist_{p_i}, T_i, SS_{p_i}, BS_{p_i}, Ans_{p_i})$

call $local\_alg(p_i, localAns_{p_i})$
if $localAns_{p_i} = true$ **then**
    $Ans_{p_i} \leftarrow true$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
    terminate
call $local\_alg(\sim p_i, localAns_{\sim p_i})$
if $localAns_{\sim p_i} = true$ **then**
    $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
    terminate
call $Support(p_i, Hist_{p_i}, T_i, sup_{p_i}, unb_{p_i}, SS_{p_i}, BS_{p_i})$
if $unb_{p_i} = false$ **then**
    $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
    terminate
$Hist_{\sim p_i} \leftarrow (Hist_{p_i} - \{p_i\}) \cup \{\sim p_i\}$
call $Support(\sim p_i, Hist_{\sim p_i}, T_i, sup_{\sim p_i}, unb_{\sim p_i}, SS_{\sim p_i}, BS_{\sim p_i})$
if $sup_{p_i} = true$ and $(unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i})$ **then**
    $Ans_{p_i} \leftarrow true$
    if $C_0 \neq C_i$ **then**
        $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
else if $sup_{\sim p_i} = true$ and $Stronger(BS_{p_i}, SS_{\sim p_i}, T_i) \neq BS_{p_i}$ **then**
    $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
else

$Ans_{p_i} \leftarrow undefined$
if $C_0 \neq C_i$ **then**
$SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$

local_alg is called by *P2P_DR* to determine whether the truth value of the queried literal can be derived from the local strict rules of a context theory (R^s). We should note again that, for sake of simplicity, we assume that there are no loops in the local context theories. *local_alg* returns either *true* or *false* as a local answer for the queried literal. The algorithm parameters are:

p_i: the queried literal (input)

$localAns_{p_i}$: the local answer for p_i (output)

local_alg$(p_i, localAns_{p_i})$
for all $r_i \in R^s[p_i]$ **do**
for all $b_i \in body(r_i)$ **do**
call $local\_alg(b_i, localAns_{b_i})$
if for all b_i: $localAns_{b_i} = true$ **then**
return $localAns_{p_i} = true$ and terminate
return $localAns_{p_i} = false$

Support is called by *P2P_DR* to determine whether there are applicable and unblocked rules for p_i. In case there is at least one applicable rule for p_i, *Support* returns $sup_{p_i} = true$; otherwise, it returns $sup_{p_i} = false$. Similarly, $unb_{p_i} = true$ is returned when there is at least one unblocked rule for p_i; otherwise, $unb_{p_i} = false$.

Support also returns two data structures for p_i:

- SS_{p_i}, the Supportive Set for p_i. This is a set of literals representing the most preferred (according to T_i) chain of applicable rules for p_i
- BS_{p_i}; the Blocking Set for p_i. This is a set of literals representing the most preferred (according to T_i) chain of unblocked rules for p_i.

To compute these structures, *Support* checks the applicability of the rules with head p_i, using the truth values of the literals in their body, as these are evaluated by *P2P_DR*. To avoid loops, before calling *P2P_DR*, it checks if the same query has been issued before during the running call of *P2P_DR*. In this case, it marks the rule with a *cycle* value, and proceeds with the remaining body literals. For each applicable rule r_i, *Support* builds its Supportive Set, SS_{r_i}; this is the union of the set of *foreign literals* contained in the body of r_i with the Supportive Sets of the local literals contained in the body of the rule. Similarly, for each unblocked rule r_i, it computes its Blocking Set BS_{r_i} using the Blocking Sets of its body literals. *Support* computes the Supportive Set of p_i, SS_{p_i}, as the *strongest* rule Supportive Set SS_{r_i}; and its Blocking Set, BS_{p_i}, as the *strongest* rule Blocking Set BS_{r_i}, using the *Stronger* function. The input parameters of *Support* are:

- p_i: the queried literal
- $Hist_{p_i}$: the list of pending queries ($[p_1, ..., p_i]$)
- T_i: the preference ordering of C_i

The output parameters of *Support* are:

- sup_{p_i}, which indicates whether p_i is supported in C
- unb_{p_i}, which indicates whether p_i is unblocked in C
- SS_{p_i}: a set of foreign literals of C_i denoting the Supportive Set of p_i
- BS_{p_i}: a set of foreign literals of C_i denoting the Blocking Set of p_i

Support$(p_i, Hist_{p_i}, T_i, sup_{p_i}, unb_{p_i}, SS_{p_i}, BS_{p_i})$

$sup_{p_i} \leftarrow false$
$unb_{p_i} \leftarrow false$
for all $r_i \in R[p_i]$ **do**
    $cycle(r_i) \leftarrow false$
    $SS_{r_i} \leftarrow \emptyset$
    $BS_{r_i} \leftarrow \emptyset$
    for all $b_t \in body(r_i)$ **do**
        if $b_t \in Hist_{p_i}$ **then**
            $cycle(r_i) \leftarrow true$
            $BS_{r_i} \leftarrow BS_{r_i} \cup \{d_t\}$ {$d_t \equiv b_t$ if $b_t \notin V_i$; otherwise d_t is the first foreign literal of C_i added in $Hist_{p_i}$ after b_t}
        else
            $Hist_{b_t} \leftarrow Hist_{p_i} \cup \{b_t\}$
            call $P2P\_DR(b_t, C_i, C_t, Hist_{b_t}, T_t, SS_{b_t}, BS_{b_t}, Ans_{b_t})$

        if $Ans_{b_t} = false$ **then**
            stop and check the next rule
        else if $Ans_{b_t} = undefined$ or $cycle(r_i) = true$ **then**
            $cycle(r_i) \leftarrow true$
            if $b_t \notin V_i$ **then**
                $BS_{r_i} \leftarrow BS_{r_i} \cup \{b_t\}$
            else
                $BS_{r_i} \leftarrow BS_{r_i} \cup BS_{b_t}$
        else
            if $b_t \notin V_i$ **then**
                $BS_{r_i} \leftarrow BS_{r_i} \cup \{b_t\}$
                $SS_{r_i} \leftarrow SS_{r_i} \cup \{b_t\}$
            else
                $BS_{r_i} \leftarrow BS_{r_i} \cup BS_{b_t}$
                $SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t}$
    if $unb_{p_i} = false$ or $Stronger(BS_{r_i}, BS_{p_i}, T_i) = BS_{r_i}$ **then**
        $BS_{p_i} \leftarrow BS_{r_i}$
    $unb_{p_i} \leftarrow true$
    if $cycle(r_i) = false$ **then**
        if $sup_{p_i} = false$ or $Stronger(SS_{r_i}, SS_{p_i}, T_i) = SS_{r_i}$ **then**
            $SS_{p_i} \leftarrow SS_{r_i}$
        $sup_{p_i} \leftarrow true$

The $Stronger(A, B, T_i)$ function computes the *strongest* between two sets of literals, A and B according to the preference order T_i. A literal a_k is *preferred* to a literal b_l, if C_k has lower rank than C_l in T_i. The strength of a set is determined by the the *weakest* (least preferred) literal in this set.

Stronger(A, B, T_i)

if $\exists b_l \in B$: $\forall a_k \in A$: C_k has lower rank than C_l in T_i
or ($A = \emptyset$ and $B \neq \emptyset$) **then**
    $Stronger = A$
else if $\exists a_k \in A$: $\forall b_l \in B$: C_l has lower rank than C_k in T_i
or ($B = \emptyset$ and $A \neq \emptyset$) **then**
    $Stronger = B$
else
    $Stronger = None$

Example 3 (continued) In the MCS depicted in Figure 3.2, given a query about x_1 issued to context C_1, *P2P_DR* is called by C_1 and proceeds as follows:

- In the first step, it fails to compute the truth value of x_1 using the strict local rules of C_1.
- In the second step, *P2P_DR* calls *Support* for x_1. The only rule with head x_1 is r^l_{11}; hence *Support* calls *P2P_DR* to compute the truth value of the only literal in the body of r^l_{11}, a_1.
- *P2P_DR* fails to compute the truth value of a_1 using the strict local rules of C_1. Therefore, it calls *Support* for a_1. a_1 is only supported by rule r^m_{12}, hence *Support* calls *P2P_DR* to compute the truth value of the only literal in the body of r^m_{12}, a_2.
- *P2P_DR* fails to compute the truth value of a_2 using the strict local rules of C_2. Therefore, it calls *Support* for a_2. a_2 is only supported by rule r^m_{21}, hence *Support* calls *P2P_DR* to compute the truth value of the only literal in the body of r^m_{21}, a_5.
- Using rule r^l_{51} (a strict rule with empty body), $local\_alg$ computes $localAns_{a_5} = true$, and *P2P_DR* returns $Ans_{a_5} = true$.
- *Support* computes $SS_{r^m_{21}} = \{a_5\}$, and since there is no other rule with head a_2, $BS_{a_2} = SS_{a_2} = \{a_5\}$.
- In the next step, *P2P_DR* calls *Support* for $\neg a_2$. $\neg a_2$ is only supported by rule r^m_{22}, hence *Support* calls *P2P_DR* to compute the truth value of the only literal in the body of r^m_{22}, a_6.
- Using rule r^l_{62} (a strict rule with empty body), $local\_alg$ computes $localAns_{\neg a_6} = true$, and *P2P_DR* returns $Ans_{a_6} = false$.
- As there is no other rule with head $\neg a_2$, *Support* computes $unb_{\neg a_2} = false$, and $Ans_{a_2} = true$.
- *Support* computes $SS_{r^m_{12}} = \{a_2\}$, and since there is no other rule with head a_1, $BS_{a_1} = SS_{a_1} = \{a_2\}$.

- In the next step, *P2P_DR* calls *Support* for $\neg a_1$. $\neg a_1$ is only supported by rule r^m_{13}, hence *Support* calls *P2P_DR* to compute the truth values of a_3 and a_4.
- Both a_3 and a_4 are derived as conclusions of the strict local rules in C_3 and C_4 respectively; hence for both literals *P2P_DR* returns *true* as an answer.
- *Support* computes $SS_{r^m_{13}} = \{a_3, a_4\}$, and since there is no other rule with head $\neg a_1$, $BS_{\neg a_1} = SS_{\neg a_1} = \{a_3, a_4\}$.
- Assuming that $T_1 = [C_3, C_2, C_4, C_5, C_6]$, *P2P_DR* determines that $SS_{a_1} = \{a_2\}$ is stronger than $BS_{\neg a_1} = \{a_3, a_4\}$, and computes $Ans_{a_1} = true$.
- As there is no rule with head $\neg x_1$, *P2P_DR* computes $Ans_{x_1} = true$. $\square$

Example 1 (continued) In the MCS of example 1, given a query about $ring_1$, *P2P_DR* proceeds as follows:

- In the first step, *P2P_DR* fails to compute an answer for $ring_1$ based on the local strict rules of C_1.
- In the second step, it calls *Support* for $ring_1$. r^d_{13} is the only rule with head $ring_1$. *Support* calls *P2P_DR* for $incoming\_call_1$ and $\neg lecture_1$, which are the only literals in the body of r^d_{13}.
- Using rule r^l_{11}, $local\_alg$ computes *true* as a local answer for $incoming\_call_1$ and therefore *P2P_DR* returns $Ans_{incoming\_call_1} = true$.
- *P2P_DR* fails to compute an answer for $\neg lecture_1$ based on the local strict rules of C_1, so it calls *Support* for $\neg lecture_1$. r^m_{16} is the only rule with head $\neg lecture_1$. *Support* calls *P2P_DR* for $\neg class\_activity_4$, which is the only literal in the body of r^m_{15}.
- *P2P_DR* fails to compute an answer for $\neg class\_activity_4$ based on the local strict rules of C_4, so it calls *Support* for $\neg class\_activity_4$. r^m_{42} is the only rule with head $\neg class\_activity_4$. *Support* calls *P2P_DR* for $projector(off)_4$ and $detected(1)_5$, which are the only literals in the body of r^m_{42}.
- Using rule r^l_{41}, $local\_alg$ computes *true* as a local answer for $projector(off)_4$ and therefore *P2P_DR* returns $Ans_{projector(off)_4} = true$.

- Using rule r^l_{51}, *local_alg* computes *true* as a local answer for $detected(1)_5$ and therefore *P2P_DR* returns $Ans_{detected(1)_5} = true$.

- *Support* computes $SS_{r^m_{42}} = \{detected(1)_5\}$, and since there is no rule with head $class\_activity_4$, *P2P_DR* returns $Ans_{\neg class\_activity_4} = true$.

- *Support* computes $SS_{r^m_{16}} = \{\neg class\_activity_4\}$, and since there is no other rule with head $\neg lecture_1$, $BS_{\neg lecture_1} = SS_{\neg lecture_1} = \{\neg class\_activity_4\}$.

- In the next step, *P2P_DR* calls *Support* for $lecture_1$. $lecture_1$ is in the head of rule r^m_{15}, hence *Support* calls *P2P_DR* to compute the truth values of $classtime_2$ and $location\_RA201_3$, which are contained in the body of r^m_{15}.

- Both $classtime_2$ and $location\_RA201_3$ are derived as conclusions of the strict local rules in C_2 and C_3 respectively; hence for both literals *P2P_DR* returns *true* as an answer.

- *Support* computes $SS_{r^m_{15}} = \{classtime_2, location\_RA201_3\}$, and since there is no other rule with head $lecture_1$, $BS_{lecture_1} = \{classtime_2, location\_RA201_3\}$.

- Assuming that $T_1 = [C_4, C_3, C_2, C_5]$, *P2P_DR* determines that $SS_{\neg lecture_1}$ is stronger than $BS_{lecture_1}$, and returns $Ans_{\neg lecture_1} = true$.

- *Support* computes $SS_{r^d_{13}} = \{\neg class\_activity_4\}$, and since there is no other rule with head $lecture_1$, it returns $BS_{ring_1} = SS_{ring_1} = \{\neg class\_activity_4\}$.

- In the next step, *P2P_DR* calls *Support* for $\neg ring_1$. $\neg ring_1$ is in the head of rule r^d_{14}, hence *Support* calls *P2P_DR* to compute the truth value of $silent\_mode_1$, which is the only literal in the body of r^d_{13}. Since there is no rule with head $silent\_mode_1$, *P2P_DR* returns $Ans_{silent\_mode_1} = false$.

- As there is no other rule with $\neg ring_1$ in its head, *Support* returns $unb_{\neg ring_1} = false$, and eventually *P2P_DR* returns $Ans_{ring_1} = true$. □

Example 4 - Infinite Argumentation Lines (continued). In the MCS using the arguments depicted in Figure 3.5, *Support* will return the following results for a_1 and $\neg a_1$. For a_1, it returns: $unb_{a_1} = true$, $sup_{a_1} = false$ and $BS_{a_1} = \{a_2\}$. For $\neg a_1$, it returns $unb_{\neg a_1} = true$, $sup_{\neg a_1} = true$, and $BS_{\neg a_1} = SS_{\neg a_1} = \{b_4\}$. Therefore, in case

$T_1 = [C_2, C_3, C_4]$, $P2P\_DR$ will return $Ans_{a_1} = undefined$ and $Ans_{\neg a_1} = undefined$. In case $T_1 = [C_4, C_2, C_3]$ it will return $Ans_{a_1} = false$ and $Ans_{\neg a_1} = true$. □

Example 5 - Self-defeating Argumentation Lines (continued). In the MCS using the arguments depicted in Figure 3.6, *Support* will return the following results for a_1 and $\neg a_1$. For a_1, it returns: $unb_{a_1} = true$, $sup_{a_1} = true$ and $SS_{a_1} = BS_{a_1} = \{a_4\}$. For $\neg a_1$, it returns $unb_{\neg a_1} = true$, $sup_{\neg a_1} = false$, and $BS_{\neg a_1} = \{a_3\}$. Therefore, in case $T_1 = [C_2, C_3, C_4]$, $P2P\_DR$ will return $Ans_{a_1} = undefined$ and $Ans_{\neg a_1} = undefined$. In case $T_1 = [C_4, C_2, C_3]$ it will return $Ans_{a_1} = true$ and $Ans_{\neg a_1} = false$ □.

4.2 Properties of the Algorithm

Below, we describe some formal properties of $P2P\_DR$ regarding its termination, complexity, and soundness and completeness with respect to the argumentation framework, and propose some optimizations that aim at the reduction of the total number of messages. The proofs for the propositions that appear in this section are provided in Appendix A.

4.2.1 Termination

Proposition 1 refers to the termination of $P2P\_DR$, and is a consequence of the cycle detection process within the algorithm.

Proposition 1 *$P2P\_DR$ is guaranteed to terminate returning one of the values true, false and undefined as an answer for the queried literal.*

4.2.2 Soundness & Completeness

Propositions 2 and 3 associate the results computed by $local\_alg$ and $P2P\_DR$ with concepts of the argumentation framework. Specifically, Proposition 2 provides a characterization of answers returned by $local\_alg$ with respect to the concept of strict local arguments, while Proposition 3 associates the answers returned by $P2P\_DR$ with the concepts of justified and rejected literals.

Proposition 2 *For a Multi-Context System C and a literal p_i in $C_i \in C$, $local\_alg$ returns*

1. $localAns_{p_i} = true$ *iff there is a strict local argument for* p_i *in* $Args_{C_i}$

2. $localAns_{p_i} = false$ *iff there is no strict local argument for* p_i *in* $Args_{C_i}$

Proposition 3 *For a Multi-Context System* C *and a literal* p_i *in* C*, P2P_DR returns:*

1. $Ans_{p_i} = true$ *iff* p_i *is justified in* C

2. $Ans_{p_i} = false$ *iff* p_i *is rejected in* C

3. $Ans_{p_i} = undefined$ *iff* p_i *is neither justified nor rejected in* C

4.2.3 Complexity

In this section we provide a complexity analysis in terms of *computational complexity* of the proposed algorithms, and *number of algorithm calls* and *number of messages* imposed by distributed query evaluation.

Computational Complexity

The term *computational complexity* refers to the total number of operations imposed by a single call of the algorithms. Obviously, the complexity of the local algorithm, *local_alg*, is related only to the strict local rules and local literals of a context. By definition of *local_alg* and by the fact that there are no loops in the local context theories, it is trivial to prove that the complexity of *local_alg* is proportional to the number of strict local rules of a context, and to the total number of its local literals.

The complexity of $Stronger(A, B, T_i)$ is related to the total number of elements contained in the two sets, A and B. *Stronger* can be implemented in a slightly different way than that described in the pseudocode provided in the previous section. Specifically, it requires a process that identifies the *weakest* literal in each of the two sets based on the rank of the contexts that the literals are defined by, and a comparison of the *strength* of the weakest literals. The complexity of this process is proportional to the total number of elements contained in each set.

Support imposes one operation for each of the literals contained in each rule that contains the queried literal or its negation in its head. It also requires computing the *strongest* between the Supportive or Blocking Sets of all such rules. Taking into account the complexity of *Stronger*, and by the fact that, in the worst case, all rules of a context may be relevant, the complexity of *Support* is in the worst case proportional to the number of rules of a context theory and to the number of literals defined in the system.

By definition, $P2P\_DR$ calls $local\_alg$, $Support$ and $Stronger$ twice. Taking into account their complexitites, we obtain the following result for the complexity of $P2P\_DR$:

Proposition 4 *The number of operations imposed by one call of $P2P\_DR$ for the evaluation of a query for literal p_i is, in the worst case that all rules of C_i contain either p_i or $\sim p_i$ in their head and all literals defined in the system in their bodies, proportional to the number of rules in C_i, r_i, and to the total number n of literals defined in the system ($O(r_i, n)$).*

Algorithm Calls & and Number of Messages

Applying the $P2P\_DR$ algorithm to conduct distributed query evaluation in a very *dense* Multi-Context System can result in a huge amount of messages, which is undesirable in settings such as Ambient Intelligence environments, which involve wireless communications and devices with limited communication capabilities. Using the term *dense*, we imply systems, in which all contexts define mapping rules that associate their local context knowledge with the knowledge of the majority of the other contexts in the system. The worst case is that each context, for each of its local literals, has defined mappings that contain all possible combinations of foreign literals in their body.

Example 6. Consider a MCS, which consists of four different context theories: C_1, C_2, C_3 and C_4, with respective vocabularies: $V_1 = \{a_1\}$, $V_2 = \{a_2\}$, $V_3 = \{a_3\}$ and $V_4 = \{a_4\}$. Assume that C_1 contains the following mapping rules with head a_1:

$r_1^m : a_2, a_3, a_4 \Rightarrow a_1$
$r_2^m : a_2, a_3 \Rightarrow a_1$
$r_3^m : a_2, a_4 \Rightarrow a_1$
$r_4^m : a_3, a_4 \Rightarrow a_1$
$r_5^m : a_2 \Rightarrow a_1$
$r_6^m : a_3 \Rightarrow a_1$
$r_7^m : a_4 \Rightarrow a_1$

Suppose, also that the other contexts have defined similar sets of mapping rules for their local literals. In that case, for the evaluation of the truth value of a_1, a number of 204 calls of *P2P_DR* and *Support* is required, and since a remote call of *P2P_DR* induces two messages (query and response), 408 messages in total must be exchanged between the four contexts. $\square$

Remark. Obviously, this case could be avoided, since it is easy to notice that when someone has defined rules r_5^m, r_6^m and r_7^m, the remaining rules are unnecessary for the computation of the Supportive and Blocking Sets and the truth value of a_1. However, we use it to highlight the complexity of the problem in the worst case scenario.

Optimizations

A major factor that causes this huge amount of messages is that for each of the literals that are contained in the body of its mapping rules, a context may have to send multiple query messages. E.g. in Example 4, C_1 has to query about a_2 four different times, while during the whole query evaluation process, each of contexts C_2, C_3 and C_4 issue 32 different queries for each of two different literals from the set $\{a_2, a_3, a_4\}$. The question is whether are all those queries necessary, and if not which of them could be avoided and how.

One way to avoid unnecessary queries would be to keep track of the answers returned for each of the queries that a context imposes through *P2P_DR* to other contexts, so that these answers are available for any subsequent calls of the algorithm. We should note, however, that the answer returned by *P2P_DR* depends not only on the queried literal, but also on the history of the query.

Example 7. Consider a MCS that consists of the following four context theories:

C_0: $r_{01}^l :\rightarrow a_0$
C_1: $r_{11}^m : a_0 \Rightarrow a_1$, $r_{12}^m : a_2 \Rightarrow a_1$
C_2: $r_{21}^m : a_3 \Rightarrow a_2$
C_3: $r_{31}^m : a_1 \Rightarrow a_3$, $r_{32}^m : a_1, a_2 \Rightarrow b_3$

Suppose that a query about b_3 is issued to C_3. To evaluate the truth value of b_3, *P2P_DR* calls *Support*, which in turn issues queries for literals a_1 and a_2 in order to check the applicability of the only rule with head b_3, r^m_{32}. In the process of evaluating the answer for a_1, *P2P_DR* will be called successively for a_2 and a_3, will detect a cycle caused by literals a_1, a_2 and a_3, and will return *undefined* as answer for both a_3 and a_2. This result is caused by the fact that a_1 is in the history of the queries imposed for a_2 and a_3. Eventually, using rule r^m_{11}, *P2P_DR* will return *true* as an answer for a_1. For a_2, the other body literal of r^m_{32}, *P2P_DR* will return *true*, using rules r^m_{21}, r^m_{31}, r^m_{11} and r^l_{01}. During this query evaluation process, two different queries are issued to a_2. The first one is issued by C_1 with history $Hist_{a_2} = [b_3, a_1, a_2]$ and results in an *undefined* answer, while the second is issued by C_3 with history $Hist_{a_2} = [b_3, a_2]$ and results in a different answer (*true*). □

Based on these observations, we modify our algorithms as follows. For each context C_i, we use two structures that the algorithms may access to store or retrieve the results obtained during a query evaluation process:

- OUT_Q, which stores the results of queries for foreign literals of C_i that are contained in the bodies of mapping rules of C_i
- INC_Q, which stores the results of queries for local literals of C_i

OUT_Q contains records of the form $rec(p_j, Hist_{p_j}) : Ans_{p_j}$, where $p_j \notin V_i$, and $Hist_{p_i}$ is a set of local or foreign literals of C_i. Each such record contains Ans_{p_j} for a query for literal p_j with history $Hist_{p_j}$, which has already been evaluated during the same query evaluation process.

INC_Q contains records of the form $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, BS_{p_i}, SS_{p_i})$, where $p_i \in V_i$ and $Hist_{p_i}$ is a set of local or foreign literals of C_i. Each record represents the results that have already been obtained during the same query evaluation process for a query for literal p_i with history $Hist_{p_i}$. The results contained in one record include Ans_{p_i}, and (in case $Ans_{p_i} \neq false$) BS_{p_i} and SS_{p_i}. For each local literal p_i, INC_Q contains an additional record of the form $rec(p_i) : localAns_{p_i}$ that retains $localAns_{p_i}$, which is independent of the history of the query as we have assumed that there are no loops in the local context theories.

Using these two structures, $P2P\_DR$ and $Support$ are modified as follows. Before calling $local\_alg$ to compute the local answers for p_i and $\sim p_i$, $P2P\_DR$ checks whether there are already records for p_i in INC_Q, In case there are such records, it retrieves the local answers for p_i and $\sim p_i$ from INC_Q (lines 1-2, 9-10). Otherwise, after evaluating the local answers using $local\_alg$, $P2P\_DR$ creates appropriate records for p_i and $\sim p_i$ in INC_Q (lines 5,13). Also, after evaluating Ans_{p_i}, BS_{p_i} and SS_{p_i} using $Support$ and $Stronger$, $P2P\_DR$ creates an appropriate record for the query for p_i with history $Hist_{p_i}$ in INC_Q (lines 20,25,30,33). The pseudocode of the optimized version of the algorithm, $P2P\_DR^O$ is given below:

P2P_DR$^O(p_i, C_0, C_i, Hist_{p_i}, T_i, SS_{p_i}, BS_{p_i}, Ans_{p_i})$

1: **if** $rec(p_i) : localAns_{p_i} \in INC_Q$ **then**
2:     retrieve $localAns_{p_i}$ from $rec(p_i)$
3: **else**
4:     call $local\_alg(p_i, localAns_{p_i})$
5:     add record $rec(p_i) : localAns_{p_i}$ in INC_Q
6: **if** $localAns_{p_i} = true$ **then**
7:     $Ans_{p_i} \leftarrow true$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
8:     terminate
9: **if** $rec(\sim p_i) : localAns_{\sim p_i} \in INC_Q$ **then**
10:     retrieve $localAns_{\sim p_i}$ from $rec(\sim p_i)$
11: **else**
12:     call $local\_alg(\sim p_i, localAns_{\sim p_i})$
13:     add record $rec(\sim p_i) : localAns_{\sim p_i}$ in INC_Q
14: **if** $localAns_{\sim p_i} = true$ **then**
15:     $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
16:     terminate
17: call $Support^O(p_i, Hist_{p_i}, T_i, sup_{p_i}, unb_{p_i}, SS_{p_i}, BS_{p_i})$
18: **if** $unb_{p_i} = false$ **then**
19:     $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
20:     add record $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, \emptyset, \emptyset)$ in INC_Q and terminate
21: $Hist_{\sim p_i} \leftarrow (Hist_{p_i} - \{p_i\}) \cup \sim \{p_i\}$
22: call $Support^O(\sim p_i, Hist_{\sim p_i}, T_i, sup_{\sim p_i}, unb_{\sim p_i}, SS_{\sim p_i}, BS_{\sim p_i})$
23: **if** $sup_{p_i} = true$ and $(unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i})$ **then**
24:     $Ans_{p_i} \leftarrow true$
25:     add record $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, BS_{p_i}, SS_{p_i})$ in INC_Q and terminate
26:     **if** $C_0 \neq C_i$ **then**
27:         $SS_{p_i} \leftarrow \emptyset$,$BS_{p_i} \leftarrow \emptyset$

else if $sup_{\sim p_i} = true$ and $Stronger(BS_{p_i}, SS_{\sim p_i}, T_i) \neq BS_{p_i}$ **then**
  $Ans_{p_i} \leftarrow false$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
  add record $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, \emptyset, \emptyset)$ in INC_Q
else
  $Ans_{p_i} \leftarrow undefined$
  add record $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, BS_{p_i}, \emptyset)$ in INC_Q
  if $C_0 \neq C_i$ **then**
    $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$

Support is also modified as follows. Before calling *P2P_DR*^{O} to compute Ans_{b_t}, SS_{b_t} and BS_{b_t} for each of the literals b_t contained in the body of the rules with head p_i, *Support*^{O} (the optimized version of *Support*) checks whether there is a record for b_t and $Hist_{b_t}$ in INC_Q or OUT_Q. In case there is no such record, it calls *P2P_DR*^{O} and creates an appropriate record for (b_t,$Hist_{b_t}$) in INC_Q if b_t is a local literal of C_i, or in OUT_Q otherwise. This modification is achieved by replacing line 13 of *Support* with the pseudocode described below.

if $b_t \in V_i$ and $rec(b_t, Hist_{b_t}) : (Ans_{b_t}, BS_{b_t}, SS_{b_t}) \in INC_Q$ **then**
  retrieve $Ans_{b_t}, BS_{b_t}, SS_{b_t}$ from $rec(b_t, Hist_{b_t})$
else if $b_t \notin V_i$ and $rec(b_t, Hist_{b_t}) : Ans_{b_t} \in OUT_Q$ **then**
  retrieve Ans_{b_t} from $rec(b_t, Hist_{b_t})$
else
  call $P2P\_DR(b_t, C_i, C_t, Hist_{b_t}, T_t, SS_{b_t}, BS_{b_t}, Ans_{b_t})$
  if $b_t \in V_i$ **then**
    add record $rec(b_i, Hist_{b_t}) : (Ans_{b_t}, BS_{b_t}, SS_{b_t})$ in INC_Q
  else
    add record $rec(b_t, Hist_{b_t}) : Ans_{b_t}$ in OUT_Q

We should note that since we assume that the state of the network remains unchanged during a query evaluation process, but may change in the meantime between two consecutive query evaluation processes, the two structures INC_Q and OUT_Q are updated every time a new query is posed to the system.

Example 7 (continued). At the start of the query evaluation process for b_3, the two structures, INC_Q and OUT_Q, that each context retains are empty. In the end, the following records will have been created in INC_Q and OUT_Q of C_2:

$rec(a_2, [b_3, a_1, a_2]) : (undefined, \{a_3\}, \emptyset)$
$rec(a_2, [b_3, a_2]) : (true, \{a_3\}, \{a_3\})$
$rec(a_3, [b_3, a_1, a_2, a_3]) : undefined$
$rec(a_3, [b_3, a_2, a_3]) : true$

The first and the third records are created during the evaluation of the query for a_1 issued by C_3. The first record is retained in INC_Q and indicates that a query for a_2 with history $[b_3, a_1, a_2]$ has returned $Ans_{a_2} = undefined$, $BS_{a_2} = \{a_3\}$ and $SS_{a_2} = \emptyset$. The third record is retained in OUT_Q, and indicates that a query for a_3 with history $[b_3, a_1, a_2]$ has returned $Ans_{a_2} = undefined$. The second and the fourth records are created during the evaluation of the query for a_2 issued by C_3. □

Complexity Analysis of the Optimized Algorithms

Using the optimized algorithms described above, we aim at reducing not only the number of messages between the system contexts, but also the total number of operations performed by a context during a query evaluation process. Specifically, using OUT_Q, many messages are replaced with retrievals of relevant answers from OUT_Q. Using INC_Q, we replace queries imposed by a context to itself (recursive calls of $P2P\_DR$) with retrievals from INC_Q. In Example 4, the total number of algorithm calls that are required for the evaluation of the query for a_1 is reduced from 204 to 15, and the total number of messages is reduced from 408 to 30. The following proposition refers to the complexity of query evaluation using the optimized versions of $P2P\_DR$ and *Support*.

Proposition 5 *The total number of calls of $P2P\_DR^O$ that are required for the evaluation of a single query is in the worst case $O(n \times \sum P(n,k))$, where n stands for the total number of literals in the system, $\sum$ expresses the sum over $k = 0, 1, ..., n$, and $P(n,k)$ stands for the number of permutations with length k of n elements. If each of the literals in the system is defined by a different context, then the total number of messages exchanged between the system contexts for the evaluation of a query is $O(2 \times n \times \sum P(n,k))$.*

We should note that $\sum P(n,k)$ stands between 2^n and $n!2^n$. If we assume that there are no loops in the global knowledge base (acyclic MCS), then the history of a query is irrelevant to how it is evaluated. In this case, INC_Q and OUT_Q retain at most one record for each of the local and foreign literals of a context, and the complexity of query evaluation is reduced as follows.

Proposition 6 *In acyclic MCS, the total number of calls of $P2P\_DR^O$ that are required for the evaluation of a single query is in the worst case $O(c \times n)$, where c stands for the total number of contexts in the system, and n stands for the total number of literals in the system. If each of the literals in the system is defined by a different context, then the total number of messages exchanged between the system contexts for the evaluation of a query is $O(2 \times c \times n)$.*

4.3 Equivalent Global Defeasible Theory

The goal of the procedure that we describe in this section is the construction of a global defeasible theory $T_v(C)$, which produces the same results as the application of $P2P\_DR$ on a Multi-Context System C under the proof theory of the ambiguity blocking version of Defeasible Logic with superiority relation [8]. The existence of this procedure enables resorting to centralized reasoning by collecting the distributed context theories in a central entity and creating an equivalent defeasible theory. In addition, this result is typical of other works in the area of Peer-to-Peer reasoning, in which the distributed query evaluation algorithm is related to querying a single knowledge base that can be constructed (see, e.g. [2]).

The procedure follows three main steps:

1. The local strict rules of each context theory are added as strict rules in $T_v(C)$.

2. The local defeasible and mapping rules of each context theory are added as defeasible rules in $T_v(C)$.

3. For each pair of rules with contradictory conclusions, a priority relation is added taking into account the preference orderings of the system contexts.

The vocabulary used by $T_v(C)$ (V) is the union of the vocabularies of the unified context theories: $V = \bigcup V_i$. For the global theory $T_v(C)$, Lemma 4 holds as an immediate consequence of the first two steps of the procedure that constructs $T_v(C)$, and from the fact that each rule in a context C_i has a head labeled by a literal in V_i (the vocabulary of C_i).

Lemma 4 *For a literal p_i in V:*

(a) The set of strict rules with head p_i in $T_v(C)$ is the same with the set of local strict rules with head p_i in C_i.

(b) The set of defeasible rules with head p_i in $T_v(C)$ is the same with the set of local defeasible or mapping rules with head p_i in C_i.

The construction of the superiority relation of the global defeasible theory is achieved using the *Priorities* process described below. The role of this process is to augment $T_v(C)$, as this is derived from the first two steps of the procedure described above, with the additional required rule priorities considering the preference orderings of the system contexts.

Priorities

The derivation of priorities between conflicting rules (rules with contradictory conclusions) in $T_v(C)$ is a finite sequence $Pr = (Pr(1), ..., Pr(n))$, where each $Pr(i)$ can be one of the followings:

- The Supportive Set of a rule in $T_v(C)$ (a set of literals).
- A priority relation between two conflicting rules in $T_v(C)$
- The Supportive Set of a literal in $T_v(C)$ (a set of literals).

In this process we use two special elements: (a) w, to mark the rules that cannot be applied to support their conclusions; and (b) s, to mark the literals the truth value of which is derived from the strict rules of $T_v(C)$.

Overall, for a rule r_i, such that the literals in its body are logical consequences of the local strict rules in $T_v(C)$, *Priorities* assigns $\{s\}$ as its Supportive Set $(1(\alpha))$. $\{w\}$ is assigned as the Supportive Set of a rule that contains a foreign literal in its body, which is not a logical consequence of $T_v(C)$ $(1(\beta))$. For the rest of the rules, *Priorities* computes their Supportive Set as the union of foreign literals contained in their body with the Supportive Sets of the local literals in their body $(1(\gamma))$. The priority relation between two applicable rules with contradictory conclusions is computed based on the *Stronger* function, which takes into account the preference order of their context (2).

For a literal p_i, which is logical consequence of $T_v(C)$, *Priorities* assigns the Supportive Set of the *strongest* supportive rule with head p_i as the Supportive Set of p_i, using the *Stronger* function (3(α)). For the rest of the literals, $\{w\}$ is assigned as their Supportive Set (3(β)).

In the followings we denote the set of strict rules with head p_i in $T_v(C)$ as $R^s[p_i]$, and the set of defeasible rules with head p_i in $T_v(C)$ as $R^d[p_i]$.

1. *If* $Pr(i+1) = S_{r_i}$ *then* $r_i \in T_v(C)$ *and either*
   (α) $S_{r_i} = \{s\}$ *and* $r_i \in R^s[p_i]$ *and*
   $\forall a_i \in body(r_i)$, $S_{a_i} = \{s\} \in Pr(1...i)$ *or*
   (β) $S_{r_i} = \{w\}$, *and* $r_i \in R^d[p_i]$ *and*
   $\exists a_j \in body(r_i)$: $a_j \notin V_i$, $S_{a_j} \in Pr(1...i)$ *and* $w \in S_{a_j}$ *or*
   (γ) (γ_1) $\forall a_i \in body(r_i) \cap V_i$: $S_{a_i} \in Pr(1...i)$ *and*
   (γ_2) $\forall a_j \in body(r_i) - V_i$: $S_{a_j} \in Pr(1...i)$ *and* $w \notin S_{a_j}$ *and*
   (γ_3) $S_{r_i} = (\bigcup_{a'_i} S_{a'_i}) \cup (\bigcup_{a_j} a_j)$, *where*
   a'_i *are the literals in the body of* r_i *s.t.* $a'_i \in V_i$ *and* $S_{a'_i} \neq \{s\}$

2. *If* $Pr(i+1) = r_i > s_i$ *then* $r_i, s_i \in T_v(C)$ *and*
   (α) $head(r_i) =\sim head(s_i)$ *and*
   (β) $S_{r_i}, S_{s_i} \in Pr(1...i)$ *and*
   (γ) $w \notin S_{r_i}$, $w \notin S_{s_i}$, $S_{r_i} \neq \{s\}$, $S_{s_i} \neq \{s\}$ *and*
   (δ) $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$

3. *If* $Pr(i+1) = S_{p_i}$ *then either*
   (α) $\exists r_i \in R[p_i]$: $S_{r_i} \in Pr(1...i)$ *and* $S_{p_i} = S_{r_i}$ *and* $w \notin S_{r_i}$ *and either*
   (α_1) $r_i \in R^s[p_i]$ *and* $S_{r_i} = \{s\}$ *or*
   (α_2) ($\alpha_{2.1}$) $\forall s_i \in R[\sim p_i]$:
   ($\alpha_{2.1.1}$) $S_{s_i} \in Pr(1...i)$ *and*
   ($\alpha_{2.1.2}$) $S_{s_i} \neq \{s\}$ *and*
   ($\alpha_{2.1.3}$) $w \in S_{s_i}$ *or* $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$ *and*
   ($\alpha_{2.2}$) $\forall t_i \in R[p_i]$:
   ($\alpha_{2.2.1}$) $S_{t_i} \in Pr(1...i)$ *and*
   ($\alpha_{2.2.2}$) $S_{t_i} \neq \{s\}$ *and*
   ($\alpha_{2.2.3}$) $w \in S_{t_i}$ *or* $Stronger(S_{r_i}, S_{t_i}, T_i) \neq S_{t_i}$ *or*

(β) $S_{p_i} = \{w\}$ *and either*

(β_1) $\forall r_i \in R[p_i]$: $S_{r_i} \in Pr(1...i)$ *and* $w \in S_{r_i}$ *or*

(β_2) $\exists s_i \in R[\sim p_i]$:

($\beta_{2.1}$) $S_{s_i} \in Pr(1...i)$ *and* $w \notin S_{s_i}$ *and* $S_{s_i} \neq \{s\}$ *and*

($\beta_{2.2}$) $\forall r_i \in R[p_i]$: $S_{r_i} \in Pr(1...i)$ *and* $S_{r_i} \neq \{s\}$ *and either*

($\beta_{2.2.1}$) $w \in S_{r_i}$ *or*

($\beta_{2.2.2}$) $w \notin S_{r_i}$ *and* $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ *or*

(β_3) $S_{\sim p_i} \in Pr(1...i)$ and $S_{\sim p_i} = \{s\}$

$Pr(n)$ will contain the Supportive Sets of all rules and literals in $T_v(C)$, and the required priority relations between all conflicting rules in $T_v(C)$. It is easy to verify that the process described above is deterministic in the sense that for a given MCS C, the same set of elements is always contained in $Pr(n)$ regardless of the order of contexts C_i in C, and the orders of the rules in each context C_i in C.

Example 3 (continued) Following the procedure described above, taking as input the context theories of the MCS depicted in Figure 3.2, all strict local rules will be added as strict rules in $T_v(C)$, all local defeasible and mapping rules will be added as defeasible rules in $T_v(C)$, and *Priorities* will proceed as follows:

- The first four derivation steps will produce: $S_{r^l_{31}} = S_{r^l_{41}} = S_{r^l_{51}} = S_{r^l_{62}} = \{s\}$.
- The next five steps will produce: $S_{a_3} = S_{a_4} = S_{a_5} = S_{\neg a_6} = \{s\}$ and $S_{a_6} = \{w\}$.
- The next three steps will be: $Pr(10) = S_{r^m_{21}} = \{a_5\}$, $Pr(11) = S_{r^m_{22}} = \{w\}$ and $Pr(12) = S_{r^m_{13}} = \{a_3, a_4\}$.
- The thirteenth step will be: $Pr(13) = S_{a_2} = \{a_5\}$.
- The fourteenth step will be: $Pr(14) = S_{r^m_{12}} = \{a_2\}$
- The fifteenth step will be: $Pr(15) = r^m_{12} > r^m_{13}$.
- The last three steps of the process will be: $Pr(16) = S_{a_1} = \{a_2\}$, $Pr(17) = S_{r^l_{11}} = \{a_2\}$ and $Pr(18) = S_{x_1} = \{a_2\}$.

Eventually, the only priority relation that will be added to $T_v(C)$ is $r^m_{12} > r^m_{13}$. $\square$

Example 1 (continued) Taking as input the context theories of the MCS of Example 1, all strict local rules of the theories will be added as strict rules in $T_v(C)$, all local defeasible and mapping rules of the theories will be added as defeasible rules in $T_v(C)$, and *Priorities* will proceed as follows:

- The first seven derivation steps will produce: $S_{r^l_{11}} = S_{r^l_{12}} = S_{r^l_{21}} = S_{r^l_{31}} = S_{r^l_{41}} = S_{r^l_{51}} = \{s\}$ and $S_{silent\_mode_1} = \{w\}$.
- The next seven steps will produce: $S_{incoming\_call_1} = S_{normal\_mode_1} = S_{classtime_2} = S_{location\_RA201_3} = S_{projector(off)_4} = S_{detected(1)_5} = \{s\}$ and $S_{r^l_{14}} = \{w\}$.
- The next three steps will produce: $S_{r^m_{42}} = \{detected(1)_5\}$, $Pr(10) = S_{r^m_{15}} = \{classtime_2, location\_RA201_3\}$ and $S_{\neg ring_1} = \{w\}$.
- $Pr(18) = S_{\neg class\_activity_4} = \{detected(1)_5\}$.
- $Pr(19) = S_{r^m_{16}} = \{\neg class\_activity_4\}$.
- The next three steps will produce $r^m_{16} > r^m_{15}$, $S_{\neg lecture_1} = \{\neg class\_activity_4\}$, $S_{lecture_1} = \{w\}$.
- The last three steps of the process will be: $Pr(23) = S_{r^d_{13}} = \{\neg class\_activity_4\}$ and $Pr(24) = S_{ring_1} = \{\neg class\_activity_4\}$.

Eventually, the only priority relation that will be added to $T_v(C)$ is $r^m_{16} > r^m_{15}$. □

Relation with Defeasible Logic

In case, $T_v(C)$ contains no loops (there is no literal p_i in $T_v(C)$, such that p_i has an infinite proof tree), the following statements describe the relationship between $T_v(C)$ and the distributed context theories C_i.

Lemmas 5 and 6 refer to the association of the answers produced by *local_alg* and *P2P_DR* with the constructs computed by *Priorities*.

Lemma 5 *For a literal p_i in V_i, local_alg computes*
(1) $localAns_{p_i} = true$ *iff* $S_{p_i} \in Pr(1...n)$ *and* $S_{p_i} = \{s\}$
(2) $localAns_{p_i} = false$ *iff* $S_{p_i} \in Pr(1...n)$ *and* $S_{p_i} \neq \{s\}$

Lemma 6 *For a literal p_i in V_i, P2P_DR computes*

(1) $localAns_{p_i} = false$, $Ans_{p_i} = true$ *and* $SS_{p_i} = \Sigma$ *iff* $S_{p_i} \in Pr(1...n)$ *and* $S_{p_i} = \Sigma$ *and* $\Sigma \neq \{s\}$, $\Sigma \neq \{w\}$.

(2) $Ans_{p_i} = false$ *iff* $S_{p_i} \in Pr(1...n)$ *and* $S_{p_i} \neq \{w\}$

Propositions 7 and 8 are consequences of the above lemmas and describe the association between the answers produced by *local_alg* and *P2P_DR* and the results produced by the application of the proof theory of the ambiguity version of Defeasible Logic [8] on $T_v(C)$.

Proposition 7 *For a literal p_i in V_i, local_alg computes*

(1) $localAns_{p_i} = true$ *iff* $T_v(C) \vdash +\Delta p_i$

(2) $localAns_{p_i} = false$ *iff* $T_v(C) \vdash -\Delta p_i$

Proposition 8 *For a literal p_i in V_i, P2P_DR computes*

(1) $Ans_{p_i} = true$ *iff* $T_v(C) \vdash +\partial p_i$

(2) $Ans_{p_i} = false$ *iff* $T_v(C) \vdash -\partial p_i$

4.4 Summary of the results

The main results regarding the formal properties of *P2P_DR* that we described in this section are summarized as follows.

- The algorithm always terminates returning one of the values *true*, *false* and *undefined* for the queried literal (Proposition 1).
- The algorithm is sound and complete with respect to the argumentation framework (Propositions 2 and 3).
- The worst case complexity of the algorithm in terms of number of messages is between 2^n and $n!2^n$ (Proposition 5), while in case of acyclic MCS, it is $O(n, c)$ (Proposition 6), where n stands for the total number of system literals and c for the total number of contexts.
- There is a standard process that takes as input the distributed context theories and their preference orderings and creates a global unified theory of Defeasible Logic, which in case there are no loops in the global theory, produces equivalent results with *P2P_DR* (Propositions 7 and 8).

Chapter 5

Alternative Strategies for Conflict Resolution

The algorithm described in Chapter 4 ($P2P\_DR$) resolves conflicts that arise when mutually inconsistent information is imported from different contexts based on the rank of these contexts in the preference ordering of the context that imports this information. This chapter describes three alternative strategies for conflict resolution (*Strict-Weak Answers*, *Propagating Mapping Sets* and *Complex Mapping Sets*), which differ in the type and extent of context information that is used to evaluate the quality of the imported knowledge. The intuition behind these strategies is that imported knowledge should be evaluated not only based on their "source" - the context that the knowledge is imported by - but also on the way that the "source" acquired this knowledge. The following sections discuss the features of the three strategies, explain their main differences through examples, and describe their implementation in three alternative versions of $P2P\_DR$.

5.1 Strict-Weak Answers

The distinct feature of *Strict-Weak Answers*, compared to the strategy implemented by $P2P\_DR$ (which for the rest of the thesis will be referred to as *Single Answers*), is that imported knowledge is not only evaluated based on its "source" (the context that it is imported by), but also based on whether the "source" derived this knowledge strictly - based on its strict local rules. The intuition behind this is that the case that the imported knowledge is part of the local context knowledge of the "source" should be treated differently (preferred) to the case that the "source" may have used knowledge imported from third parties to derive this knowledge. For example, in the case of the system in Example 1 (Figure 1.1), the knowledge that the mobile phone imports from the laptop and the localization service is part of their local knowledge bases, and hence should be preferred to the knowledge that is imported from the classroom manager, which is derived based on information that the classroom manager imports from an external source.

5.1.1 Distributed Query Evaluation

The version of the distributed algorithm that implements this strategy, $P2P\_DR_{SWA}$, uses two types of answers for the literals with a positive truth value:

- a *strict* answer indicates that the truth value is derived from the strict rules of the queried context;
- a *weak* answer indicates that the truth value is derived from the combination of local and mapping rules of the queried context

For *undefined* answers, there is no need to define two different types. Since we have assumed that there are no loops in the local context theories, an *undefined* answer cannot be derived using only the local strict rules of a context. $P2P\_DR_{SWA}$ follows the four main steps of $P2P\_DR$, but with the following modifications:

- For a literal p_i, Ans_{p_i} can take one of the following four values:
  1. $str(true)$, indicating a strictly derived positive truth value
  2. $weak(true)$ indicating a non-strictly derived positive truth value
  3. $undefined$
  4. $false$
- An element of a Supportive/Blocking Set is actually a signed literal; the sign of the literal indicates whether the truth value of the literal is derived from the local strict rules of the queried context (e.g. $+p_i$), or from the combination of the strict rules and the local defeasible and mapping rules of the context ($-p_i$).

- The strength of an element in a Supportive/Blocking Set is determined primarily by the type of answer (*strict* answers are considered stronger than *weak* ones), and secondly by the rank of the queried context in the preference order of the querying context.

The pseudocode of $P2P\_DR_{SWA}$ is derived from $P2P\_DR$ as follows:

- Lines 2-4 are replaced by:

  if $localAns_{p_i} = true$ **then**
  $Ans_{p_i} \leftarrow str(true)$, $SS_{p_i} \leftarrow \emptyset$, $BS_{p_i} \leftarrow \emptyset$
  terminate

- Lines 15-16 are replaced by:

  if $sup_{p_i} = true$ and $(unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i})$ **then**
  $Ans_{p_i} \leftarrow weak(true)$

$local\_alg$ remains unchanged, while *Support* is modified as follows:

- Lines 8-10 are replaced by:

  if $b_t \in Hist_{p_i}$ **then**
  $cycle(r_i) \leftarrow true$
  $BS_{r_i} \leftarrow BS_{r_i} \cup \{-d_t\}$

- Lines 18-19 are replaced by:

  if $b_t \notin V_i$ **then**
  $BS_{r_i} \leftarrow BS_{r_i} \cup \{-b_t\}$

- Lines 23-25 are replaced by:

  if $b_t \notin V_i$ and $Ans_{b_t} = str(true)$ **then**
  $BS_{r_i} \leftarrow BS_{r_i} \cup \{+b_t\}$
  $SS_{r_i} \leftarrow SS_{r_i} \cup \{+b_t\}$
  else if $b_t \notin V_i$ and $Ans_{b_t} = weak(true)$ **then**
  $BS_{r_i} \leftarrow BS_{r_i} \cup \{-b_t\}$
  $SS_{r_i} \leftarrow SS_{r_i} \cup \{-b_t\}$

Finally, the *Stronger* function is also modified as follows:

Parameters

A, B: sets of signed literals of the form $+p_i/-p_i$

$\underline{C_1}$	$\underline{C_2}$	$\underline{C_3}$
$r^l_{11} : a_1 \rightarrow x_1$	$r^l_{21} : c_2 \rightarrow a_2$	$r^l_{31} : \rightarrow a_3$
$r^m_{12} : a_2 \Rightarrow a_1$	$r^l_{22} : b_2 \rightarrow a_2$	
$r^m_{13} : a_3, a_4 \Rightarrow \neg a_1$	$r^m_{23} : b_5 \Rightarrow b_2$	
	$r^m_{24} : b_6 \Rightarrow b_2$	
$\underline{C_4}$	$\underline{C_5}$	$\underline{C_6}$
$r^l_{41} : \rightarrow a_4$	$r^l_{51} : \rightarrow b_5$	$r^l_{61} : \rightarrow b_6$

Figure 5.1: A MCS of Six Context Theories (example 6)

T_i: a preference order

Stronger(A, B, T_i)

if $\exists b_l$: $-b_l \in B$ and $\forall \pm a_k \in A$ either $+a_k \in A$ or C_k has lower rank than C_l in T_i **then**
    Stronger $= A$
else if $\exists a_k$: $-a_k \in A$ and $\forall \pm b_l \in B$ either $+b_l \in B$ or C_l has lower rank than C_k in T_i **then**
    Stronger $= B$
else if $\forall \pm a_k \in A, \pm b_l \in B$: $+a_k \in A$ and $+b_l \in B$ **then**
    if $\exists + b_l \in B$: $\forall + a_k \in A$, C_k has lower rank than C_l in T_i **then**
        Stronger $= A$
    else if $\exists + a_k \in A$: $\forall + b_l \in B$, C_l has lower rank than C_k in T_i **then**
        Stronger $= B$
else
    Stronger $=$ *None*

Example 6. Consider that a query about x_1 is issued to C_i in the MCS depicted in Figure 5.1. An interesting feature of this system is that a_3 and a_4, which constitute the premises of r^m_{13}, are strict consequences of the local theories of C_3 and C_4 respectively, while a_2, which is the only premise of the rule that is in conflict with r^m_{13} (r^m_{12}) is not a strict consequence of the local theory of C_2.

P2P_DR, the algorithm that implements the *Single Answers* strategy, receives positive answers (*true*) for a_2, a_3 and a_4 (from the instances of the algorithm called by C_2, C_3 and C_4 respectively), and resolves the conflict that arises for literal a_1 by comparing its Supportive Set, $SS_{a_1} = SS_{r_{12}} = \{a_2\}$, with the Blocking Set of $\neg a_1$,

$BS_{\neg a_1} = BS_{r_{13}} = \{a_3, a_4\}$. Assuming that the preference order defined by C_1 is $T_1 = [C_4, C_2, C_6, C_3, C_5]$, *P2P_DR* determines that SS_{a_1} is *stronger* than $BS_{\neg a_1}$ (as C_2 precedes C_3 in T_1) and returns a positive answer for a_1 and eventually for x_1 as well.

On the other hand, *P2P_DR*^{SWA} proceeds as follows: For a_3 and a_4, it returns $Ans_{a_3} = str(true)$ and $Ans_{a_3} = str(true)$, as both are strict local conclusions of C_3 and C_4, respectively. For a_2, it returns $Ans_{a_3} = weak(true)$, as for the evaluation of this answer it uses both local and mapping rules of C_2. Hence, the Supportive Sets of rules r_{12} and r_{13} are respectively: $SS_{r^m_{12}} = \{-a_2\}$, and $SS^m_{r_{13}} = \{+a_3, +a_4\}$. Hence, $BS_{a_1} = \{-a_2\}$ and $SS_{\neg a_1} = \{+a_3, +a_4\}$, and $SS_{\neg a_1}$ is *stronger* than BS_{a_1}. Eventually, *P2P_DR*^{SWA} returns negative truth values for a_1 and, hence for x_1 as well. □

Example 1 (continued). In the MCS of example 1 (Figure 1.1), *P2P_DR* receives positive answers (*true*) for $classtime_2$, $location\_RA201_3$ and $\neg class\_activity_4$ (from the instances of the algorithm called by C_2, C_3 and C_4 respectively), and resolves the conflict that arises for
$\neg lecture_1$ by comparing its Supportive Set, $SS_{\neg lecture_1} = SS_{r^m_{16}} = \{\neg class\_activity_4\}$, with the Blocking Set of $lecture_1$, $BS_{lecture_1} = BS_{r^m_{15}} = \{classtime_2, location\_RA201_3\}$. Using $T_1 = [C_4, C_3, C_2, C_5]$, *P2P_DR* determines that $SS_{\neg lecture_1}$ is *stronger* than $BS_{lecture_1}$ (as C_4 precedes both C_2 and C_3 in T_1) and returns a positive answer for $\neg lecture_1$ and eventually for $ring_1$ as well.

On the other hand, the version of the algorithm that implements *Strict-Weak Answers*, *P2P_DR*^{SWA}, returns $Ans_{classtime_2} = str(true)$ and $Ans_{location\_RA201_3} = str(true)$, as both are strict local conclusions of C_2 and C_3 respectively. For
$\neg class\_activity_4$, it returns $Ans_{\neg class\_activity_4} = weak(true)$, as for the evaluation of this answer it uses both local and mapping rules of C_4. Hence, the Supportive Sets of rules r^m_{15} and r^m_{16} are respectively: $SS_{r^m_{12}} = \{+classtime_2, +location\_RA201_3\}$, and $SS^m_{r_{13}} = \{-\neg class\_activity_4\}$, and *P2P_DR*^{SWA} evaluates a negative truth value for $\neg lecture_1$ and, eventually returns a negative truth value for $ring_1$ as well. □

5.1.2 Complexity Analysis

The implementation of the *Strict-Weak Answers* strategy does not require drastic modifications to the algorithms that implement the *Single Answers* strategy. Specifically, *local_alg* remains as it is, while the modifications of *Support*, *Stronger* and *P2P_DR* do not affect the overall complexity of the algorithms. Therefore, Proposition 4 holds as it is for the versions of the algorithms that implement the *Strict-Weak Answers* strategy as well.

Similarly with the case of $P2P\_DR^O$, which implements the *Single Answers* strategy, $P2P\_DR_{SWA}$ can also be optimized using structures that retain the answers for incoming and outgoing queries. In fact, $P2P\_DR^O_{SWA}$ uses structures of exactly the same form with INC_Q and OUT_Q. The only difference is that for a record for $(p_j, Hist_{p_j})$, the field that corresponds to Ans_{p_j} is filled with one of the values: $str(true)$, $weak(true)$, $undefined$ and $false$.

Using these optimizations, the number of algorithm calls and messages is reduced in the same way with the *Single Answers* strategy, to $O(n \times \sum P(n,k))$ and $O(2 \times n \times \sum P(n,k))$ respectively for the general case, and $O(c \times n)$ and $O(2 \times c \times n)$ respectively for acyclic MCS, where c stands for the total number of contexts and n for the total number of literals in the system.

5.2 Propagating Mapping Sets

The third strategy, *Propagating Mapping Sets*, goes one step further compared to *Strict-Weak Answers*. Similarly with the second strategy, the evaluation of the imported knowledge is based on how the "source" of this knowledge infers this knowledge. In *Strict-Weak Answers*, we only care about whether the imported knowledge is inferred from the strict local knowledge of the "source" or it is derived from both its local theory and its mappings (and therefore from the knowledge of other contexts). *Propagating Mapping Sets* further requires the "source" to return information about the identity of other contexts that are involved in the derivation of the imported knowledge, and evaluates it based on the ranks of these contexts in the preference ordering of the context that imports this knowledge. In the MCS of example 1 (Figure 1.1), following this strategy, the classroom manager does not only notify the mobile phone that there is no class activity in the classroom, but also informs it that to reach this conclusion it had to import knowledge from the person detection service. Hence, the evaluation of this knowledge will take into account the preference that the mobile phone has in both the classroom manager and the person detection service.

5.2.1 Distributed Query Evaluation

The only modifications of *P2P_DR* that are required to implement this strategy are two:

- the Supportive Set and the Blocking Set of the queried literal are always returned along with the computed truth value - in contrast with *P2P_DR*, which in case that a query is posed by some other context, the queried context just returns the truth value of the literal it is queried about.
- The Supportive (Blocking) Set of a rule is the union of the foreign literals in the body of the rule and the Supportive (Blocking) Sets of all literals in the body of the rule - in contrast with *P2P_DR*, where the Supportive (Blocking) Sets of the foreign literals are not taken into account.

The pseudocode of $P2P\_DR_{PS}$, the version of the distributed algorithm that implements *Propagating Mapping Sets*, is derived from *P2P_DR* as follows: Lines 15-24 are replaced by:

if $sup_{p_i} = true$ and ($unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i}$) **then**
  $Ans_{p_i} \leftarrow true$
else if $sup_{\sim p_i} = true$ and $Stronger(BS_{p_i}, SS_{\sim p_i}, T_i) \neq BS_{p_i}$ **then**
  $Ans_{p_i} \leftarrow false$, $SS_{p_i} = \emptyset$, $BS_{p_i} = \emptyset$
else
  $Ans_{p_i} \leftarrow undefined$

The pseudocodes of *local_alg* and *Stronger* remain unchanged, while *Support* is modified as follows:

- Lines 8-10 are replaced by:
  if $b_t \in Hist_{p_i}$ **then**
    $cycle(r_i) \leftarrow true$
    $BS_{r_i} \leftarrow BS_{r_i} \cup (\bigcup\{d_t\})$ {d_t are the foreign literals of C_i added in $Hist_{p_i}$ after b_t including b_t in case $b_t \notin V_i$}
- Lines 18-19 are replaced by:
  if $b_t \notin V_i$ **then**
    $BS_{r_i} \leftarrow BS_{r_i} \cup BS_{b_t} \cup \{b_t\}$

- Lines 23-25 are replaced by:
  if $b_t \notin V_i$ **then**
  $BS_{r_i} \leftarrow BS_{r_i} \cup BS_{b_t} \cup \{b_t\}$
  $SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t} \cup \{b_t\}$

Example 6 (continued). In the MCS depicted in Figure 5.1, $P2P\_DR_{PS}$ returns positive (*true*) answers for a_3, a_4, b_5 and b_6 along with empty Supportive and Blocking Sets, as all of them are conclusions of C_3, C_4, C_5 and C_6 respectively. Called by C_2, *Support* computes $SS_{r^m_{23}} = \{b_5\}$ and $SS_{r^m_{24}} = \{b_6\}$. Assuming that the preference ordering defined by C_2 is $T_2 = [C_4, C_3, C_5, C_2, C_6]$, *Support* determines that $SS_{r^m_{23}}$ is stronger than $SS_{r^m_{24}}$, and $P2P\_DR_{PS}$ returns *true* as an answer for b_2 and a_2 and $SS_{a_2} = SS_{b_2} = \{b_5\}$. For r^m_{12} and r^m_{13}, *Support* respectively computes $SS_{r^m_{12}} = \{a_2, b_5\}$ and $SS_{r^m_{13}} = \{a_3, a_4\}$. According to $T_1 = [C_4, C_2, C_6, C_3, C_5]$, C_5 does not precede neither C_3 nor C_4 in T_1. Therefore, $SS_{\neg a_1} = SS_{r^m_{13}}$ is computed to be *stronger* than $BS_{a_1} = SS_{r^m_{12}}$, and $P2P\_DR_{PS}$ returns *false* as answer for a_1, and eventually $Ans_{x_1} = false$. □

Example 1 (continued) In the MCS of Example 1 (Figure 1.1), $P2P\_DR_{PS}$ returns positive (*true*) answers for $classtime_2$, $location\_RA201_3$, $projector(off)_4$ and $detected(1)_5$ along with empty Supportive and Blocking Sets, as all of them are included in the strict local knowledge of C_2 (laptop), C_3 (localization service), C_4 (classroom manager) and C_5 (person detection service) respectively. For $\neg class\_activity_4$, it returns a positive (*true*) answer along with its Supportive Set $SS_{\neg class\_activity_4} = \{detected(1)_5\}$. For the two conflicting rules, r^m_{15} and r^m_{16}, *Support* respectively computes $SS_{r^m_{15}} = \{classtime_2, location\_RA201_3\}$ and $SS_{r^m_{16}} = \{\neg class\_activity_4, detected(1)_5\}$. According to $T_1 = [C_4, C_3, C_2, C_5]$, C_5 does not precede neither C_3 nor C_4 in T_1. Therefore, $SS_{lecture_1} = SS_{r^m_{15}}$ is computed to be *stronger* than $BS_{\neg lecture_1} = SS_{r^m_{16}}$, and $P2P\_DR_{PS}$ returns *false* as answer for $\neg lecture_1$, and eventually $Ans_{ring_1} = false$. □

5.2.2 Complexity Analysis

The implementation of the *Propagating Mapping Sets* strategy requires each context to return the Supportive and Blocking Sets of the queried literal, along with the answer that indicates its truth value. This affects the overall complexity of distributed query evaluation as follows: The overall complexity of the algorithms remains the same with the case of *Single Answers*. However, the *worst case* is not the same in the two cases. The worst case in *Single Answers* is when all rules of C_i (the context that evaluates the query) contain either p_i or $\sim p_i$ in their head and all literals defined in the system in their bodies. In *Propagating Mapping Sets*, the worst case is when all rules of C_i contain either p_i or $\sim p_i$ in their head, and the Supportive / Blocking Sets of all rules contain all literals defined in the system, which means that the applicability of each rule in C_i depends on the truth values of all literals defined in the system. Obviously, the worst case described for *Single Answer* is a subcase of the case that we describe here. This result is described in the Proposition below.

Proposition 9 *The number of operations imposed by one call of $P2P\_DR$ for the evaluation of a query for literal p_i is in the worst case that all rules of C_i contain either p_i or $\sim p_i$ in their head, and the applicability of each rule in C_i depends on the truth values of all literals defined in the system, proportional to the number of rules in C_i, and to the total number of literals in the system.*

Similarly with the case of $P2P\_DR^O$, which implements the *Single Answers* strategy, $P2P\_DR_{PS}$ can also be optimized using structures that retain the answers for incoming and outgoing queries. In this case both INC_Q and OUT_Q contain records of the form: $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, BS_{p_i}, SS_{p_i})$ for each query about local/foreign p_i with history $Hist_{p_i}$ that has already been evaluated.

Using these optimizations, the number of algorithm calls and messages is reduced in the same way with the *Single Answers* strategy, to $O(n \times \sum P(n,k))$ and $O(2 \times n \times \sum P(n,k))$ respectively for the general case, and $O(c \times n)$ and $O(2 \times c \times n)$ respectively for acyclic MCS, where c stands for the total number of contexts and n for the total number of literals in the system.

5.3 Complex Mapping Sets

The main feature of *Propagating Mapping Sets* is that a context that imports knowledge from another context, requires the "source" to return information about the identity of all other contexts that are involved in the derivation of the imported knowledge. Specifically, the "source" returns a set of literals that corresponds to the *most preferred* reasoning chain that leads to the inferred knowledge. However, preference is subjective.

The *most preferred* between two or more different reasoning chains may vary according to the viewpoint of two different contexts. The fourth strategy, *Complex Mapping Sets*, has the distinct feature that the *most preferred* between two or more reasoning chains is not determined by the *queried* context, but by the context that imports the knowledge. In the MCS used in Example 1, for each piece of knowledge that is exchanged between the ambient agents, there is only one reasoning chain that leads to its inference. Suppose, however, that there are two different services that provide knowledge about the presence of people in the classroom, C_5 and C_6, and that both detect one person in the classroom. Using the strategy described in the previous section (*Propagating Mapping Sets*), the classroom manager will use its local preference ordering to determine which of them is *preferred* (e.g. more trustworthy), and will return the identity of that service (e.g. C_5) to the mobile phone. In this case, the mobile phone cannot be aware that C_6 also provided the same knowledge, and will evaluate the knowledge it imports from the classroom manager based on its preference in the classroom manager and C_5. Following *Complex Mapping Sets*, the classroom manager will inform the mobile phone that there are two different ways to infer that there is no activity in the classroom; one that involves knowledge derived from C_5, and another one that involves the local knowledge of C_6. In this case, the mobile phone will separately evaluate the two different reasoning chains, using its own preference in the two person detection services. If at least one of these chains is preferred to the reasoning chains that lead to contradictory conclusions, the mobile phone will be able to use this knowledge (that there is no class activity) to derive further conclusions. Obviously, this approach is the richest w.r.t. the extent of context knowledge that it exploits, but also the one with the highest computational complexity.

5.3.1 Distributed Query Evaluation

To support the features of *Complex Mapping Sets*, *P2P_DR* is modified as follows:

- the Supportive Set and the Blocking Set of the queried literal are always returned along with the computed truth value.
- The Supportive Set (Blocking Set) of a literal is actually the set of the Supportive Sets (Blocking Sets) of all rules that can be applied to support this literal.

- The Supportive Set (Blocking Set) of a rule is the *Union Product* (Definition 18) of the Supportive Sets (Blocking Sets) of all literals in the body of the rule.

Definition 18 *Let A, B be two sets, the elements of which are sets of literals. Their Union Product is defined as:*

$$A \otimes B = \{a_i \cup b_j | a_i \in A, b_j \in B\}$$

The Union Product of n sets $A_1, A_2, ..., A_n$ is defined as follows:

$$\bigotimes_i A_i = (...((A_1 \otimes A_2) \otimes A_3) ... \otimes A_n)$$

$P2P\_DR_{CS}$ is derived from $P2P\_DR$ by replacing lines 15-24 with:

if $sup_{p_i} = true$ and ($unb_{\sim p_i} = false$ or $\exists A \in BS_{p_i}$: $\forall B \in SS_{\sim p_i}$ $Stronger(A, B, T_i) = A$) **then**
    $Ans_{p_i} \leftarrow true$
else if $sup_{\sim p_i} = true$ and $\exists B \in SS_{\sim p_i}$: $\forall A \in BS_{p_i}$ $Stronger(A, B, T_i) \neq A$ **then**
    $Ans_{p_i} \leftarrow false$, $SS_{p_i} = \emptyset$, $BS_{p_i} = \emptyset$
else
    $Ans_{p_i} \leftarrow undefined$

The pseudocodes of $local\_alg$ and $Stronger$ remain unchanged, while $Support$ is modified as follows:

Support$(p_i, Hist_{p_i}, T_i, sup_{p_i}, unb_{p_i}, SS_{p_i}, BS_{p_i})$
$sup_{p_i} \leftarrow false$
$unb_{p_i} \leftarrow false$
for all $r_i \in R[p_i]$ **do**
    $cycle(r_i) \leftarrow false$
    $SS_{r_i} \leftarrow \emptyset$
    $BS_{r_i} \leftarrow \emptyset$
    for all $b_t \in body(r_i)$ **do**
        if $b_t \in Hist_{p_i}$ **then**
            $cycle(r_i) \leftarrow true$
            $BS_{r_i} \leftarrow BS_{r_i} \otimes (\bigcup\{d_t\})$ {d_t are the foreign literals of C_i added in $Hist_{p_i}$ after b_t including b_t in case $b_t \notin V_i$}
        else
            $Hist_{b_t} \leftarrow Hist_{p_i} \cup \{b_t\}$
            call $P2P\_DR(b_t, C_i, C_t, Hist_{b_t}, T_t, SS_{b_t}, BS_{b_t}, Ans_{b_t})$

```
        if Ans_{b_t} = false then
            stop and check the next rule
        else if Ans_{b_t} = undefined or cycle(r_i) = true then
            cycle(r_i) ← true
            if b_t ∉ V_i then
                BS_{r_i} ← BS_{r_i} ⊗ (BS_{b_t} ⊗ {b_t})
            else
                BS_{r_i} ← BS_{r_i} ⊗ BS_{b_t}
        else
            if b_t ∉ V_i then
                BS_{r_i} ← BS_{r_i} ⊗ (BS_{b_t} ⊗ {b_t})
                SS_{r_i} ← SS_{r_i} ⊗ (SS_{b_t} ⊗ {b_t})
            else
                BS_{r_i} ← BS_{r_i} ⊗ BS_{b_t}
                SS_{r_i} ← SS_{r_i} ⊗ SS_{b_t}
    BS_{p_i} ← BS_{p_i} ∪ BS_{r_i}
    unb_{p_i} ← true
    if cycle(r_i) = false then
        SS_{p_i} ← SS_{p_i} ∪ SS_{r_i}
        sup_{p_i} ← true
```

Example 6 (continued). In the MCS depicted in Figure 5.1, $P2P\_DR_{CS}$ returns positive (*true*) answers for a_3, a_4, b_5 and b_6 along with empty Supportive and Blocking Sets, as all of them are local conclusions of C_3, C_4, C_5 and C_6 respectively. Called by C_2, *Support* computes $SS_{r^m_{23}} = \{\{b_5\}\}$ and $SS_{r^m_{24}} = \{\{b_6\}\}$, and $P2P\_DR_{CS}$ returns *true* as an answer for b_2 and a_2 and $SS_{a_2} = SS_{b_2} = \{\{b_5\},\{b_6\}\}$. For r^m_{12} and r^m_{13}, *Support* respectively computes $SS_{r^m_{12}} = \{\{a_2, b_5\},\{a_2, b_6\}\}$ and $SS_{r^m_{13}} = \{\{a_3, a_4\}\}$, and $SS_{a_1} = SS_{r^m_{12}}$, $BS_{\neg a_1} = SS_{r^m_{13}}$. According to $T_1 = [C_4, C_2, C_6, C_3, C_5]$, C_6 and C_2 both precede C_3 in T_1, and $A = \{a_2, b_6\} \in SS_{a_1}$ is computed to be *stronger* than $B = \{a_3, a_4\}$, which is the only set in $BS_{\neg a_1}$, and therefore $P2P\_DR_{CS}$ returns *true* as answer for a_1, and eventually $Ans_{x_1} = true$. □

Example 1 (continued). In the system described in Example 1 (Figure 1.1), suppose that there is an additional person detection service, C_6, and its local knowledge is encoded in rule r^l_{61}, which states that it has detected one person it the classroom:

$$r^l_{61} :\rightarrow persons(1)_6$$

Suppose also that C_4 (the classroom manager) uses an additional mapping rule, r^m_{43}, which states that if the projector is off, and it receives information from C_6 that there is only one person in the classroom, then there is no class activity.

$r^m_{43} : projector(off)_4, persons(1)_6 \Rightarrow \neg class\_activity_4$

Assume also that the preference ordering defined by C_1 and C_4 are, respectively, $T_1 = [C_4, C_3, C_6, C_2, C_5]$ and $T_4 = [C_4, C_3, C_5, C_2, C_6]$. *P2P_DR*$_{PS}$, the algorithm that implements *Propagating Mapping Sets*, will compute $Ans_{\neg class\_activity_4} = true$ and $SS_{\neg class\_activity_4} = \{detected(1)_5\}$, since C_5 precedes C_6 in T_4. *Support* will compute $SS_{r^m_{15}} = \{classtime_2, location\_RA201_3\}$ and $SS_{r^m_{16}} = \{\neg class\_activity_4, detected(1)_5\}$. Since C_5 does not precede neither C_3 nor C_4 in T_1, $SS_{lecture_1} = SS_{r^m_{15}}$ will be computed to be *stronger* than $BS_{\neg lecture_1} = SS_{r^m_{16}}$, and *P2P_DR*$_{PS}$ will return *false* as answer for $\neg lecture_1$, and eventually $Ans_{ring_1} = false$.

On the other hand, *P2P_DR*$_{CS}$ will also return $Ans_{\neg class\_activity_4} = true$, but the Supportive Set of $\neg class\_activity_4$ is in this case $SS_{\neg class\_activity_4}$=$\{\{detected(1)_5\}, \{persons(1)_6\}\}$. For the two conflicting rules, r^m_{15} and r^m_{16}, *Support* respectively computes $SS_{r^m_{15}} = \{\{classtime_2, location\_RA201_3\}\}$ and $SS_{r^m_{16}} = \{\{\neg class\_activity_4, detected(1)_5\}, \{\neg class\_activity_4, persons(1)_6\}\}$. According to T_1, C_6 and C_4 both precede C_2 in T_1, and $A = \{\neg class\_activity_4, persons(1)_6\} \in SS_{\neg lecture_1}$ is computed to be *stronger* than $B = \{classtime_2, location\_RA201_3\}$, which is the only set in $BS_{lecture_1}$. Eventually, *P2P_DR*$_{CS}$ returns *true* as answer for $\neg lecture_1$, and $Ans_{ring_1} = true$. □

5.3.2 Complexity Analysis

The main difference of *Complex Mapping Sets* with the first three strategies is that the Supportive / Blocking Sets used in this case are actually sets of sets of literals, with each different set representing a different way to prove a literal. As a result, each Supportive / Blocking Set may contain a number of sets, which is in the worst case equal to the total number of different combinations of the literals defined in the system. As a result, the complexity of comparing two Supportive / Blocking Sets is in this case $O(n^n)$, where n is the total number of literals defined in the system, making the overall complexity of the algorithms exponential to the size of the knowledge base.

Proposition 10 *The number of operations imposed by one call of P2P_DR for the evaluation of a query for literal p_i is in the worst case that all rules of C_i contain either p_i or $\sim p_i$ in their head, and each different combination of system literals can be used to derive either p_i or $\sim p_i$, $O(n^n)$, where n is the total number of literals defined in the system.*

$P2P\_DR_{CS}$ can also be optimized using structures of the same form with those used by $P2P\_DR_{PS}$, which implements *Propagating Mapping Sets*. Each record of INC_Q or OUR_Q is of the form: $rec(p_i, Hist_{p_i}) : (Ans_{p_i}, BS_{p_i}, SS_{p_i})$, and the only difference with the respective structures used by $P2P\_DR_{PS}$ is in the form of BS_{p_i} and SS_{p_i}.

Using these optimizations, the number of algorithm calls and messages is reduced in the same way with the *Single Answers* strategy, to $O(n \times \sum P(n,k))$ and $O(2 \times n \times \sum P(n,k))$ respectively for the general case, and $O(c \times n)$ and $O(2 \times c \times n)$ respectively for acyclic MCS, where c stands for the total number of contexts and n for the total number of literals in the system.

Chapter 6

Implementation & Evaluation

This chapter describes two different implementations of the four strategies for conflict resolution - the *Single Answers* strategy presented in Chapter 4, and the three alternative strategies (*Strict-Weak Answers*, *Propagating Mapping Sets* and *Complex Mapping Sets*) presented in Chapter 5. In the first implementation, we used a simulated peer-to-peer environment in order to evaluate and compare the four strategies in terms of computational complexity. In the second one, the four strategies are implemented in Logic Programming. The aim of the second implementation is to highlight the relation between our rule-based approach and Logic Programming, and to enable reasoning in real ambient environments using lightweight Prolog machines running on a variety of stationary and mobile ambient devices.

6.1 Simulation-driven evaluation

6.1.1 Simulation Environment

In order to evaluate the four strategies, we implemented the respective versions of $P2P\_DR$, and a P2P system simulating the proposed Multi-Context framework in Java. The main reasons for choosing this particular programming language are

1. Java contains several data structures that can be used easily and efficiently.

2. It is a *"write-once, use many"* language, thus giving us the opportunity to use the peer-to-peer system virtually anywhere a virtual machine can be installed,

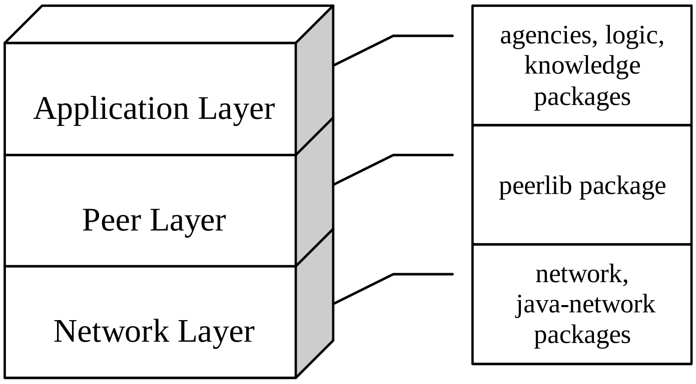

Figure 6.1: System Layered Architecture.

from personal computers to mobile phones. This adds an extra advantage when creating applications that involve multiple types of devices.

Software Architecture

For the network library as well as the peer-to-peer communication library, we used a custom-built library based on the java.network packages. Libraries such as *JXTA* [58] would be inefficient due to the complexity in configuring such a simple ad-hoc peer-to-peer network. The message exchanging protocol in our custom library is also simple and straightforward. However, one can use any other peer communication libraries, as the system uses an abstract network manager interface.

The system is composed of 5 packages: *agencies*, *logic*, *knowledge*, *network*, *peerlib*. The *agencies* package contains the classes that implement the text file parsers as well as those that implement the four versions of *P2P_DR*. The *logic* package contains the classes the represent (in memory) the literals and the rules. The *knowledge* package includes the *KnowledgeBase* class, which stores the local and mapping rules, the preference ordering and any other required information, and some cache classes. The *network* package includes the mechanism that associates a new socket connection with a new thread, whereas the *peerlib* contains the higher-level classes that operate the communication between two peers.

Figure 6.1 depicts the architecture, organized in a protocol stack manner. The main class that operates the peer instance is called *Node*. Its functionality includes parsing the preference ordering and theory files, as well as the initialization of the network libraries and knowledge base. When initialization is complete, it waits for pending queries. Finally, another class named *Client* can be used to connect to a *Node* specified by IP address, so that one can manually make specific queries to that specific peer instance.

Network Formation

When receiving an incoming query, a system node firstly uses its local knowledge to compute the answer. In case it fails to answer based on its local theory, the node attempts to use its mappings. In the latter case, it invokes other peers in the network, by sending them queries about the truth value of some of their local literals. When issuing a query about a given literal, a node determines which peer to ask by checking the name of the literal, which associates the literal with the system node vocabulary it belongs to. One query may result in a sequence of queries spread over the network. The response path of this sequence will eventually follow the exact opposite path of the query sequence. The response message will include an answer indicating the truth value of the literal that the initial query was about, and depending on the strategy additional relevant information; namely (a) in the case of *Strict-Weak Answers*, a sign that indicates whether this is a strict or weak answer; (b) in the case of *Propagating Mapping Sets*, a set of literals that corresponds to a reasoning chain that leads to the computed answer; and (c) in the case of *Complex Mapping Sets*, a set of sets of literals, each of which corresponds to a reasoning chain that leads to the computed answer. Figure 6.2 depicts the information flow in such a hypothetical network.

6.1.2 Experimental Evaluation

The goal of the experiments was to compare the four different strategies in terms of actual computational time spent by a system peer to evaluate the answer to a single query, and to test their scalability. Below we present the test theories that we used and the setup of the experiments, and discuss the results of the evaluation of the four strategies using systems with various peer populations.

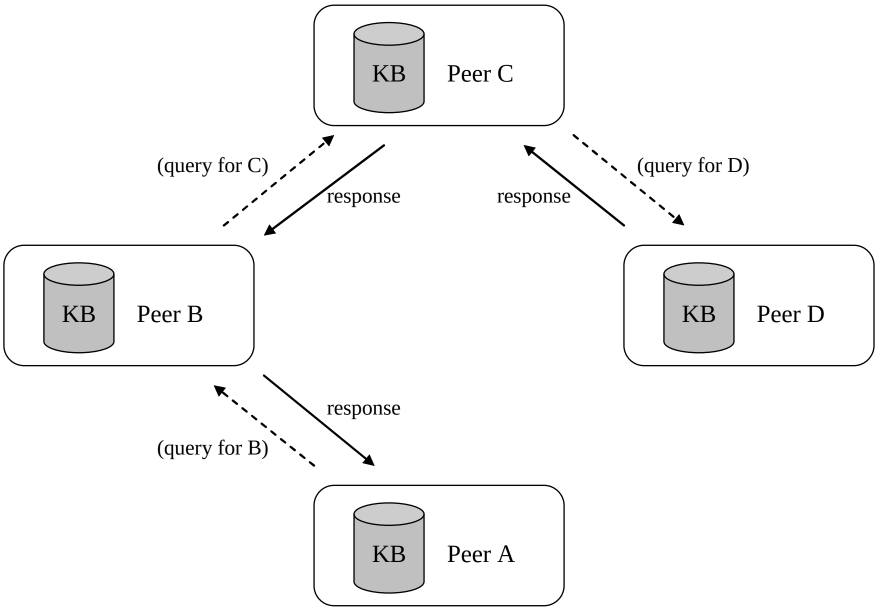

Figure 6.2: Network Formation.

Setup of the Experiments

Using a tool that we built for the needs of the experiments, we created theories that correspond to the case that the evaluation of a single query requires the evaluation of the truth values of all literals from all system nodes. For sake of simplicity, we did not include the case of loops in the global knowledge base; hence, for each literal the returned answer was either *true* or *false* The test theories have the following form:

$r_1^m : a_2, a_3, ..., a_n \Rightarrow a_0$
$r_2^m : a_1, a_3, ..., a_n \Rightarrow a_0$
...
$r_{n/2}^m : a_1, ..., a_{n/2-1}, a_{n/2+1}, ..., a_n \Rightarrow a_0$
$r_{n/2+1}^m : a_1, ..., a_{n/2}, a_{n/2+2}, ..., a_n \Rightarrow \neg a_0$
...
$r_n^m : a_1, a_2, ..., a_{n-1} \Rightarrow \neg a_0$

The above mapping rules are defined by C_0 and associate the truth value of its local literal a_0 with the truth values of the literals from n other system peers. Half of them support a_0 as their conclusion, while the remaining rules contradict a_0 ($\neg a_0$ is in their head). In case the answers returned for all foreign literals a_1, a_2,...,a_n are all *true*, then all mapping rules are applicable and are involved in the computation of the truth value of a_0.

Number of Messages

As it has already been proved, the number of messages that are required for the computation of a single query is the same for all alternative strategies. Specifically, for the case that we describe above, given a query about a_o and using the optimized algorithms for query evaluation ($P2P\_DR^O$), C_0 will make one query for each of the foreign literals that appear in the body of its mapping rules, sending in total n *query messages*, while it will receive one response for each of the query messages. In total n *response messages* will be received regardless of the followed strategy. In the same sense, assuming that all system peers use theories of the same form, each of the peers that receive *query messages* from C_0 has to make (in the worst case) one query for each of the foreign literals that appear in the body of its mapping rules (n query messages) and will receive an equal number of *query responses.* Hence, totally, in the worst case that $P2P\_DR^O$ uses all mapping rules from all system peer theories, the total number of messages that need to be exchanged for the evaluation of the query about a_0 are $2n^2$.

Size of Messages

A major difference between the four versions of $P2P\_DR$ that implement respectively the four alternative strategies for conflict resolution, is in the size of messages exchanged between the system contexts. Specifically, the size of the query messages is the same for all strategies. Each such message will contain the queried literal, the ids of the querying and the queried contexts and a set of literals representing the *history* of the query. However, the size and form of the query responses largely depend on the conflict resolution strategy. Specifically, for the type of experiments that we conducted, a response message for a literal a_i has one of the following forms:

- In the case of $P2P\_DR$, which implements the *Single Answers* strategy, the response message will contain only the truth value of a_i; namely, one of the values *true* and *false* (since there are no loops in the global knowledge base).

Table 6.1: Size of Response Messages for the Four Strategies

SA	SWA	PS	CS
1	1	$2\times(n-1)+1$	$2\times(n-1)\times(n-2)+1$

- In the case of $P2P\_DR_{SWA}$, which implements the *Strict-Weak Answers* strategy, the response message will contain one of the values *str(true)*, *weak(true)* and *false*.
- In the case of $P2P\_DR_{PS}$, which implements the *Propagating Mapping Sets* strategy, the response message will contain the truth value of a_i (either *true* or *false*) and two sets of literals - SS_{a_i} representing the Supportive Set of a_i and BS_{a_i} representing the Blocking Set of a_i. By construction of the test theories, each of the two sets will contain one literal from each of the peers (except C_i) in the system. Therefore, the size of a response message is in this case $2 \times (n-1) + 1$, where n is the total number of peers in the system.
- In the case of $P2P\_DR_{CS}$, which implements the *Complex Mapping Sets* strategy, the response message will contain the truth value of a_i (either *true* or *false*) and two sets of literals - SS_{a_i} representing the Supportive Set of a_i and BS_{a_i} representing the Blocking Set of a_i. By construction of the test theories, each of the two sets will contain $n-1$ different sets of literals, and each of these sets will contain $n-2$ literals. Therefore, the total size of each response message will be $2 \times (n-1) \times (n-2) + 1$, where n is the total number of peers in the system.

The results about the size of *response messages* are summarized in Table 6.1.

Processing Time

In order to exclude the communication overhead from the total time spent by C_0 to evaluate the truth value of a_0, we filled a local cache class with appropriate answers for all the foreign literals. Specifically, for each version of $P2P\_DR$, this class is filled with answers for all foreign literals $(a_1, a_2, ..., a_n)$ as follows:

1. $P2P\_DR$: Positive truth values (*true*) for all literals

Table 6.2: Processing Time for the Four Strategies

♯ literals (n)	SA	SWA	PS	CS
10	78	80	1313	2532
20	469	540	1534	4305
40	2422	3102	3466	207828
60	5719	6390	7188	-
80	10437	10302	15484	-
100	16484	15550	27484	-

2. $P2P\_DR^{SWA}$: Positive strict/weak answers (chosen randomly) for all literals.

3. $P2P\_DR^{PS}$: Positive truth values with Supportive Sets that contain all other foreign literals:

$$SS_{a_i} = \{a_1, a_2, ..., a_{i-1}, a_{a+1}, ..., a_n\}$$

4. $P2P\_DR^{PS}$: Positive truth values with Supportive Sets of the form:

$$SS_{a_i} = \{\{a_2, ..., a_{i-1}, a_{a+1}, ..., a_n\}, \{a_1, a_3, ..., a_{i-1}, a_{a+1}, ..., a_n\},$$

$$..., \{a_1, a_2, ..., a_{i-1}, a_{a+1}, ..., a_{n-1}\}$$

For each version of the algorithm, we conducted six experiments with a variant size of the global knowledge base in terms of total number of literals, which in this case coincides with the total number of system peers: 10, 20, 40, 60, 80, and 100. The test machine was an Intel Celeron M at 1.4 GHz with 512 MB of RAM.

Table 6.2 shows in milliseconds the processing time for each version of *P2P_DR* (*SA* refers to *P2P_DR*, which implements the *Single Answers* strategy, *SWA* refers to $P2P\_DR_{SWA}$, which implements *Strict-Weak Answers*, *PS* stands for $P2P\_DR_{PS}$, which implements *Propagating Mapping Sets*, while *CS* refers to $P2P\_DR_{CS}$, which implements *Complex Mapping Sets*). For the case of $P2P\_DR^{CS}$, we were able to measure the computation time only for the cases where $n = 10, 20, 40$; in the other cases the test machine ran out of memory.

As it is obvious from Table 6.2, the results for the first three strategies are similar; the computation time is proportional to the square of the number of system peers, verifying our expectations from the theoretical results presented in the previous chapters. The *Complex Sets* strategy requires much more memory space and computation time (exponential to the number of peers), which make it inapplicable in cases of very dense systems. The results also verify the tradeoff between the computational complexity and the extent of context information that each algorithm exploits to evaluate the quality of the imported context information.

6.2 The Algorithms in Logic Programming

In this section we present the logic metaprograms that implement the four alternative strategies for conflict resolution. They are driven by the logic metaprogram of Defeasible Logic, as the latter is described in [9]. We should note that for the negation operator that we use in the metaprograms we have adopted the Well-Founded Semantics.

Overall, the goal of the translation of the context theories C_i in a Multi-Context System C into a logic program $P(C)$ is to show that in case there are no loops in C, then for a literal $p \in C$:

p is justified in C iff
p is included in the Well-Founded model of $P(C)$

To achieve this goal, for each of the context C_i in C, we add a fact of the form $context(c_i)$. For each of the four strategies, we use the respective logic metaprogram, and for each context theory C_i, we use facts representing the elements of the theory as follows:

- For each strict local rule in C_i, $r_i^l : a_i^1, a_i^2, ...a_i^{n-1} \rightarrow a_i^n$, we add a fact of the form: $strict(r_i, C_i, lit(a^n, C_i), [lit(a^1, C_i), ..., lit(a^{n-1}, C_i)])$.
- For each defeasible local rule in C_i, $r_i^d : a_i^1, a_i^2, ...a_i^{n-1} \Rightarrow a_i^n$, we add a fact of the form: $defeasible(r_i, C_i, lit(a^n, C_i), [lit(a^1, C_i), ..., lit(a^{n-1}, C_i)])$.
- For each mapping rule in C_i, $r_i^m : a_i^1, a_j^2, ...a_k^{n-1} \Rightarrow a_i^n$, we add a fact of the form: $mapping(r_i, C_i, lit(a^n, C_i), [lit(a^1, C_i), ..., lit(a^{n-1}, C_k)])$.

- The preference ordering of context C_i, $T_i = [C_k, C_l, ..., C_n]$ is added as a fact of the form $pref(C_i, [C_k, C_l, ..., C_n])$.

6.2.1 Single Answers metaprogram

The first three clauses of the metaprogram that implements the *Single Answers* strategy define the classes of rules used in a context theory.

```
c1: supportive_rule(Name,Context,Head,Body):-
strict(Name,Context,Head,Body).

c2: supportive_rule(Name,Context,Head,Body):-
defeasible(Name,Context,Head,Body).

c3: supportive_rule(Name,Context,Head,Body):-
mapping(Name,Context,Head,Body).
```

The following clauses define local provability: a literal is locally provable in context C if it is in the head of a strict local rule in C, the premises of which are locally provable in C.

```
c4: locally(X,C):- strict(R,C,X,L), locally_provable(L,C).

c5: locally_provable([],C):- context(C).

c6: locally_provable([X1|X2],C):- locally(X1,C),
locally_provable(X2,C).
```

The next clauses define provability of literals. Specifically, a literal $lit(X, C)$ is provable in context C in two cases:

1. If it is locally provable in C. In this case an empty set is assigned as the Supportive Set of the literal (*c7*).

2. If it is in the head of an applicable rule in C, which is not blocked, and the negation of the literal is not locally provable in C. In this case, the Supportive Set of this rule (SS_r) is assigned as the Supportive Set of the literal (*c8*).

```
c7: provable(X,C,[]):- locally(X,C).

c8: provable(X,C,SSr):- applicable_rule(R,C,X,L,SSr),
not(locally(~X,C)), not(blocked(R,C,X,SSr)).
```

We use clause *c9* to denote that if a literal $lit(X, K)$ is provable in a context K in one of the two ways described above, then we can also consider it provable in any other context C. In this case $[K]$ is assigned as the Supportive Set of $lit(X, K)$ in C.

```
c9: provable(lit(X,K),C,[K]):- provable(lit(X,K),K,SSx), K\=C.
```

The next clause defines applicable rules. A supportive rule is applicable in C if all its premises are provable. The Supportive Set of the rule (SS_r) is the union of the Supportive Sets in C of the literals in the body of the rule.

```
c10: applicable_rule(R,C,X,L,SSr):- supportive_rule(R,C,X,L),
provable_list(L,C,SSr).
```

The next clauses denote that a list of literals is provable in C if all the members of the list are provable in C, and that the Supportive Set of the list is the union of the Supportive Sets of the elements of the list. In clause $c14$, $merge(SS1, SS2, SSL)$ creates SSL as the union of sets $SS1$ and $SS2$.

```
c11: provable_list([],C,[]):- context(C).

c12: provable_list([X1|X2],C,SSL):- provable(X1,C,SS1),
provable_list(X2,C,SS2), merge(SS1,SS2,SSL).
```

Clause $c13$ defines when a rule is blocked. A rule R is blocked in C when there is an applicable rule S with a contradictory conclusion in C, such that R is not stronger than C according to the preference ordering defined by C.

```
c13: blocked(R,C,X,SSr):- applicable_rule(S,C,~X,L,SSs), pref(C,T),
not(stronger(SSr,SSs,T,SSr)).
```

Finally the last clauses define how the strongest between two rule Supportive Sets (A,B) is determined based on a preference ordering T.

```
stronger(A,B,T,B):- weakest(A,A1,T), weakest(B,B1,T),
weaker(A1,B1,T,A1).

stronger(A,B,T,A):- weakest(A,A1,T), weakest(B,B1,T),
weaker(A1,B1,T,B1).

weakest([X],X,_).
```

```
weakest([X|Tail],M,T):- weakest(Tail,M1,T),
weaker(X,M1,T,M).

weaker(Y1,Y2,T,Y1):- ith(Pos1,T,Y1),ith(Pos2,T,Y2),Pos1>Pos2.
weaker(Y1,Y2,T,Y2):- ith(Pos1,T,Y1),ith(Pos2,T,Y2),Pos2>Pos1.
```

6.2.2 Strict-Weak Answers metaprogram

In order to support the features of *Strict-Weak Answers* strategy, the *Single Answers* metaprogram is modified as follows:

To support the two different types of answers (*strict* and *weak* answers), clause $c9$ is replaced with $c9a$ and $c9b$. In clause $c9a$, $loc(K)$ indicates that $lit(X,K)$ is locally provable in K, while in clause $c9b$, $map(K)$ indicates that $lit(X,K)$ is provable in K, but not locally provable.

```
c9a: provable(lit(X,K),C,[loc(K)]):- locally(lit(X,K),K),K\=C.

c9b: provable(lit(X,K),C,[map(K)]):- provable(lit(X,K),K),K\=C,
not(locally(lit(X,K),K)).
```

The strength of an element of a Supportive Set ($loc(K)/map(K)$) in context C is determined primarily by the type of answer described in the element (loc/map), and secondly by the rank of K in the preference ordering of C. To support this feature the clauses that define *weaker* are modified as follows:

```
weaker(loc(Y1),map(Y2),_,map(Y2)).
weaker(map(Y1),loc(Y2),_,map(Y1)).

weaker(loc(Y1),loc(Y2),T,loc(Y1)):- ith(Pos1,T,Y1), ith(Pos2,T,Y2),
Pos1>Pos2.
weaker(loc(Y1),loc(Y2),T,loc(Y2)):- ith(Pos1,T,Y1),
ith(Pos2,T,Y2), Pos2>Pos1.

weaker(map(Y1),map(Y2),T,map(Y1)):- ith(Pos1,T,Y1), ith(Pos2,T,Y2),
Pos1>Pos2.
weaker(map(Y1),map(Y2),T,map(Y2)):- ith(Pos1,T,Y1),
ith(Pos2,T,Y2), Pos2>Pos1.
```

6.2.3 Propagating Mapping Sets metaprogram

To implement the differences between *Propagating Supportive Sets* and *Single Answers*, the *Single Answers* metaprogram is modified as follows:
Clause $c8$, which describes the second way in which a literal is provable in a context C is replaced with $c8a$.

```
c8a: provable(X,C,SSx):- canprove(X,C), supportive_set(X,C,SSx).
```

We add clause $c14$ to describe when a literal can be proved in C. This is the same with the definition of *provable* in *Single Answers* with the difference that *canprove* carries no information about the Supportive Set of the literal.

```
c14: canprove(X,C):- applicable_rule(R,C,X,L,SSr),
not(locally(~X,C)), not(blocked(R,C,X,SSr)).
```

Two clauses are added that define the Supportive Set of a literal (SS_x) in context C. Clause $c15$ states that in case X is locally provable in C, its Supportive Set is an empty set, while clause $c16$ states that in any other case the *strongest* (according to the preference order of C) Supportive Set of the applicable supportive rules with head X is assigned as the Supportive Set of X in C. In clause $c16$, $findall$ creates SS as the union of the Supportive Sets (SSr) of all applicable rules with head X.

```
c15: supportive_set(X,C,[]):- locally(X,C).
```

```
c16: supportive_set(X,C,SSx):- not(locally(X,C)),
findall(SSr,applicable_rule(R,C,X,L,SSr),SS), pref(C,T),
strongest_set(SS,SSx,T).
```

Clause $c9$ is replaced with clause $c9c$, which states that if a literal $lit(X, K)$ is provable in a context K then we can also consider it provable in any other context C, and the union of $[K]$ and SSx (the Supportive Set of $lit(X, K)$ in K) is assigned as the Supportive Set of $lit(X, K)$ in C.

```
c9c: provable(lit(X,K),C,[K|SSx]):- provable(lit(X,K),K,SSx), K\=C.
```

Finally, the following two clauses are used to compute the strongest between a number of Supportive Sets.

```
strongest_set([X],X,_).
strongest_set([X|Tail],M,T):-strongest_set(Tail,M1,T),
stronger(X,M1,T,M).
```

6.2.4 Complex Mapping Sets metaprogram

To support the features of *Complex Mapping Sets*, the *Single Answers* metaprogram is modified as follows: Clause $c7$ is replaced with $c7a$, because Supportive Sets in this case are actually sets of sets.

```
c7a: provable(X,C,[[]]):- locally(X,C).
```

Similarly with Propagating Mapping Sets clause $c8$ is replaced with $c8a$, and we add clause $c14$.

```
c9a: provable(X,C,SSx):- canprove(X,C), supportive_set(X,C,SSx).

c16: canprove(X,C):- applicable_rule(R,C,X,L,SSr),
not(locally(~X,C)), not(blocked(R,C,X,SSr)).
```

We add the following two clauses ($c15a$ and $c16a$) to define the Supportive Set of literal. In this case, the Supportive Set of a literal in C, which is not locally provable in C, is defined as the union of the Supportive Sets of the applicable supportive rules in C.

```
c15a: supportive_set(X,C,[[]]):- locally(X,C).

c16a: supportive_set(X,C,SSx):- not(locally(X,C)),
findall(SSr,applicable_rule(R,C,X,L,SSr),SSx).
```

Clause $c9$ is replaced with clause $c9d$, which states that if a literal $lit(X, K)$ is provable in a context K then we can also consider it provable in any other context C, and the *union product* of $[[K]]$ and SSx (the Supportive Set of $lit(X, K)$ in K) is assigned as the Supportive Set of $lit(X, K)$ in C. *uproduct* implements the *union product* operator that we defined in Chapter 5.

```
c9d: provable(lit(X,K),C,SSxc):- provable(lit(X,K),K,SSx), K\=C,
uproduct([SSx,[[K]]],SSxc).
```

Clause $c12$ is replaced with $c12a$, which states that the Supportive Set of the union of two sets of literals $X1$ and $X2$ is the *union product* of their Supportive Sets ($SS1$ and $SS2$ respectively).

```
c12a: provable_list([X1|X2],C,SSL):- provable(X1,C,SS1),
provable_list(X2,C,SS2), uproduct([SS1,SS2],SSL).
```

Finally, the *stronger* predicate in clause $c13$ is replaced by *complex_stronger*, which defines the *stronger* relation between two sets of sets.

```
complex_stronger(A,B,T,B):- strongest_set(A,A1,T),
strongest_set(B,B1,T), stronger(A1,B1,T,B1).

complex_stronger(A,B,T,A):- strongest_set(A,A1,T),
strongest_set(B,B1,T), stronger(A1,B1,T,A1).

strongest_set([X],X,_).
strongest_set([X|Tail],M,T):-strongest_set(Tail,M1,T),
stronger(X,M1,T,M).
```

Chapter 7

Conclusion

To conclude this thesis, we summarize and discuss its main contributions, and propose possible directions for future research.

7.1 Synopsis

The imperfect nature of context knowledge and the special characteristics of ambient devices and Ambient Intelligence environments have introduced new challenges in the field of Distributed Artificial Intelligence. Most current Ambient Intelligence systems have not successfully addressed most of them, by relying on unrealistic simplifying assumptions, such as perfect knowledge of context, centralized context, and unbounded computational and communicating capabilities. The requirements, though, are much different in such environments. The uncertainty of context and its distribution to heterogeneous devices with restricted capabilities, impose the need for relaxing these assumptions and for employing different reasoning approaches.

This thesis describes a formal model for representing and reasoning with the imperfect and distributed context knowledge in Ambient Intelligence environments. The proposed representation model is based on Multi-Context Systems; a formalism in which the notions of distribution of the available knowledge, and interrelation between knowledge possessed by different ambient agents are naturally represented through *contexts* and *mappings* between contexts. In order to handle cases of uncertain, missing or ambiguous context, we extended the basic model of MCS, as this was introduced in [57], with new features, such as defeasible mapping rules and a preference relation over the system contexts. On top of this model, we developed an argumentation framework, which enables distributed reasoning with the available context and preference information. The proposed framework extends the argumentation semantics of Defeasible Logic, proposed in [61], which in turn is based on the grounded semantics of Dung's abstract argumentation framework [45]. Specifically, it introduces the notions of *rank* of an argument, which is determined according to the available preference information, and of *argumentation line*, which accounts for the fact that arguments are interrelated through the mappings defined by the system contexts.

In chapter 4, we described an operational model in the form of a distributed algorithm for query evaluation. We studied the formal properties of the algorithm with respect to *termination*, *number of messages*, and *computational complexity*. We also proved that the algorithm is sound and complete with respect to the argumentation framework, and that there is a standard process that unifies the distributed context theories in a global theory of Defeasible Logic, which produces the same results with the algorithm under the proof theory of Defeasible Logic [8]. The latter result enables resorting to centralized reasoning by collecting the distributed context theories in a central entity and creating an equivalent defeasible theory.

In chapter 5, we described three alternative strategies for conflict resolution, which differ in the extent and type of context information that is exploited to resolve conflicts caused by the interaction of contexts through their mappings. Specifically, the alternative strategies evaluate the imported knowledge taking into account not only the preference rank of the *source* of the imported knowledge, but also how the *source* derived this knowledge. In *Strict-Weak Answers*, we define two types of derivation: local derivation, which is based on the strict local rules of the *source*'s context theory, and distributed derivation, which also uses the *source*'s mappings. In the other two strategies (*Propagating Mapping Sets* and *Complex Mapping Sets*), the source, in case of *distributed derivation*, returns also information about which other contexts are involved in this derivation. We also discussed the tradeoff between the extent of context knowledge that is exploited to conduct conflict resolution, and the computational overhead imposed to the context that resolves the conflict.

Finally, in chapter 6, we presented two different implementation of the four strategies. The first one aimed at evaluating the four strategies in terms of computational complexity, while the second one is based on the translation of context theories into logic programs, and on the use of four different metaprograms, each of which implements one of the four strategies.

7.2 Future Directions

Our study on defeasible contextual reasoning can be extended in various dimensions, which are discussed below.

7.2.1 Extending our approach to multiple dimensions

In the introductory chapter, we already referred to a number of assumptions that our reasoning methods depend on. It is among our plans to relax some of these assumptions in order to generalize our reasoning methods, and enable their applicability in a greater range of applications. Below, we discuss some main directions to which our approach can be extended.

Overlapping Vocabularies

One of the main assumptions that we make is that each different agent (*context*) uses a distinct vocabulary to represent its context knowledge. However, this is not always the case in real environments. There are some types of *words*, such as URIs, which may be commonly used by different agents, and this fact is in contrast with our assumption. Without altering our approach, we can overcome this problem by adding a context identifier e.g. as a prefix in each such word, add the modified words in the vocabularies of the contexts, and use appropriate mappings to associate them. E.g. assume that uri is a word that both C_1 and C_2 wish to use in their local theories. C_1 and C_2 may add $c_1 : uri$ and $c_2 : uri$ respectively in their vocabularies, while the following mapping rules should be added in C_1 and C_2 respectively:

$$r_1^m : c_2 : uri \Rightarrow c_1 : uri$$
$$r_2^m : c_1 : uri \Rightarrow c_2 : uri$$

These two mapping rules express the equivalence between the two words, $c_1 : uri$ and $c_1 : uri$.

Although this method does not provide a convenient solution, it does not require any changes to our representation and reasoning models. An alternative more convenient solution, which we plan to study in the future, would be to modify our models, so that contexts may use overlapping vocabularies, with the overlapping parts containing such types of words.

Study cases of reduced complexity

In Chapters 4 and 5, we proved that the worst case complexity (in terms of number of messages) of the distributed algorithms for query evaluation that we propose is between 2^n and $n!2^n$, where n stands for the total number of system literals, while in acyclic MCS the number of messages is proportional to n and to the total number of system contexts. It is, however, also very interesting to define the *average case* and study the complexity of the algorithms in that case. It is also interesting to define cases that are more common in practice, based on real-world scenarios of Ambient Intelligence, and study how much better our methods can perform in such situations.

Relevance with loop checking variants of Defeasible Logic

In Chapter 4, we studied a method for building a global defeasible theory using the distributed context theories and their preference orderings. Through Propositions 7 and 8, we showed that in case there are no loops in the global knowledge base, the global theory produces the same results as the application of $P2P\_DR$ on a Multi-Context System C under the proof theory of the ambiguity blocking version of Defeasible Logic with superiority relation [8].

In Chapter 6, we also described how are reasoning methods are implemented in Logic Programming, driven by the translation of Defeasible Logic in Logic Programming, as the latter is described in [9]. We argued that the logic programs produce equivalent results with our reasoning algorithms, in case there are no loops in the global knowledge base.

It is among our future plans to extend these results for the case of *non-acyclic* MCS; namely for systems with loops in the global knowledge base. Specifically, we plan to study the relation between our reasoning approach and loop checking variants of Defeasible Logic, such as those described by Nute in [88; 89]. For the translation into logic programs, we will extend the translation schemes proposed by Maier and Nute in [82; 83].

Integration with Abstract Argumentation Frameworks

As we already stated in Chapters 2 and 3, the argumentation semantics that we presented in Chapter 3 extends the argumentation framework of Defeasible Logics proposed by Governatori et al. in [61], with the notions of *argumentation lines* and *ranks of arguments*, which are derived using the preference orderings defined by the system contexts.

Regarding the recent prominent studies on preference-based argumentation frameworks that we discuss in Chapter 2, we argued that our approach is also closely related to the abstract argumentation framework with contextual preferences of Amgoud *et al.* [7], where each context also defines its own preference ordering on the set of arguments. An interesting extension of our approach would be to integrate the concepts that we use and our reasoning methods in this argumentation framework, as well as in other similar frameworks proposed by Amgoud and her colleagues [4; 5; 6], which use a partial preordering on arguments, or in the value-based argumentation frameworks [14; 72], which relate the preference over an argument with the preference over the *value* it promotes.

7.2.2 Extending Contextual Default Logic with Priorities

One of the main limitations of Contextual Default Logic [36] regarding its applicability to Ambient Intelligence is that it does not include the notion of preference between contexts. This is, however, an important context parameter, which can be used to encode the confidence that an ambient agent has in the knowledge imported by other agents, to evaluate the quality of imported information, and to resolve inconsistencies that arise when importing mutually inconsistent information.

One of the next steps of this work is to integrate such a preference relation in Contextual Default Logic and implement strategies similar to those that we propose in Chapter 5 for global conflicts resolution. This would require the use of versions of Default Logic that introduce a priority relation in either the object or the meta language. An extensive survey for such Logics is available at [43]. Two interesting examples of deriving priorities for Default Logic using an external preference relation, are described in [74] and [101]. The first approach takes as input a trust relation computed in a Web-based Social Network and creates priorities between defaults in *Prioritized Default*

Logic [13]. The limitation of this approach is that the trust relation is fixed and shared between all system peers. The second approach deals with the problem of multiple extensions that can be generated from a particular default theory using priorities that are based on a learned confidence function. An additional advantage, here, is that the priority relation is not static. The second approach has been applied in reasoning with partitioned default theories in Multi-Agent Systems with similar requirements with Ambient Intelligence systems.

7.2.3 Deployment in Real Ambient Intelligence Environments

The final goal of this study is to deploy the proposed reasoning methods in real Ambient Intelligence environments and implement application scenarios such as those described in Chapter 1. The first steps towards this direction have already been completed. The four algorithms described in Chapter 5 have already implemented in Java and Prolog, as we already described in Chapter 6.

For the deployment of the logic programs, which implement the four algorithms for query evaluation in various mobile devices, such as PDAs or mobile phones, we plan to use Prolog machines that are specifically designed for mobile phones, such as JIProlog (Java Internet Prolog [115]), which is is compliant with MIDP 1.0/2.0 or Symbian OS mobile phones. For the communication of the various ambient devices, we plan to use the IEEE 802.11 wireless network infrastructure of FO.R.T.H. The ambient environments will also include several sensory subsystems, such as the Collaborative Location Sensing system [49] which exploits the IEEE 802.11 wireless network infrastructure for positioning, and a multi-camera vision system supporting the development of wide-area exertainment applications [120].

This deployment is expected to raise several issues that we will need to handle. These challenges are inherent in ad-hoc systems in settings with wireless communications, and include peer detection, message exchange, peers joining or leaving the network during query evaluation, errors and delays in the communication and others.

Appendix A

Proofs

Lemma 1. The sequences of sets of arguments J_i^C and $R_i^C(T)$ are monotonically increasing.

Proof. We prove the Lemma by induction on i. The inductive base is trivial in both cases since $J_0^C = \emptyset$ and $R_0^C(T) = \emptyset$ and thus $J_0^C \subseteq J_1^C$ and $R_0^C(T) \subseteq R_1^C(T)$.

By definition strict local arguments are acceptable w.r.t. every set of arguments; thus they are in every J_i^C.

Let A be an argument in J_n^C and let B be an argument defeating A. By definition, B is undercut by J_{n-1}^C; namely for every argumentation line B_L with head B, there is a literal q and an argument D, such that D is supported by J_{n-1}^C and D defeats a proper subargument of B or an argument in $B_L - \{B\}$ at q. By inductive hypothesis $J_{n-1}^C \subseteq J_n^C$; hence D is also supported by J_n^C. Consequently, B is undercut by J_n^C. Since A is an argument in J_n^C, by definition A is supported by J_{n-1}^C, and by inductive hypothesis, A is also supported by J_n^C. Therefore A is acceptable w.r.t. J_n^C, and $A \in J_{n+1}^C$.

We consider now the sequence of rejected arguments. Let A be an argument is $R_n^C(T)$. By definition, A is not a strict local argument and one of the three following conditions hold: (a) A proper subargument of A, A' is in $R_{n-1}^C(T)$. By inductive hypothesis $R_{n-1}^C(T) \subseteq R_n^C(T)$; hence $A' \in R_n^C(T)$ and $A \in R_{n+1}^C(T)$; (b) for every argumentation line A_L with head A, a subargument A' of an argument in $A_L - \{A\}$ is in $R_{n-1}^C(T)$, and by inductive hypothesis $A' \in R_n^C(T) \Rightarrow A \in R_{n+1}^C(T)$; or (c) a proper subargument of A or an argument in $A_L - \{A\}$ is defeated by an argument supported by T. In this case $A \in R_i^C(T)$ for every i, and therefore $A \in R_{n+1}^C(T)$

Lemma 2. In a Multi-Context System C:

1. No argument is both justified and rejected.
2. No literal is both justified and rejected.

Proof Suppose that there is an argument that is both justified and rejected. Let n be the smallest index such that for some argument A, $A \in RArgs^C(JArgs^C)$ and $A \in J_n^C$. Since $A \in J_n^C$, it holds that either (a) A is a strict local argument; or (b) A is supported by J_{n-1}^C and every argument defeating A is undercut by J_{n-1}^C. Since $A \in RArgs^C(JArgs^C)$, (a) does not hold. Hence, there is an argumentation line A'_L with head A such that for every subargument of A or argument in $A'_L - \{A\}$, A', it holds that $A' \in J_{n-1}^C$, and by Lemma 1 $A' \in J_n^C$. By definition, every argument defeating A' is undercut by J_{n-1}^C.

Since $A \in RArgs^C(JArgs^C)$, it holds by definition that for every argumentation line A_L with head A either (c) there exists an argument B that is supported by $JArgs^C$ and defeats a subargument of A or an argument in $A_L - \{A\}$; or (d) a subargument of A or an argument in $A_L - \{A\}$ is in $RArgs^C(JArgs^C)$. However, we have already proved that $A' \in J_{n-1}^C$, and by supposition n is the smallest index such that for some argument A, $A \in RArgs^C(JArgs^C)$ and $A \in J_n^C$; therefore (d) does not hold.

By (b) and (c), there exists an argument B', such that B' defeats A', and is supported by $JArgs^C$ and undercut by J_{n-1}^C. Hence, for every argumentation line B_L with head B' there is an argument D that is supported by J_{n-1}^C and defeats an argument in $B_L - \{B'\}$ or a proper subargument of B'. By definition of supported, there is an argumentation line B'_L with head B' such that every argument defeating an argument in $B'_L - \{B\}$ or a proper subargument of B' is undercut by $JArgs^C$. Hence D is undercut by $JArgs^C$; namely, for every argumentation line D_L with head D there is an argument E that is supported by $JArgs^C$ and defeats an argument in $D_L - \{D\}$ or a proper subargument of D. Since D is supported by J_{n-1}^C, there is an argumentation line D'_L with head D s.t. for every subargument of D or argument in $D'_L - \{D\}$, D', $D' \in J_{n-1}^C$. However, since D is undercut by $JArgs^C$, D' is defeated by an argument E' supported by $JArgs^C$; therefore $D' \in RArgs^C(JArgs^C)$ and $D' \in J_{n-1}^C$, which contradicts the assumed minimality of n. Hence the original supposition is false, and no argument is both justified and rejected.

The second part follows easily from the first: if p is justified there is an argument A for p in $JArgs^C$. From the first part, $A \in Args^C - RArgs^C(JArgs^C)$. Thus if p is justified then it is not rejected.

Lemma 3. If the set of justified arguments of C, $JArgs^C$ contains two arguments with conflicting conclusions, then both are strict local arguments.

Proof. Let the two arguments be A and B. Suppose B is a strict local argument. Then, for A to be acceptable with respect to every S, A must also be a strict local argument (otherwise B would defeat A, and B cannot be undercut by S). Thus, by symmetry, either A and B are both strict local arguments, or they are both defeasible local or mapping arguments. Suppose that both are defeasible local or mapping arguments and B defeats A. Then A must be rejected because it is defeated by an argument supported by $JArgs_C$, and is justified by assumption. By Lemma 2, this is not possible. Similarly, if we assume that A defeats B, we will conclude that B is both justified and rejected. Therefore, the two arguments are strict local arguments.

Proposition 1. $P2P\_DR$ terminates in finite time returning one of the values *true*, *false* and *undefined* as an answer for the queried literal.

Proof. At each recursive call, $P2P\_DR$ makes at most two calls of $local\_alg$ (for p_i and $\sim p_i$), two calls of $Support$ (for p_i and $\sim p_i$) and two calls of $Stronger$.

$local\_alg$ checks the local answers for all literals in the bodies of all strict local rules with head p_i (or $\sim p_i$). By definition, all such rules are defined by context C_i, and are finite in number. Since V_i (the vocabulary used by C_i) is a finite set of literals, each local rule contains a finite set of literals in its body. Therefore, one call of $local\_alg$ induces a finite number of operations. Since, one type of such operations involves a recursive call of $local\_alg$, we also have to prove that the total number of recursive calls of $local\_alg$ is not indefinite. Assume that one call of $local\_alg$ induces indefinite recursive calls of $local\_alg$. Since we have assumed that there are no loops in a context theory, each such call would be for a different literal in V_i. However, by the fact that there is a finite number of literals in V_i, the total number of recursive calls of $local\_alg$ is bounded by the number of literals in V_i. Therefore, no call of $local\_alg$ can induce indefinite recursive calls of $local\_alg$. Consequently, $local\_alg$ terminates in finite time returning either $true$ or $false$ as a local answer for the queried literal.

Support checks the answers for all literals in the bodies of all rules with head p_i (or $\sim p_i$). By definition, all such rules are defined by context C_i, and are finite in number. Since each literal in the bodies of these rules is in the vocabulary V_j of a context $C_j \in C$, and by the facts that there is a finite number of contexts in C, and that each vocabulary is a finite set of literals, each such rule contains a finite set of literals in its body. Therefore, one call of *Support* induces a finite number of checking operations. Since, one type of such operations involves a recursive call of $P2P\_DR$, we also have to prove that the total number of recursive calls of $P2P\_DR$ is not indefinite. Assume that one call of $P2P\_DR$ induces through *Support* indefinite recursive calls of $P2P\_DR$ and *Support*. At each recursive call, the structure that keeps track of the *history* of the query ($Hist$) is augmented with a literal q_j, where q_j belongs to the vocabulary V_j of a context $C_j \in C$, and q_j is not already contained in $Hist$. As the total number of contexts in C is finite, and the vocabulary of each context is a finite set of literals, the total number of recursive calls of $P2P\_DR$ and *Support* is bounded by the total number of literals in $V = \bigcup V_i$. Therefore, no call of *Support* can induce indefinite recursive calls of $P2P\_DR$ and *Support*. Since, *Support* additionally induces at most two calls of $Stronger$, we also have to prove that $Stronger$ also terminates in finite time.

$Stronger$ requires checking the preference ranks of the contexts that have defined the literals contained in two Supportive/Blocking Sets. A Supportive/Blocking Set is a set of literals derived by contexts in C. Since, we have already proved that there is a finite number of literals defined in C, each such set contains a finite number of elements. Therefore, given two sets A and B, and a preference order T_i, $Stronger$ terminates in finite time, returning either A, B or *none*.

Consequently, since $local\_alg$, *Support* and $Stronger$ terminate in finite time, $P2P\_DR$ also terminates in finite time. By definition of the algorithm, it is trivial to verify, that one of the values *true*, *false* and *undefined* is returned as an answer for p_i upon termination.

Proposition 2. For a MCS C and a literal p_i in $C_i \in C$, $local\_alg$ returns:

1. $localAns_{p_i} = true$ iff there is a strict local argument for p_i in $JArgs^C$

2. $localAns_{p_i} = false$ iff there is no strict local argument for p_i in $JArgs^C$

Proof (1, $\Rightarrow$). We use induction on the number of calls of $local\_alg$ that are required to produce the answer for p_i.

Inductive Base. Suppose that $local\_alg$ returns $localAns_{p_i} = true$ in one call. This means that there is a local strict rule with head p_i in C_i, r_i, such that $body(r_i) = \emptyset$. Using r_i we can build a strict local argument for p_i.

Inductive Step. Suppose that $n+1$ calls of $local\_alg$ are required to compute $localAns_{p_i} = true$. This means that there is a strict local rule with head p_i (say r_i) such that $\forall a_i \in body(r_i)$, $local\_alg$ returns $localAns_{p_i} = true$ in n or less calls. By inductive hypothesis, for every a_i there is a strict local argument for a_i in $Args_C$. Using the arguments for a_i and rule r_i we can build a strict local argument for p_i.

(1, $\Leftarrow$). We prove the left to right part of (1) using induction on the height of strict local arguments for p_i in $Args_C$.

Inductive Base. Suppose that there is a strict local argument for p_i in $Args_C$ (say A) with height 1. This means that there is a strict local rule with head p_i with empty body in C_i; hence $local\_alg$ will return $localAns_{p_i} = true$.

Inductive Step. Suppose that A is a strict local argument for p_i with height $n+1$ in $Args_C$. Then, there is a strict local rule with head p_i (r_i) in C_i, such that for every literal a_i in its body there is a strict local argument with height $\leq n$ in $Args_C$. By inductive hypothesis, $local\_alg$ returns $localAns_{a_i} = true$ for every $a_i \in body(r_i)$. Consequently $local\_alg$ will return $localAns_{p_i} = true$.

(2, $\Rightarrow$). By the definition of $local\_alg$ it is trivial to verify that $local\_alg$ cannot return both $true$ and $false$ as an answer for a literal p_i. Suppose that $local\_alg$ returns $localAns_{p_i} = false$. Suppose that there is a strict local argument for p_i in $Args_C$. Then (by the first part of the Proposition) $localAns_{p_i} = true$, which contradicts our original hypothesis. Consequently there is no strict local argument for p_i in $Args_C$.

(2, $\Leftarrow$). Similarly (for the right to left part) we suppose that there is no strict local argument for p_i in $Args_C$. Supposing that $local\_alg$ returns $localAns_{p_i} = true$, we conclude (by the first part of the Proposition) that there is a strict local argument for p_i in $Args_C$, which contradicts our original hypothesis.

Auxiliary Lemma 1. For a MCS C and a literal p_i in C:

1. If $P2P\_DR$ returns $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$, then there is an argument A for p_i in $Args_C$, such that A uses applicable rules, and $R(A, C_i)$ equals 0 in case $\Sigma = \emptyset$, or $max_{a \in \Sigma}(R(a, C_i))$ otherwise, and for any other argument B for p_i in $Args_C$, such that B uses applicable rules: $R(A, C_i) \leq R(B, C_i)$.

2. If $P2P\_DR$ returns $Ans_{p_i} = true$ or $Ans_{p_i} = undefined$ and $BS_{p_i} = \Sigma$, then there is an argument A for p_i in $Args_C$, such that A uses unblocked rules, and $R(A, C_i)$ equals 0 in case $\Sigma = \emptyset$, or $max_{a \in \Sigma}(R(a, C_i))$ otherwise, and for any other argument B for p_i in $Args_C$, such that B uses unblocked rules: $R(A, C_i) \leq R(B, C_i)$.

Proof (1). We use induction on the number of calls of $P2P\_DR$ that are required to compute Ans_{p_i} and SS_{p_i}.

Inductive Base. $Ans_{p_i} = true$ derives in one call of $P2P\_DR$. This means that either (a) $localAns_{p_i} = true$ and $SS_{p_i} = \emptyset$, and by Proposition 2, there is a strict local argument A for p_i in $Args_C$. For all literals a in the body of the rules contained in A, $localAns_\alpha = true$. Hence, A uses only applicable rules. Since A is a local argument, $R(A, C_i) = 0$. Hence, there is no argument B such that $R(B, C_i) < R(A, C_i)$; or (b) there is a local defeasible rule with empty body and head p_i in C_i. Using this rule, we can build an argument A for p_i such that $R(A, C_i) = 0$; therefore, there is no argument B such that $R(B, C_i) < R(A, C_i)$.

Inductive Step. $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$ derives in $n + 1$ calls of $P2P\_DR$. This means that there is a rule r_i with head p_i in C_i, such that $\forall \alpha \in body(r_i)$: $P2P\_DR$ returns $Ans_\alpha = true$ and SS_α in at most n calls, and $\Sigma = SS_{r_i}$. By inductive hypothesis, for all α there is an argument A_α for α in $Args_C$ such that A_α uses applicable rules, and $R(A_\alpha, C_j)$ equals 0 in case $SS_\alpha = \emptyset$ or $max_{a' \in SS_\alpha}(R(a', C_j))$ otherwise (where C_j is the context such that $\alpha \in V_j$), and for any other argument B_α for α in $Args_C$ that uses applicable rules: $R(B_\alpha, C_j) \geq R(A_\alpha, C_j)$.

Using the arguments A_α and rule r_i we build an argument A for p_i as follows: The subset of the arguments A_α that support local literals of C_i (denoted as A_{α_i}) are used as proper subarguments of A, and r_i is used in A to support p_i, which labels the root of A. By the definition of rank of arguments:

$$R(A, C_i) = max(max_{A_{\alpha_i}}(R(A_{\alpha_i}, C_i)), max_{a_j}(R(a_j, C_i)))$$

where a_j are the literals in the body of r_i such that $a_j \notin V_i$. By inductive hypothesis:

$$R(A, C_i) = max(max_{a' \in \bigcup SS_{\alpha_i}}(R(a', C_j)), max_{a_j}(R(a_j, C_i)))$$

$$\Rightarrow R(A, C_i) = max_{d \in (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)}(R(d, C_i))$$

$$\Rightarrow R(A, C_i) = max_{d \in SS_{r_i}}(R(d, C_i))$$

and for any other argument A' for p_i in $Args_C$ that uses rule r_i and applicable rules to support p_i, $R(A, C_i) \leq R(A', C_i)$. In case $\Sigma = SS_{r_i} = \emptyset$, which means that there is no foreign literal in the body of r_i, and for every $a_i \in body(r_i)$: $SS_{a_i} = \emptyset$, using inductive hypothesis it is easy to verify that $R(A, C_i) = 0$.

By the definition of *P2P_DR*, it also holds that for any other rule t_i with head p_i in C_i, either (a) there is a literal γ in the body of t_i such that *P2P_DR* returns either $Ans_\gamma = undefined$ or $Ans_\gamma = false$ - in this case t_i is not applicable; or (b) $\forall \gamma \in body(t_i)$: $Ans_\gamma = true$ and $Stronger(\Sigma, SS_{t_i}, T_i) \neq SS_{t_i}$. The latter results are obtained in n or less calls of *P2P_DR*. By inductive hypothesis, the argument for p_i that uses rule t_i and applicable rules to support p_i with the lowest rank w.r.t. C_i is F with rank: $R(F, C_i) = max_{f \in SS_{t_i}}(R(f, C_i))$, and by the definition of *Stronger* it holds that there is a literal f' in SS_{t_i} such that for all d in $\Sigma = SS_{r_i}$, $R(f', C_i) \geq R(d, C_i)$. Therefore $R(A, C_i) \leq R(F, C_i)$. Overall, the rank of A is equal or lower than the rank of any other argument in $Args_C$ that uses applicable rules to support p_i.

(2). We use induction on the number of calls of *P2P_DR* that are required to compute Ans_{p_i} and BS_{p_i}.

Inductive Base. As there are no loops in the local context theories, at least two calls of *P2P_DR* are required to return *undefined* as an answer for p_i. Hence, the Inductive Base for the case that $Ans_{p_i} = true$ is that this answer is returned by *P2P_DR* in one call. Similarly with the first part of the Lemma, we can prove that $BS_{p_i} = \emptyset$, and there is an argument $A \in Args_C$ for p_i, such that $R(A, C_i) = 0$, and A uses only applicable (and therefore unblocked) rules.

The Inductive Base for the case that $Ans_{p_i} = undefined$ and $BS_{p_i} = \Sigma$ is two calls of *P2P_DR*. Since we assume that there are no loops in the local context theories, there are no rules such that the literal in their head also belongs to the body of the rule. Hence, the following conditions must hold: (a) $localAns_{p_i} = false$; by Proposition 2 this means that there is no strict local argument for p_i in $Args_C$; (b) there is no rule with head $\sim p_i$ in C_i; and (c) there is only one rule r_i with head p_i in C_i, with one literal in its body (say q_j), for which it holds (c$_1$) $q_j \notin V_i$; (c$_2$) there is no rule with head $\sim q_j$ in C_j; and (c$_3$) there is only one rule with head q_j (say r_j) in C_j, such that p_i is the only literal in the body of t_j. Hence, the only argument for p_i (A) can be obtained using rule r_i, and the only argument for q_j (A') can be obtained using rule r_j. Neither r_i nor r_j are blocked since there are no rules with contradictory conclusions, and $\Sigma = BS_{r_i} = \{q_j\}$. Therefore A uses unblocked rules, $R(A, C_i) = R(q_j, C_i) = max_{a \in \Sigma} R(a, C_i)$ and there is no other argument for p_i in $Args_C$.

Inductive Step. $Ans_{p_i} = true$ or $Ans_{p_i} = undefined$ and $BS_{p_i} = \Sigma$ derive in $n + 1$ calls of *P2P_DR*. This means that there is a rule r_i with head p_i in C_i, such that $\forall \alpha \in body(r_i)$: *P2P_DR* returns either $Ans_\alpha = true$ or $Ans_\alpha = undefined$ and BS_α in at most n calls, and $\Sigma = BS_{r_i}$. By inductive hypothesis, for all α there is an argument A_α for α in $Args_C$ such that A_α uses unblocked rules, and $R(A_\alpha, C_j)$ equals 0 in case $SS_\alpha = \emptyset$ or $max_{a' \in SS_\alpha}(R(a', C_j))$ otherwise, and for any other argument B_α for α in $Args_C$ that uses unblocked rules: $R(B_\alpha, C_j) \geq R(A_\alpha, C_j)$.

Similarly with the first part of the Lemma, using the arguments for a and rule r_i, we can build an argument A for p_i, such that A uses unblocked rules, and

$$\Rightarrow R(A, C_i) = max_{d \in BS_{r_i}}(R(d, C_i))$$

and for any other argument A' for p_i in $Args_C$ that uses unblocked rules to support p_i, $R(A, C_i) \leq R(A', C_i)$. In case $\Sigma = BS_{r_i} = \emptyset$, using inductive hypothesis it is easy to verify that $R(A, C_i) = 0$.

Proposition 3. For a MCS C and a literal p_i in C_i, $P2P\_DR$ returns:

1. $Ans_{p_i} = true$ iff p_i is justified
2. $Ans_{p_i} = false$ iff p_i is rejected by $JArgs^C$
3. $Ans_{p_i} = undefined$ iff p_i is neither justified nor rejected by $JArgs^C$

Proof. ($\Rightarrow$). We prove the left to right part of the proposition using induction on the calls of $P2P\_DR$.

Inductive Base. (1) $P2P\_DR$ returns $Ans_{p_i} = true$ in one call. This means that either (a) $localAns_{p_i} = true$ - then, by Proposition 2, there is a strict local argument A for p_i in $Args_C$. Hence, $A \in JArgs^C$ and p_i is justified; or (b) there is a local defeasible rule r_i in C_i such that $body(r_i) = \emptyset$ and there is no rule with head $\sim p_i$ in C_i. Therefore, there is an argument A for p_i in $Args_C$ with root p_i, which contains only rule r_i, and there is no argument attacking A. Since A has no proper subarguments and it is not attacked by any argument, $A \in JArgs^C$; therefore p_i is justified.

(2). $P2P\_DR$ returns $Ans_{p_i} = false$ in one call. This means that $localAns_{p_i} = false$ (by Proposition 2 this means that there is no strict local argument for p_i in $Args_C$) and either (a) $localAns_{\sim p_i} = true$ - there is a strict local argument B for $\sim p_i$ in $Args_C$, which by definition is supported by $JArgs^C$, defeats any non-strict argument for p_i in $Args_C$, and is not undercut by $JArgs^C$, and therefore p_i is rejected by $JArgs^C$; or (b) there is a local defeasible rule s_i with head $\sim p_i$ in C_i, such that $body(s_i) = \emptyset$. Therefore, there is an argument B for $\sim p_i$ in $Args_C$, with root p_i, which contains only rule s_i. For B it holds that it has no proper subarguments - therefore it is supported and not undercut by $JArgs^C$ - and $R(B, C_i) = 0$ - therefore it defeats any non-strict argument for p_i. Since there is no strict local argument for p_i in $Args_C$, every argument for p_i is defeated by B; therefore p_i is rejected by $JArgs^C$.

(3). At least two calls of $P2P\_DR$ are required to compute *undefined* as an answer for p_i. Since we assume that there are no loops in the local context theories, there are no rules such that the literal in their head also belongs to the body of the rule. Hence, the following conditions must hold: (a) $localAns_{p_i} = false$; by Proposition 2 this means that there is no strict local argument for p_i in $Args_C$; (b) there is no rule with head $\sim p_i$ in C_i, which means that there is no argument in $Args_C$ attacking the arguments for p_i at their root; and (c) there is only one rule r_i with head p_i in C_i, with one literal in its body (say q_j), for which it holds (c_1) $q_j \notin V_i$; (c_2) there is no rule with head $\sim q_j$ in C; and (c_3) there is only one rule with head q_j (say r_j) in C, such that p_i is the only literal in the body of t_j. Hence, the only argument for p_i (A) can be obtained using rule r_i, and the only argument for q_j (A') can be obtained using rule r_j. None of the two arguments is neither justified by $JArgs^C$ nor rejected by $JArgs^C$ (since there are not attacking arguments). Therefore, p_i is neither justified nor rejected by $JArgs^C$.

Inductive Step. (1). *P2P_DR* returns $Ans_{p_i} = true$ in $n + 1$ calls. The following conditions must hold:

(a) there is a rule r_i with head p_i in C_i, such that for all literals α in its body it holds that $Ans_\alpha = true$ is returned by *P2P_DR* in at most n calls. By inductive hypothesis, for every α, there is an argument A_a with conclusion a in $JArgs^C$. Therefore, for every A_a it holds that either A_a is a local argument, or it is the head of an argumentation line A_{La}, such that every argument in A_{La} is in $JArgs^C$. Using arguments A_a, argumentation lines A_{La} and rule r_i, we can build an argument A for p_i and an argumentation line A_L with head A, such that every proper subargument of A and every argument in $A_L - \{A\}$ are in $JArgs^C$ - in other words, A is supported by $JArgs^C$.

(b) $localAns_{\sim p_i} = false$ - by Proposition 2, there is no strict local argument for $\sim p_i$ in $Args_C$

(c) for all rules s_i with head $\sim p_i$ in C_i, either (c_1) there is a literal b in the body of s_i for which *P2P_DR* returns $Ans_b = false$ in n calls. By inductive hypothesis, b is rejected by $JArgs^C$, which means that every argument for b is defeated by an argument supported by $JArgs^C$. Hence, every argument B using rule s_i in $Args_C$ is undercut by $JArgs^C$; or (c_2) $\forall b \in body(s_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for b (in at most n calls) and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = SS_{r_i}$. By Auxiliary Lemma 1, we conclude that there is an argument A for p_i in $Args_C$, which uses rule r_i and applicable rules to support p_i, and has rank $R(A, C_i) = max_{d \in SS_{r_i}}(R(d, C_i))$, and for every argument B for $\sim p_i$ in $Args_C$ that uses unblocked rules and rule s_i to support

$\sim p_i$, it holds that $R(A, C_i) < R(B, C_i)$; therefore every such argument B does not defeat A at p_i.

Suppose that one of these arguments B defeats a proper subargument of A, D at q_i. Since A uses applicable rules, for q_i *P2P_DR* returns $Ans_{q_i} = true$ in n or less calls. Therefore, by inductive hypothesis, there is an argument D' for q_i in $JArgs^C$. B is not a strict local argument, as in that case q_i would be rejected. Suppose B defeats D' at q_i. Since D' is in $JArgs^C$, B is undercut by $JArgs^C$. In case B attacks but cannot defeat D', by definition it holds that $R(D', C_i) < R(B, C_i)$. But since we have already supposed that B defeats D; $R(B, C_i) \leq R(D, C_i)$. Therefore, $R(D', C_i) < R(D, C_i)$ and $R(A', C_i) < R(A, C_i)$, where A' is the argument for p_i that derives from A by replacing D with D'. Following the same process for every subargument D of A, we can obtain an argument A' for p_i, such that A' is supported by $JArgs^C$ and every argument B, such that B uses unblocked rules and B defeats a proper subargument of A', B is undercut by $JArgs^C$. And since $R(A', C_i) < R(A, C_i)$, it holds that for every such argument B, $R(A', C_i) < R(B, C_i)$; therefore B does not defeat A neither at its inner nodes nor at its root.

Suppose that an argument B for $\sim p_i$ in $Args_C$ uses a rule s_i that is not unblocked. By inductive hypothesis, for some literal b in B, it holds that b is rejected; hence B is undercut by $JArgs^C$.

Overall, using A' and the justified argumentation lines for the foreign literals in the body of r_i, we can obtain an argument for p_i, which is supported by $JArgs^C$, and every argument defeating A' is undercut by $JArgs^C$; therefore A' is acceptable w.r.t. $JArgs^C$, and p_i is justified.

(2). *P2P_DR* returns $Ans_{p_i} = false$ in $n+1$ calls. The following two conditions must hold: (a) $localAns_{p_i} = false$; hence there is no strict local argument for p_i in $Args_C$; and (b) for every rule r_i with head p_i, either (b_1) there is a literal a in the body of r_i, such that *P2P_DR* returns $Ans_a = false$ in at most n calls. By inductive hypothesis, this means that a is rejected, and therefore if $a \in V_i$, every argument A using r_i is defeated by an argument supported by $JArgs^C$, while if $a \notin V_i$, every argumentation line with head A contains an argument that is defeated by an argument supported by $JArgs^C$. In any of the two cases, the arguments using r_i are rejected by $JArgs^C$; or (b_2) there is a rule s_i with head $\sim p_i$ in C_i, such that *P2P_DR* returns $Ans_b = true$ for any literal b in the body of s_i, and for all literals a in the body of r_i, $P2P\_DR$ returns *true* or *undefined* as an answer for a in at most n calls, and $Stronger(BS_{r_i}, SS_{s_i}, T_i) \neq BS_{r_i}$. By inductive hypothesis and Auxiliary Lemma 1, in the same way with before, using rule s_i we can build an argument B for $\sim p_i$ such that B is supported by $JArgs^C$ and has lower or equal rank than any argument A for p_i that uses unblocked rules and rule r_i; therefore B defeats any such argument for p_i.

Consider now the arguments for p_i in $Args_C$ that use at least one rule that is not unblocked. In the same way with before, we can prove that these arguments are defeated by an argument supported by $JArgs^C$.

Therefore, for every argument A for p_i it holds that either A or an argument in every argumentation line with head A is defeated by an argument supported by $JArgs^C$; therefore p_i is rejected by $JArgs^C$.

(3). $P2P\_DR$ returns $Ans_{p_i} = undefined$ in $n+1$ calls. The following conditions must hold:

(a) $localAns_{p_i} = false$ and $localAns_{\sim p_i} = false$; by Proposition 2, there are no strict local arguments for p_i and $\sim p_i$ in $Args_C$;

(b) for all rules r_i with head p_i in C either (b_1) there is a literal a in the body of r_i such that $P2P\_DR$ returns either *false* or *undefined* as an answer for a in n or less calls; or (b_2) for all a, $P2P\_DR$ returns *true* as an answer for a in n or less calls, but there is a rule s_i with head $\sim p_i$ in C_i, such that for every literal b in the body of s_i, $P2P\_DR$ returns either *true* or *undefined* as an answer for b in n or less calls, and $Stronger(SS_{r_i}, BS_{s_i}, T_i) \neq SS_{r_i}$. For the case described in (b_1), using inductive hypothesis, a is not justified; therefore there is no argument for p_i in $Args_C$ that is supported by $JArgs^C$. For the case of (b_2), by inductive hypothesis and Auxiliary Lemma 1, there is an argument B in $Args_C$ that uses s_i to support $\sim p_i$, such that B uses unblocked rules, and for every argument A in $Args_C$ that uses applicable rules and r_i to support p_i, it holds that $R(B, C_i) \leq R(A, C_i)$, which means that B defeats A at p_i. In the same way with before, we can prove that there is an argument B' in $Args_C$, which also uses rule s_i, and has lower rank than B w.r.t. C_i, and an argumentation line B_L with head B', such that no subargument of B or argument in B_L is defeated by an argument supported by $JArgs^C$. Therefore B' is not undercut by $JArgs^C$, and defeats any non-strict argument with applicable rules with head p_i. Since there is no strict local argument for p_i, and for the arguments for p_i that use rules that are not applicable w.r.t. C, it is easy to verify that they are not supported by $JArgs^C$, we reach to the conclusion that for every argument A for p_i in $Args_C$, A is either not supported by $JArgs^C$, or it is attacked by an argument in $Args_C$, which is not undercut by $JArgs^C$; therefore p_i is not justified.

(c) for all rules s_i with head $\sim p_i$ in C_i either (c_1) there is a literal b in the body of s_i, such that $P2P\_DR$ returns either *false* or *undefined* as an answer for b in n or less calls; or (c_2) for all b, $P2P\_DR$ returns *true* as an answer for b in n or less calls, but there is a rule r_i with head p_i in C_i, such that for every literal a in the body of r_i, $P2P\_DR$ returns either *true* or *undefined* as an answer for a in n or less calls, and $Stronger(BS_{r_i}, SS_{s_i}, T_i) = BS_{r_i}$. In the same way with before, we can reach to the conclusion that there is an argument A in $Args_C$, which uses rule r_i, and an argumentation line A_L with head A, such that there is no argument B that is supported by $JArgs^C$ and defeats an argument in A_L. Therefore p_i is not rejected by $JArgs^C$.

$\Leftarrow$ (1). We use induction on the stage of acceptability of arguments with conclusion p_i in $Args_C$.

Inductive Base. Suppose that an argument A for p_i in $Args_C$ is acceptable w.r.t. J_0^C. This means that either: (a) A is a strict local argument for p_i; in this case, by Proposition 2, $P2P\_DR$ will return $localAns_{p_i} = true$, and therefore $Ans_{p_i} = true$; or (b) A is a defeasible local argument in $Args^{C_i}$ that is supported by J_0^C, and every argument defeating A is undercut by J_0^C. Since A is supported by J_0^C, A contains one defeasible rule with head p_i (say r_i) with empty body. Suppose that there is a rule s_i with head $\sim p_i$ in C_i, such that for all literals b in its body, $P2P\_DR$ returns either $Ans_b = true$ or $Ans_b = undefined$ and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = BS_{s_i}$. This means, by Auxiliary Lemma 1, that for all arguments for p_i using applicable rules and rule r_i, there is an argument B' that uses rule s_i and unblocked rules, which has lower rank than A in C_i. But, since $R(A, C_i) = 0$ (A is a local argument of C_i), $R(B', C_i) < 0$, which is not possible. Therefore for every rule s_i with head $\sim p_i$ in C_i, either (c) there is a literal b in the body of s_i for which $P2P\_DR$ returns $Ans_b = false$, or (d) $Stronger(SS_{r_i}, BS_{s_i}, T_i) \neq SS_{s_i}$. Suppose that (d) holds and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = none$. By definition of $Stronger$ and Auxiliary Lemma 1, this means that there is an argument B for $\sim p_i$ in $Args_C$ that uses unblocked rules and rule s_i and $R(B, C_i) = 0$. Therefore B defeats A, and for every rule used in B it holds that for all literals in its body $Ans_b \neq false$. By the first part of the Proposition, this means that there is an argumentation line B_L with head B, such that and no argument in B_L is defeated by an argument supported by J_0^C. However, since B defeats A, B is undercut by J_0^C. Hence for every argumentation line B_L with head B, there is an argument D that is supported by J_0^C and defeats a proper subargument of B or an argument in $B_L - \{B\}$; the latter conclusion contradicts our previous conclusion that no argument in B_L is attacked by J_0^C. Therefore our supposition that $Stronger(SS_{r_i}, BS_{s_i}, T_i) = none$ does not hold. Consequently, for every rule s_i with head $\sim p_i$ in C_i, either (a) there is a literal b in the body of s_i for which $P2P\_DR$ returns $Ans_b = false$, or (b) $Stronger(SS_{r_i}, BS_{s_i}, T_i) = SS_{r_i}$. Overall, $P2P\_DR$ will compute $sup_{p_i} = true$, and either $unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i}$, and eventually will return $true$ as an answer for p_i.

Inductive Step. Suppose that A is an argument for p_i in $Args_C$ that is acceptable w.r.t. J_{n+1}^C. This means that either: (a) A is a strict local argument for p_i - in this case, by Proposition 2, $P2P\_DR$ will return $localAns_{p_i} = true$, and $Ans_{p_i} = true$; or (b) A is supported by J_{n+1}^C and every argument defeating A is undercut by J_{n+1}^C. That A is supported by J_{n+1}^C means that every proper subargument of A is acceptable w.r.t. J_n^C, and there is an argumentation line A_L with head A such that every argument in A_L is acceptable w.r.t. J_n^C. By inductive hypothesis, there is a rule r_i with head p_i in C_i, such that for every literal a in the body of r_i, $P2P\_DR$ returns $Ans_a = true$. Suppose that B is an argument in $Args_C$ that defeats A. By definition, B is undercut by J_{n+1}^C; namely, for every argumentation line B_L with head B, there is an argument D in $Args_C$, which is supported by J_{n+1}^C, and defeats a proper subargument of B or an argument in $B_L - \{B\}$. Since D is supported by J_{n+1}^C, every subargument of D is acceptable w.r.t. J_n^C, and there is an argumentation line D_L with head D, such that every argument in D_L is acceptable w.r.t. J_n^C. Suppose that D defeats B' at q_j (where B' is either a proper subargument of B or an argument in an argumentation line with head B). By inductive hypothesis, there is a rule t_j for q_j in C_j, such that for all literals d in the body of t_j, $P2P\_DR$ returns $Ans_d = true$. It also holds that $R(D, C_j) \leq R(B', C_j)$. Suppose that there is a rule s_j with head q_j in C_j such that for all literals b in the body of s_j, $P2P\_DR$ returns $Ans_b \neq false$ and $Stronger(BS_{s_j}, SS_{t_j}, T_i) = BS_{s_j}$. By

Auxiliary Lemma 1, for every argument D' for $\sim q_j$ in $Args_C$ that uses applicable rules and rule t_j, there is an argument E for q_j in $Args_C$, which uses unblocked rules and rule s_j s.t. $R(E, C_j) < R(D', C_j)$. As E uses unblocked rules, it easy to verify by the first part of the Proposition, that E contains no literals that are rejected by $JArgs^C$, and that there is an argumentation line E_L for q_j such that no argument in E_L is defeated by an argument supported by $JArgs^C$. Following the same process, for every literal that B' is undercut at, we can build an argumentation line B_L for $\sim p_i$, such that no argument in B_L is defeated by an argument supported by $JArgs^C$. This contradicts the fact that every argument defeating A is undercut by $JArgs^C$. Therefore, for all rules s_j with head q_j in C_j, either there is a literal b' in the body of s_j, such that $Ans'_b = false$, or there is a rule t_j with head q_j in C_j, such that for all literals d in the body of t_j, *P2P_DR* returns $Ans_d = true$ and $Stronger(BS_{s_j}, SS_{t_j}, T_i) \neq BS_{s_j}$. These conditions suffice for *P2P_DR* to return *false* as an answer for q_j. Therefore, for rule s_i, which is used in B to support $\sim p_i$, it holds that there is a literal b in the body of s_i, such that $Ans_b = false$.

Consider now a rule s_i with head $\sim p_i$ in C_i, which is contained in an argument in $Args_C$, which does not defeat A. Suppose that for all literals b in the body of s_i, *P2P_DR* returns either *true* or *undefined* as an answer for b, and $Stronger(SS_{r_i}, BS_{s_i}, T_i) \neq SS_{r_i}$. Then, by Auxiliary Lemma 1 and by the first part of the Proposition, we can verify that there is an argument B and an argumentation line B_L with head B, such that no argument in B_L is defeated by an argument supported by $JArgs^C$, and for every argument A' that uses applicable rules and r_i to support p_i, $R(A', C_i) \geq R(B, C_i)$. Using the same reasoning with before, we conclude that p_i is not justified, which contradicts our original supposition. Therefore, for every rule s_i with head $\sim p_i$ in C_i, which is contained in an argument B that does not defeat A, either there is a literal b in the body of s_i, such that *P2P_DR* returns *false* as an answer for b, or $Stronger(SS_{r_i}, BS_{s_i}, T_i) \neq SS_{r_i}$.

Overall, there is a rule r_i with head p_i in C_i, such that for every literal a in the body of r_i, *P2P_DR* returns *true* as an answer for a, and for for every rule s_i for $\sim p_i$ in C_i, either there is a literal b in the body of s_i, such that *P2P_DR* returns *false* as an answer for b, or $Stronger(SS_{r_i}, BS_{s_i}, T_i) \neq SS_{r_i}$. Therefore, *P2P_DR* will compute $sup_{p_i} = true$, and either $unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i}$, and eventually will return *true* as an answer for p_i.

$\Leftarrow (2)$. Suppose that for a literal p_i that is rejected by $JArgs^C$, it holds that $P2P\_DR$ returns either *true* or *undefined* as an answer for p_i. By definition, either (a) $localAns_{p_i} = true$, which means there is a strict local argument for p_i in $Args_C$, and leads to the conclusion that p_i is justified, which by Lemma 2 contradicts our original supposition that p_i is rejected by $JArgs^C$; or (b) there is a rule r_i with head p_i in C_i, such that for all literals a in the body of r_i, $P2P\_DR$ returns either *true* or *undefined* as an answer for a, and for all rules s_i with head $\sim p_i$ in C_i, either there is a literal b in the body of s_i, such that $Ans_b \neq true$ or $Stronger(BS_{r_i}, BS_{s_i}, T_i) = BS_{r_i}$. By Auxiliary Lemma 1 and the first part of the Proposition, this implies that there is an argument A for p_i in $Args_C$ and an argumentation line A_L with head A, such that no argument in $A_L - \{A\}$ and no proper subargument of A is defeated by an argument supported by $JArgs^C$, and for every argument B for $\sim p_i$ in $Args_C$, either B is not supported by $JArgs^C$, or there is an argument D for p_i in $Args_C$ and an argumentation line D_L with head D, such that no argument in $D_L - \{D\}$ and no proper subargument of D is defeated by an argument supported by $JArgs^C$, and $R(D, C_i) < R(B, C_i)$. This leads to the conclusion that p_i is not rejected, which contradicts our original supposition. Therefore, $P2P\_DR$ will return *false* as an answer for p_i.

$\Leftarrow (3)$. This is trivial to prove using the first part of the theorem. For a literal p_i, which is neither justified nor rejected by $JArgs^C$, suppose that $Ans_{p_i} = true$. By the first part of the theorem, p_i is justified (contradiction). Suppose that $Ans_{p_i} = false$. By the first part of the theorem, this means that p_i is rejected by $JArgs^C$ (contradiction). Therefore, for p_i, $P2P\_DR$ will return $Ans_{p_i} = undefined$.

Proposition 5. The total number of calls of $P2P\_DR^O$ that are required for the evaluation of a single query is in the worst case $O(n \times \sum P(n,k))$ (exponential), where n stands for the total number of literals in the system, $\sum$ expresses the sum over $k = 0, 1, ..., n$, and $P(n,k)$ stands for the number of permutations with length k of n elements. If each of the literals in the system is defined by a different context, then the total number of messages exchanged between the system contexts for the evaluation of a query is $O(2 \times n \times \sum P(n,k))$.

Proof. In the worst case in a distributed query evaluation process with $P2P\_DR^O$, each context has to call $P2P\_DR$ once for each of the literals that appear in its local theory and mappings and for each different *history* of the query. Assume that all contexts use all system literals in their theories. Obviously, two different calls of $P2P\_DR^O$ for a literal p_j with same query *history* cannot be made by two different contexts. This means, that overall for each literal in the system and for each different *history* of a query for that literal, at most one (and in the worst case exactly one) call of $P2P\_DR^O$ will be made. Hence, the total number of calls of $P2P\_DR^O$ will be proportional to the total number of system literals, and to the number of different *histories* that may appear in one call. One query *history* is actually a permutation of a subset of the system literals. One literal cannot appear more than once in a query *history*, as in this case $P2P\_DR^O$ will have already detected a loop (*cycle*), and will not permit a recursive call. Since, the number of different permutations of the system literals is equal to $\sum P(n,k))$, where n stands for the total number of system literals, $\sum$ expresses the sum over $k = 0, 1, ..., n$, and $P(n,k)$ stands for the number of permutations with length k of n elements, the total number of calls of $P2P\_DR^O$ is $O(n \times \sum P(n,k))$. With regard to the number of messages, if each literal is defined by a different context, then each call of $P2P\_DR^O$ actually is implemented as a query message between two different contexts. Taking into account also the response messages, the total number of messages for the evaluation of a query is $O(2 \times n \times \sum P(n,k))$.

Proposition 6. In acyclic MCS, the total number of calls of $P2P\_DR^O$ that are required for the evaluation of a single query is in the worst case $O(c \times n)$, where c stands for the total number of contexts in the system, and n stands for the total number of literals in the system. If each of the literals in the system is defined by a different context, then the total number of messages exchanged between the system contexts for the evaluation of a query is $O(2 \times c \times n)$.

Proof. In acyclic MCS, there are no loops in the global knowledge base, and therefore there is no case for $P2P\_DR$ to detect a cycle during query evaluation. In this case, the process and outcome of a query evaluation do not depend on the *history* of the query. Therefore, for $P2P\_DR^O$ we need structures for incoming and outgoing queries with only one record for each of the literals that a query about them has already been evaluated. In the worst case, that each context uses all system literals in its mappings, and the evaluation of a query involves all mapping rules from all contexts, $P2P\_DR^O$ is called at most (and in the worst case exactly) once by each context for each literal in the system. Therefore the number of calls of $P2P\_DR^O$ is in the worst case $c \times n$, where c is the total number of contexts and n is the total number of literals in the system. With regard to the number of messages, if each literal is defined by a different context, then each call of $P2P\_DR^O$ actually is implemented as a query message between two different contexts. Taking into account also the response messages, the total number of messages for the evaluation of a query is in the worst case $2 \times c \times n$.

Lemma 5. For a literal p_i in V_i, $local\_alg$ computes

(1) $localAns_{p_i} = true$ iff $S_{p_i} \in Pr(1...n)$ and $S_{p_i} = \{s\}$

(2) $localAns_{p_i} = false$ iff $S_{p_i} \in Pr(1...n)$ and $S_{p_i} \neq \{s\}$

Proof. ($\Rightarrow$). We use induction on the number of calls of $local\_alg$ that are required to compute the answer for p_i.

Inductive Base. (1) Suppose that $local\_alg$ returns $localAns_{p_i} = true$ in one call. This means that there is a local strict rule with head p_i in C_i, r_i, such that $body(r_i) = \emptyset$. By Lemma 4, it holds that r_i is a strict rule in $T_v(C)$ and $body(r_i) = \emptyset$; therefore $S_{p_i} \in Pr(1...n)$ and $S_{p_i} = \{s\}$.

(2) Suppose that $local\_alg$ returns $localAns_{p_i} = false$ in one call. This means that there is no strict local rule with head p_i in C_i. By Lemma 4, there is no strict rule with head p_i in $T_v(C)$; therefore $S_{p_i} \in Pr(1...n)$ (since $p_i \in V$) and $S_{p_i} \neq \{s\}$.

Inductive Step. (1) Suppose that $n+1$ calls of $local\_alg$ are required to compute $localAns_{p_i} = true$. This means that there is a strict local rule r_i with head p_i in C_i, such that $\forall a_i \in body(r_i)$, $local\_alg$ returns $localAns_{p_i} = true$ in n or less calls. By inductive hypothesis, for every a_i it holds that $S_{a_i} \in Pr(1...n)$ and $S_{a_i} = \{s\}$. By Lemma 4, it also holds that r_i is a strict rule in $T_v(C)$. Therefore, $S_{p_i} \in Pr(1...n)$ and $S_{p_i} = \{s\}$.

(2) Suppose that $n+1$ calls of $local\_alg$ are required to compute $localAns_{p_i} = false$. This means that for every strict local rule r_i with head p_i in C_i, there is a literal a_i in the body of r_i, such that $localAns_{a_i} = false$ is returned by $local\_alg$ in n or less calls. By Lemma 4 and inductive hypothesis, for every strict rule r_i with head p_i in $T_v(C)$, there is one literal a_i in the body of r_i such that $S_{a_i} \neq \{s\}$. Therefore, $S_{p_i} \neq \{s\}$.

($\Leftarrow$). We use induction on the derivation steps in $Pr(1...n)$.

Inductive Base. (1) Suppose that $Pr(2) = S_{p_i} = \{s\}$ (S_{p_i} can be derived in the first step of $Pr(1...n)$ only if there is no rule with head p_i in $T_v(C)$, and in that case $S_{p_i} = \{w\}$). This means that there is a strict rule r_i with head p_i in $T_v(C)$, such that $body(r_i) = \emptyset$. By Lemma 4, r_i is a strict local rule with head p_i in C_i; therefore $local\_alg$ will compute $localAns_{p_i} = true$.

(2) Suppose that $Pr(1) = S_{p_i} \neq \{s\}$. This means that there is no strict rule r_i with head p_i in $T_v(C)$. By Lemma 4, there is no local strict rule with head p_i in C_i; therefore $local\_alg$ will compute $localAns_{p_i} = false$.

Inductive Step. (1) Suppose that $Pr(n+1) = S_{p_i} = \{s\}$. This means that there is a strict rule r_i with head p_i in $T_v(C)$, such that for all literals a_i in the body of r_i, $S_{a_i} = \{s\} \in Pr(1...n)$. By inductive hypothesis, for every a_i it holds that $localAns_{a_i} = true$. By Lemma 4, r_i is a strict local rule with head p_i in C_i; therefore $local\_alg$ will compute $localAns_{p_i} = true$.

(2) Suppose that $Pr(n+1) = S_{p_i} \neq \{s\}$. This means that for every strict rule r_i with head p_i in $T_v(C)$, there is one literal a_i in the body of r_i, such that $S_{a_i} \neq \{s\}$. By Lemma 4 and inductive hypothesis, for every rule r_i with head p_i in C_i, there is a literal a_i in the body of r_i, such that $localAns_{a_i} = false$; therefore $local\_alg$ will compute $localAns_{p_i} = false$.

Lemma 6. For a literal p_i in V_i, $P2P\_DR$ computes

(1) $localAns_{p_i} = false$, $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$ iff $S_{p_i} \in Pr(1...n)$ and $S_{p_i} = \Sigma$ and $\Sigma \neq \{s\}$, $\Sigma \neq \{w\}$.

(2) $Ans_{p_i} = false$ iff $S_{p_i} \in Pr(1...n)$ and $S_{p_i} \neq \{w\}$

Proof. ($\Rightarrow$). We use induction on the number of calls of *P2P_DR* that are required to compute the answer for p_i.

Inductive Base (1). Suppose that *P2P_DR* computes (a) $localAns_{p_i} = false$ and (b) $Ans_{p_i} = true$ in its first call. By Lemma 5, (a) implies that $S_{p_i} \neq \{s\}$. (b) implies that there is a local defeasible rule r_i in C_i, such that $body(r_i) = \emptyset$ and $SS_{r_i} = \emptyset$, and there is no rule with head $\sim p_i$ in C_i. By Lemma 4, r_i is a defeasible rule $T_v(C)$, and since $body(r_i) = \emptyset$, $S_{r_i} = \emptyset \in Pr(1...n)$, and $Stronger(S_{r_i}, S_{t_i}, T_i) \neq S_{t_i}$ for every rule t_i and preference order T_i. Therefore, $S_{p_i} \in Pr(1...n)$ and $S_{p_i} = S_{r_i} = \emptyset = SS_{p_i}$.

(2). Suppose that *P2P_DR* computes $Ans_{p_i} = false$ in its first call. This means that $localAns_{p_i} = false$ and either (a) $localAns_{\sim p_i} = true$ or (b) there is no rule with head p_i in C_i. By Lemma 5, (a) implies that $S_{negp_i} = \{s\} \in Pr(1...n) \Rightarrow$; hence $S_{p_i} = \{w\} \in Pr(1...n)$. (b) also implies $S_{p_i} = \{w\} \in Pr(1...n)$.

Inductive Step (1). Suppose that $localAns_{p_i} = false$ and *P2P_DR* returns $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$ in $n+1$ calls. By Lemma 5, $S_{p_i} \neq \{s\}$. It also holds that:

(a) there is a rule r_i with head p_i in C_i, such that for all literals α in its body, it holds that $Ans_\alpha = true$ is returned by *P2P_DR* in at most n calls, and $SS_{r_i} = (bigcup_{a_i \in V_i} SS_{a_i}) \cup (\bigcup_{a_j \notin V_i} a_j) = \Sigma$, where $\bigcup a_i \cup \bigcup a_j = body(r_i)$. For every a it holds that either (a) $localAns_a = true$ and $SS_a = \emptyset$ - in this case, by Lemma 5, it holds that $S_a \in Pr(1...n)$ and $S_a = \{s\}$ - or (b) $localAns_a = false$, which (by Inductive Hypothesis) means that $S_a \in Pr(1...n)$ and $S_a = SS_a$. Therefore, $(\bigcup_{a_i \in V_i} SS_{a_i}) \cup (\bigcup_{a_j \notin V_i} a_j) = (\bigcup_{a_i'} S_{a_i'}) \cup (\bigcup a_j)$, where a_i' are the literals in the body of r_i s.t. $a_i' \in V_i$ and $S_{a_i'} \neq \{s\}$, and $S_{r_i} = \Sigma$.

(b) $localAns_{\sim p_i} = false$ - by Lemma 5, $S_{\sim p_i} \neq \{s\}$

(c) for all rules s_i with head $\sim p_i$ in C_i, either (c$_1$) there is a literal b in the body of s_i for which *P2P_DR* returns $Ans_b = false$ in n or less calls. By inductive hypothesis, $S_b \in Pr(1...n)$ and $S_b \neq \{w\}$, which means that $w \in S_{s_i}$, or (c$_2$) $\forall b \in body(s_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for b (in at most n calls) and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = SS_{r_i}$. Since there are no loops in the global knowledge base, for all b: $Ans_b = true$, which means that $BS_{s_i} = SS_{s_i}$, and by Inductive Hypothesis $S_{s_i} \in Pr(1...n)$ and $S_{s_i} = SS_{s_i}$. Therefore, it holds that $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$.

(d) for all rules $t_i \neq r_i$ with head p_i in C_i, either (d_1) there is a literal d in the body of t_i for which *P2P_DR* returns $Ans_d = false$ in n calls or (d_2) $\forall d \in body(t_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for d (in at most n calls) and $Stronger(SS_{r_i}, SS_{t_i}, T_i) \neq SS_{t_i}$. Similarly with (c), we can prove that $S_{t_i} \in Pr(1...n)$ and either $w \in S_{t_i}$ or $Stronger(S_{r_i}, S_{t_i}, T_i) \neq S_{t_i}$.

The consequences of (a),(b),(c) and (d) suffice to prove that $S_{p_i} = Pr(1...n)$ and $S_{p_i} = \Sigma$, and Σ does not contain neither s nor w.

(2) Suppose that *P2P_DR* returns $Ans_{p_i} = false$ in $n + 1$ calls. The following two conditions must hold:

(a) $localAns_{p_i} = false$; by Lemma 5, this implies that either $S_{p_i} \neq \{s\}$;

(b) for all rules r_i with head p_i in C_i, either (b_1) there is a literal a in the body of r_i for which *P2P_DR* returns $Ans_a = false$ in n or less calls. By inductive hypothesis, $S_a \in Pr(1...n)$ and $S_a = \{w\}$, which means that $w \in S_{r_i}$, or (b_2) $\forall a \in body(r_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for a (in at most n calls) and there is a rule s_i with head $\sim p_i$ in C_i, for which it holds that $\forall b \in body(s_i)$, *P2P_DR* returns *true* as an answer for b (in at most n calls) and $Stronger(BS_{r_i}, SS_{s_i}, T_i) \neq BS_{r_i}$. By Inductive Hypothesis, and by the fact that there are no loops in the global knowledge base, similarly with before we can prove that for all a: $Ans_a = true$, which means that $BS_{r_i} = SS_{r_i}$, and $S_{r_i}, S_{s_i} \in Pr(1...n)$ and $S_{r_i} = SS_{r_i}$, $S_{s_i} = SS_{s_i}$. Therefore, it holds that $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$. Hence, by ($b_1$) and ($b_2$), it holds that $S_{p_i} = \{w\}$.

($\Leftarrow$). We use induction on the number of calls of derivation steps in $Pr(1...n)$

Inductive Base (1). Suppose that $Pr(2) = S_{p_i} = \Sigma$ and $\Sigma \neq \{s\}$, $\Sigma \neq \{w\}$ (S_{p_i} can be derived in the first step of $Pr(1...n)$ only if there is no rule with head p_i in $T_v(C)$, and in that case $S_{p_i} = \{w\}$). This means that there is a defeasible rule r_i with head p_i in $T_v(C)$, such that $body(r_i) = \emptyset$, and there is no strict rule with head p_i and no rule with head $\sim p_i$ in $T_v(C)$, which means that $\Sigma = \emptyset$. By Lemma 4, r_i is a local defeasible rule in C_i, and there is no strict rule with head p_i and no rule with head $\sim p_i$ in C_i. Therefore, by Lemma 5, $S_{p_i}, S_{\sim p_i} \neq \{s\}$ and *P2P_DR* will compute $SS_{r_i} = \emptyset$, $sup_{p_i} = true$, and $unb_{\sim p_i} = false$, and will eventually return $Ans_{p_i} = true$ and $SS_{p_i} = \emptyset = \Sigma$.

(2). Suppose that $Pr(1) = S_{p_i} = \Sigma$ and $\Sigma = \{w\}$. By definition, this means that there is no rule with head p_i in $T_v(C)$, which by Lemma 4 implies that there is no rule with head p_i in C_i. Therefore, *P2P_DR* will compute $unb_{p_i} = false$ and will eventually return $Ans_{p_i} = false$.

Inductive Step (1) Suppose that $Pr(n+1) = S_{p_i}$ and $\Sigma = S_{p_i}$, and $\Sigma \neq \{s\}$, $\Sigma \neq \{w\}$. By Lemma 5, $S_{p_i} \neq \{s\}$ implies that $localAns_{p_i} = false$. It also holds that there is a rule r_i with head p_i in $T_v(C)$, such that $S_{r_i} = \Sigma \in Pr(1...n)$ and $w \notin S_{r_i}$. By Lemma 4, $r_i \in C_i$. The following conditions must also hold:

(a) for all rules s_i with head $\sim p_i$ in $T_v(C)$ (and therefore in C_i), (a$_1$) $S_{s_i} \in Pr(1...n)$ and $S_{s_i} \neq \{s\}$, which, by Lemma 5, implies that $localAns_{\sim p_i} = false$ and either (a$_2$) $w \in S_{s_i}$, which means that there is a literal b in the body of s_i such that $S_b \in Pr(1...n)$ and $S_b = \{w\}$. By inductive hypothesis, this implies that *P2P_DR* will return $Ans_b = false$ or (a$_3$) $w \notin S_{s_i}$ and $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$. Since $S_{r_i}, S_{s_i} \neq \{s\}$ and $w \notin S_{r_i}, S_{s_i}$, it holds that for all literals $a \in body(r_i)$ and $b \in body(s_i)$, $S_a, S_b \in Pr(1...n)$ and $S_a, S_b \neq w$ and $S_a, S_b \neq \{s\}$. Therefore by inductive hypothesis, for all these literals, *P2P_DR* will return $Ans_a = true$, $SS_a = S_a$ and $Ans_b = true$, $SS_b = S_b$. For the literals for which $S_a = \{s\}$ ($S_b = \{s\}$, by Lemma 5, $SS_a = \emptyset$ ($SS_a = \emptyset$). Hence, $SS_{r_i} = S_{r_i}$ and $SS_{s_i} = S_{s_i}$, and $BS_{s_i} = SS_{s_i}$ (since for all literals b in the body of s_i: $Ans_b = true$), and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = SS_{r_i}$.

(b) for all rules t_i with head p_i in $T_v(C)$ (and therefore in C_i), (b$_1$) $S_{t_i} \in Pr(1...n)$ and $S_{t_i} \neq \{s\}$ and either (b$_2$) $w \in S_{s_i}$ or (b$_3$) $w \notin S_{s_i}$ and $Stronger(S_{r_i}, S_{t_i}, T_i) \neq S_{t_i}$. In the same way with (a), we can prove that for every rule t_i with head p_i in C_i, either there is a literal d in the body of t_i such that $Ans_d = false$ or for all d, $Ans_d = true$ and $Stronger(SS_{r_i}, SS_{t_i}, T_i) \neq SS_{t_i}$.

The above conditions suffice to prove that *P2P_DR* will compute $sup_{p_i} = true$, and $SS_{p_i} = SS_{r_i} = S_{r_i} = \Sigma$, and either $unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{r_i}$, and will eventually return $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$.

(2) Suppose that $Pr(n+1) = S_{p_i}$ and $\Sigma = S_{p_i} = \{w\}$. This means that either:

(a) for all rules r_i with head p_i in $T_v(C)$, $S_{r_i} \in Pr(1...n)$ and $w \in S_{r_i}$. This means that there is a literal a in the body of r_i, such that $S_a \in Pr(1...n)$ and $S_a = w$. By Lemma 4 and inductive hypothesis, for all rules $r_i \in C_i$ it holds that *P2P_DR* will return $Ans_a = false$. Therefore, *P2P_DR* will compute $unb_{p_i} = false$ and will eventually return $Ans_{p_i} = false$.

(b) there is a rule s_i with head $\sim p_i$ in $T_v(C)$, such that $S_{s_i} = \Sigma \in Pr(1...n)$ and $w \notin S_{s_i}$. By Lemma 4, $r_i \in C_i$. It also holds that for all rules r_i with head p_i in $T_v(C)$ (and therefore in C_i), (b_1) $S_{r_i} \in Pr(1...n)$ and $S_{r_i} \neq \{s\}$, and either (b_2) $w \in S_{r_i}$, or (b_3) $w \notin S_{s_i}$ and $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$. In the same way with (1), we can prove that for all literals b in the body if s_i, $Ans_b = true$ and for all rules r_i with head p_i in C_i, either there is a literal a in the body of r_i such that $Ans_a = false$ or for all literals a, $Ans_a = true$ and $Stronger(SS_{r_i}, SS_{s_i}, T_i) \neq SS_{r_i}$. Therefore, *P2P_DR* will compute $sup_{\sim p_i} = true$, and either $unb_{p_i} = false$ or $Stronger(BS_{p_i}, SS_{\sim p_i}, T_i) = SS_{\sim p_i}$, and will eventually return $Ans_{p_i} = false$.

(c) $S_{\sim p_i} \in Pr(1...n)$ and $S_{\sim p_i} = \{s\}$. By Lemma 5, this implies that $localAns_{\sim p_i} = true$; therefore for p_i, *P2P_DR* will return $Ans_{p_i} = false$.

Proposition 7. For a literal p_i in V_i, *local_alg* computes

(1) $localAns_{p_i} = true$ iff $T_v(C) \vdash +\Delta p_i$

(2) $localAns_{p_i} = false$ iff $T_v(C) \vdash -\Delta p_i$

Proof (1,2 $\Rightarrow$). We use induction on the number of calls of *local_alg* that are required to compute the answer for p_i.

Inductive Base. (1) Suppose that *local_alg* returns $localAns_{p_i} = true$ in one call. This means that there is a local strict rule r_i with head p_i in C_i, such that $body(r_i) = \emptyset$. By Lemma 4, it holds that r_i is a strict rule in $T_v(C)$ and $body(r_i) = \emptyset$. Therefore $T_v(C) \vdash +\Delta p_i$ (according to the proof theory of Defeasible Logic presented in Appendix B.

(2) Suppose that *local_alg* returns $localAns_{p_i} = false$ in one call. This means that there is no strict local rule with head p_i in C_i. By Lemma 4, there is no strict rule with head p_i in $T_v(C)$; therefore $T_v(C) \vdash -\Delta p_i$.

Inductive Step. (1) Suppose that $n + 1$ calls of $local\_alg$ are required to compute $localAns_{p_i} = true$. This means that there is a strict local rule with head p_i (say r_i) such that $\forall a_i \in body(r_i)$, $local\_alg$ returns $localAns_{p_i} = true$ in n or less calls. By inductive hypothesis, for every a_i it holds that $T_v(C) \vdash +\Delta a_i$. By Lemma 4, it also holds that r_i is a strict rule in $T_v(C)$. Therefore, $T_v(C) \vdash +\Delta p_i$.

(2) Suppose that $n + 1$ calls of $local\_alg$ are required to compute $localAns_{p_i} = false$. This means that for every strict local rule r_i with head p_i in C_i, there is a literal a_i in the body of r_i, such that $localAns_{a_i} = false$ is returned by $local\_alg$ in n or less calls. By Lemma 4 and inductive hypothesis, for every strict rule r_i with head p_i in $T_v(C)$, there is one literal a_i in the body of r_i such that $T_v(C) \vdash -\Delta a_i$. Therefore, $T_v(C) \vdash -\Delta p_i$.

(1,2 $\Leftarrow$). We use induction on the number of derivation steps in $P(1...n)$ (as this is defined in Appendix B) taking as input the global defeasible theory $T_v(C)$.

Inductive Base. (1) Suppose that $P(1) = +\Delta p_i$. Then there is a strict rule r_i with head p_i in $T_v(C)$, such that $body(r_i) = \emptyset$. By Lemma 4, r_i is a strict local rule in C_i; therefore $local\_alg$ will return $localAns_{p_i} = true$.

(2). Suppose that $P(1) = -\Delta p_i$. Then, there is no strict rule with head p_i in $T_v(C)$. By Lemma 4, there is no local strict rule with head p_i in C_i; therefore $local\_alg$ will return $localAns_{p_i} = false$.

Inductive Step. (1) Suppose that $P(n + 1) = +\Delta p_i$. Then there is a strict rule r_i with head p_i in $T_v(C)$, such that for all literals a_i in the body of r_i: $+\Delta a_i \in P(1...n)$. By Lemma 4 and inductive hypothesis, r_i is a strict local rule in C_i and for all literals a_i in the body of r_i, $local\_alg$ returns $localAns_{a_i} = true$; therefore $local\_alg$ will return $localAns_{p_i} = true$.

(2) Suppose that $P(n+1) = -\Delta p_i$. Then for every strict rule r_i with head p_i in $T_v(C)$, there is a literal a_i in the body of r_i such that $-\Delta a_i \in P(1...n)$. By Lemma 4 and inductive hypothesis, for all local strict rules with head p_i in C_i, there is a literal a_i in

the body of r_i, such that $local\_alg$ returns $localAns_{a_i} = false$; therefore $local\_alg$ will return $localAns_{p_i} = false$.

Proposition 8. For a literal p_i in V_i, $P2P\_DR$ computes

(1) $Ans_{p_i} = true$ iff $T_v(C) \vdash +\partial p_i$

(2) $Ans_{p_i} = false$ iff $T_v(C) \vdash -\partial p_i$

Proof. ($\Rightarrow$). We use induction on the number of calls of $P2P\_DR$ that are required to produce the answer for p_i.

Inductive Base (1). Suppose that $P2P\_DR$ computes $Ans_{p_i} = true$ in its first call. Then, either

(a) $local\_alg$ returns $localAns_{p_i} = true$, which by Proposition 7, implies $T_v(C) \vdash +\Delta p_i$, therefore $T_v(C) \vdash +\partial p_i$; or

(b) there is a local defeasible rule r_i in C_i such that $body(r_i) = \emptyset$ and there is no rule with head $\sim p_i$ in C_i. By Lemma 4, r_i is a defeasible rule $T_v(C)$, and there is no rule with head $\sim p_i$ in $T_v(C)$. Therefore, $T_v(C) \vdash +\partial p_i$.

(2) Suppose that $P2P\_DR$ computes $Ans_{p_i} = false$ in its first call. This means that $localAns_{p_i} = false$ and either (a) $localAns_{\sim p_i} = true$ or (b) there is no rule with head p_i in C_i. By Proposition 7, (a) implies $T_v(C) \vdash +\Delta \sim p_i$; therefore $T_v(C) \vdash -\partial p_i$. By Lemma 4, (b) implies that there is no rule with head p_i in $T_v(C)$; therefore $T_v(C) \vdash -\partial p_i$.

Inductive Step (1) Suppose that $P2P\_DR$ returns $Ans_{p_i} = true$ and $SS_{p_i} = \Sigma$ in $n+1$ calls. The following conditions must hold:

(a) $localAns_{p_i} = false$ (in case $localAns_{p_i} = true$ then $Ans_{p_i} = true$ would be returned by $P2P\_DR$ in its first call). Therefore, by Proposition 7: $T_v(C) \vdash -\Delta p_i$.

(b) $localAns_{\sim p_i} = false$, which, by Proposition 7, implies $T_v(C) \vdash -\Delta \sim p_i$.

(c) there is a rule r_i with head p_i in C_i (by Lemma 4, r_i is also in $T_v(C)$), such that for all literals α in its body, it holds that $Ans_\alpha = true$ is returned by $P2P\_DR$ in at most n calls, and $SS_{r_i} = (bigcup_{a_i \in V_i} SS_{a_i}) \cup (\bigcup_{a_j \notin V_i} a_j) = \Sigma$, where $\bigcup a_i \cup \bigcup a_j$ $= body(r_i)$. For every a, by inductive hypothesis, it holds that $T_v(C) \vdash +\partial a$ and either (a) $localAns_a = true$ and $SS_a = \emptyset$ - in this case, by Lemma 5, it holds that $S_a \in Pr(1...n)$ and $S_a = \{s\}$ - or (b) $localAns_a = false$, which by Lemma 6, implies that $S_a \in Pr(1...n)$ and $S_a = SS_a$ and $w, s \notin S_a$. Therefore, $(\bigcup_{a_i \in V_i} SS_{a_i}) \cup (\bigcup_{a_j \notin V_i} a_j)$ $= (\bigcup_{a'_i} S_{a'_i}) \cup (\bigcup a_j)$, where a'_i are the literals in the body of r_i s.t. $a'_i \in V_i$ and $S_{a'_i} \neq \{s\}$, $w, s \notin S_{r_i}$ and $S_{r_i} = \Sigma$.

(d) for all rules s_i with head $\sim p_i$ in C_i (and therefore in $T_v(C)$), either (d$_1$) there is a literal b in the body of s_i for which *P2P_DR* returns $Ans_b = false$ in n calls - by inductive hypothesis, $T_v(C) \vdash -\partial b$ - or (c$_2$) $\forall b \in body(s_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for b (in at most n calls) and $Stronger(SS_{r_i}, BS_{s_i}, T_i) = SS_{r_i}$. Since there are no loops in the global knowledge base, for all b: $Ans_b = true$, which means that $BS_{s_i} = SS_{s_i}$. By inductive hypothesis, for all b it holds $T_v(C) \vdash +\partial b$. By Lemma 6, $S_{s_i} \in Pr(1...n)$ and $S_{s_i} = SS_{s_i}$ and $w, s \notin S_{s_i}$, which means that $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$, and therefore $r_i > s_i \in Pr(1...n)$.

Overall, it holds that: (i) $T_v(C) \vdash -\Delta \sim p_i$; (ii) there is a rule r_i with head p_i in $T_v(C)$, such that for all literals a in its body: $T_v(C) \vdash +\partial a$; and (iii) for all rules with head $\sim p_i$ in $T_v(C)$ either there is a literal b in the body of s_i, such that $T_v(C) \vdash -\partial b$, or $r > s$. These three conditions suffice to prove: $T_v(C) \vdash +\partial p_i$.

(2) Suppose that *P2P_DR* returns $Ans_{p_i} = false$ in $n+1$ calls. The following two conditions must hold:

(a) $localAns_{p_i} = false$; by Proposition 7, this implies that $T_v(C) \vdash -\Delta p_i$;

(b) for all rules r_i with head p_i, either (b$_1$) there is a literal a in the body of r_i for which *P2P_DR* returns $Ans_a = false$ in n calls. By inductive hypothesis, $T_v(C) \vdash -\partial a$, or (b$_2$) $\forall a \in body(r_i)$: *P2P_DR* returns either *true* or *undefined* as an answer for a (in at most n calls) and there is a rule s_i in C_i with head $\sim p_i$, for which it holds that $\forall b \in body(s_i)$, *P2P_DR* returns *true* as an answer for b (in at most n calls) and $Stronger(BS_{r_i}, SS_{s_i}, T_i) \neq BS_{r_i}$. By Inductive Hypothesis, and by the fact that there are no loops in the global knowledge base, similarly with before we can prove that for all a: $T_v(C) \vdash +\partial a$, $BS_{r_i} = SS_{r_i}$, and $S_{r_i}, S_{s_i} \in Pr(1...n)$, $S_{r_i} = SS_{r_i}$ and $S_{s_i} = SS_{s_i}$, and w, s are not contained neither in S_{r_i} nor in S_{s_i}. Therefore, it holds that $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ and $r_i > s_i \notin Pr(1...n)$.

The above conditions suffice to prove that $T_v(C) \vdash -\partial p_i$.

(1,2 $\Leftarrow$). We use induction on the number of derivation steps in $P(1...n)$.

Inductive Base. (1) Suppose that $P(2) = +\partial p_i$ (according to the proof theory of Defeasible Logic, a defeasible conclusion cannot be derived in one step). This means that either:

(a) $P(1) = +\Delta p_i$, which, by Proposition 7, implies that $localAns_{p_i} = true$ will be returned by $local\_alg$; therefore $P2P\_DR$ will return $Ans_{p_i} = true$; or

(b) there is a defeasible rule r_i with head p_i in $T_v(C)$, such that $body(r_i) = \emptyset$ - by Lemma 4, r_i is a local defeasible rule in C_i; and there is no rule with head $\sim p_i$ in $T_v(C)$, which, by Lemma 4, implies that and there is no rule with head $\sim p_i$ in C_i, hence $unb_{simp_i} = false$. Therefore, $P2P\_DR$ will return $Ans_{p_i} = true$.

(2) Suppose that $P(1) = -\partial p_i$. This means that $P(1) = -\Delta p_i$, which by Proposition 7, implies that $localAns_{p_i} = false$, and either:

(a) there is no rule with head p_i in $T_v(C)$, which, by Lemma 4, implies that there is no rule with head p_i in C_i (therefore $unb_{p_i} = false$); or

(b) $P(1) = +\Delta \sim p_i$, which, by Proposition 7, implies that $localAns_{\sim p_i} = true$.

In both cases, $P2P\_DR$ will return $Ans_{p_i} = false$.

Inductive Step. (1) Suppose that $P(n+1) = +\partial p_i$. This means that either: (a) $+\Delta p_i \in P(1...n)$, which by Proposition 7, implies that $localAns_{p_i} = true$; therefore $P2P\_DR$ will return $Ans_{p_i} = true$; or (b) the following three conditions must hold:

(b_1) $-\Delta \sim p_i \in P(1...n)$, which by Proposition 7, implies that $localAns_{simp_i} = false$;

(b_2) there is a rule r_i with head p_i in $T_v(C)$, such that for all literals a in its body $+\partial a \in P(1...n)$. By Lemma 4 and inductive hypothesis, this implies that there is a rule r_i with head p_i in C_i, such that for all literals a in its body $P2P\_DR$ returns $Ans_a = true$;

(b_3) for all rules s_i with head $\sim p_i$ in $T_v(C)$ (and therefore in C_i), either there is a literal b in the body of s_i, such that $-\partial b \in P(1...n)$ - in this case, by inductive hypothesis, $P2P\_DR$ returns $Ans_b = false$ - or there is a rule t_i with head p_i in $T_v(C)$, such that for all literals d in its body $+\partial d \in P(1...n)$, and $t_i > s_i$. Similarly with (b_2), we can prove for t_i that $t_i \in C_i$, and for all literals d in its body $P2P\_DR$ returns

$Ans_d = true$. That $t_i > s_i$ means that $S_{t_i}, S_{s_i} \in Pr(1...n)$, w, s are not contained neither in S_{t_i} nor in S_{s_i}, and $Stronger(S_{t_i}, S_{s_i}, T_i) = S_{t_i}$. By Lemma 6, $SS_{t_i} = S_{t_i}$, $SS_{s_i} = S_{s_i}$, and since there are no loops in the global knowledge base $BS_{s_i} = SS_{s_i}$. Therefore, it holds that: $Stronger(SS_{t_i}, BS_{s_i}, T_i) = SS_{t_i}$.

By the definition of $Stronger$ it holds that if for every s_i such that for every literal in its body $P2P\_DR$ returns $true$ as an answer, there is a rule t_i that satisfies the same conditions, and $Stronger(SS_{t_i}, SS_{s_i}, T_i) = SS_{t_i}$, then there is a rule r_i that satisfies the same conditions, for which it holds that $Stronger(SS_{r_i}, SS_{s_i}, T_i) = SS_{r_i}$, for every s_i.

Overall, we have proved that $localAns_{simp_i} = false$, and there is a rule r_i with head p_i in C_i, such that for all literals a in its body $P2P\_DR$ returns $Ans_a = true$, and for every rule s_i with head $\sim p_i$ in C_i, either there is a literal b in the body of s_i, such that $Ans_b = true$ or $Stronger(SS_{t_i}, BS_{s_i}, T_i) = SS_{t_i}$. Hence, $sup_{p_i} = true$, and either $unb_{\sim p_i} = false$ or $Stronger(SS_{p_i}, BS_{\sim p_i}, T_i) = SS_{p_i}$, and therefore $P2P\_DR$ returns $Ans_{p_i} = true$.

(2) Suppose that $P(n+1) = -\partial p_i$. This means that $-\Delta p_i \in P(1...n)$, which by Proposition 7, implies that $localAns_{p_i} = false$, and one of the following conditions hold:

(a) $+\Delta \sim p_i \in P(1...n)$, which by Proposition 7, implies that $localAns_{\sim p_i} = true$; therefore $P2P\_DR$ will return $Ans_{p_i} = false$; or

(b) for all rules r_i with head p_i in $T_v(C)$, there is a literal a in the body of r_i such that $-\partial a \in P(1...n)$. By Lemma 4 and inductive hypothesis, this implies that for every rule r_i with head p_i in C_i, there is a literal a in the body of r_i such that $Ans_a = false$; therefore $unb_{p_i} = false$ and $P2P\_DR$ will return $Ans_{p_i} = false$.

(c) there is a rule s_i with head $\sim p_i$ in $T_v(C)$ (by Lemma 4 $s_i \in C_i$), such that for all literals b in its body, $+\partial d \in P(1...n)$ - by Inductive Hypothesis, this implies that $P2P\_DR$ returns $Ans_b = true$, and therefore $sup_{\sim p_i} = true$ - and for all rules t_i with head p_i in $T_v(C)$ (and therefore in C_i) either there is a literal d in the body of t_i such that $-\partial d \in P(1...n)$ - by Inductive Hypothesis, $P2P\_DR$ returns $Ans_d = false$ - or for all literals d, $+\partial d \in P(1...n)$ (by inductive hypothesis $P2P\_DR$ returns $Ans_d = true$) and $t_i > s_i \notin Pr(1...n)$. The latter case implies that $Stronger(S_{t_i}, S_{s_i}, T_i) \neq S_{t_i}$. In the same way with before, we can prove that $BS_{t_i} = S_{t_i}$, $SS_{s_i} = S_{s_i}$, and therefore $Stronger(BS_{t_i}, SS_{s_i}, T_i) \neq BS_{t_i}$. Similarly with before, it is easy to verify that $Stronger(BS_{p_i}, SS_{\sim p_i}, T_i) \neq BS_{p_i}$. Overall, for this case, it holds that $sup_{\sim p_i} = true$ and either $unb_{p_i} = false$ or $Stronger(BS_{t_i}, SS_{s_i}, T_i) \neq BS_{t_i}$, which means that $P2P\_DR$ will return $Ans_{p_i} = false$.

Appendix B

Defeasible Logic

Below, we describe the syntax and proof theory of the ambiguity version of Defeasible Logic, as these were originally presented in [8].

B.1 Syntax

A *defeasible theory* is a triple $(F, R, >)$, where F is a set of literals (called *facts*), R a finite set of rules, and $>$ a superiority relation on R. In expressing the proof theory we consider only propositional rules. Rules containing free variables are interpreted as the set of their variable-free instances.

There are two kinds of rules (fuller versions of defeasible logics include also defeaters): *Strict rules* are denoted by $A \rightarrow p$, where A is a finite set of literals and p is a literal, and are interpreted in the classical sense: whenever the premises are indisputable (e.g. facts) then so is the conclusion. An example of a strict rule is "*Professors are faculty members*". Written formally:

$$professor(X) \rightarrow faculty(X)$$

Inferences from facts and strict rules only are called *definite inferences*. Facts and strict rules are intended to define relationships that are definitional in nature. Thus defeasible logics contain no mechanism for resolving inconsistencies in definite inference.

Defeasible rules are denoted by $A \Rightarrow p$, and can be defeated by contrary evidence. An example of such a rule is

$$professor(X) \Rightarrow tenured(X)$$

which reads as follows:"*Professors are typically tenured*".

A superiority relation is an acyclic relation $>$ on R (that is, the transitive closure of $>$ is irreflexive). Given two rules r_1 and r_2, if we have that $r_1 > r_2$, then we will say that r_1 is *superior* to r_2, and r_2 *inferior* to r_1. This expresses that r_1 may override r_1. For example, given the rules

$$r_1 : professor(X) \Rightarrow tenured(X)$$

$$r_2 : visiting(X) \Rightarrow \neg tenured(X)$$

which contradict each other, no conclusive decision can be made about whether a visiting professor is tenured. But if we introduce a superiority relation $>$ with $r_2 > r_1$, then we can indeed conclude that he/she cannot be tenured.

B.2 Proof Theory

Informally, a conclusion q is defeasibly derivable given a defeasible theory $D = (F, R, >)$ when (a) q is a fact; or (b) there is an applicable strict or defeasible rule for q, and either all the rules for q-complementary literals are discarded or every rule for a q-complementary literal is weaker than an applicable rule for q.

Formally, a *conclusion* of a defeasible theory D is a tagged literal and can have one of the following four forms:

- $+\Delta q$ which is intended to mean that q is a definite consequence of D
- $-\Delta q$ which is intended to mean that we have proved that q is not a definite consequence of D
- $+\partial q$ which is intended to mean that q is defeasible provable in D
- $-\partial q$ which is intended to mean that we have proved that q is not defeasible provable in D

Provability is based on the concept of a derivation in D [8]. A derivation is a finite sequence $P = (P(1), , P(n))$ of tagged literals satisfying the following conditions ($P(1..i)$ denotes the initial part of the sequence P of length i, $R^s[q]$ the set of strict rules that support q and $R^d[q]$ the set of defeasible rules that support q):

$+\Delta$: *If* $P(i+1) = +\Delta q$ *then either*
$q \in F$ *or*
$\exists r \in R^s[q] \forall \alpha \in body(r)$: $+\Delta\alpha \in P(1...i)$

$-\Delta$: *If* $P(i+1) = +\Delta q$ *then either*
$q \notin F$ *and*
$\forall r \in R^s[q] \exists \alpha \in body(r)$: $-\Delta\alpha \in P(1...i)$

$+\partial$: *If* $P(i+1) = +\partial q$ *then either*
(1) $+\Delta q \in P(1...i)$ *or*
(2) (2.1) $\exists r \in R^{sd}[q] \forall \alpha \in body(r)$:
$+\partial\alpha \in P(1...i)$ *and*
(2.2) $-\Delta \sim q \in P(1...i)$ *and*
(2.3) $\forall s \in R[\sim q]$
(2.3.1) $\exists \alpha \in body(s)$: $-\partial\alpha \in P(1...i)$ *or*
(2.3.2) $\exists t \in R^{sd}[q]$:
$\forall \alpha \in body(t)$: $+\partial\alpha \in P(1...i)$ *and* $t > s$

$-\partial$: *If* $P(i+1) = -\partial q$ *then*
(1) $-\Delta q \in P(1...i)$ *and*
(2) (2.1) $\forall r \in R^{sd}[q] \exists \alpha \in body(r)$:
$-\partial\alpha \in P(1...i)$ *or*
(2.2) $+\Delta \sim q \in P(1...i)$ *or*
(2.3) $\exists s \in R[\sim q]$ *such that*
(2.3.1) $\forall \alpha \in body(s)$: $+\partial\alpha \in P(1...i)$ *and*
(2.3.2) $\forall t \in R^{sd}[q]$ *either*
$\exists \alpha \in body(t)$: $-\partial\alpha \in P(1...i)$ *or* $t \not> s$

Governatori et. al describe in [61] Defeasible Logic and its variants in argumentation-theoretic terms. A model theoretic semantics is discussed in [81].

Bibliography

[1] GREGORY D. ABOWD, ANIND K. DEY, PETER J. BROWN, NIGEL DAVIES, MARK SMITH, AND PETE STEGGLES. **Towards a Better Understanding of Context and Context-Awareness**. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307. Springer-Verlag, 1999. 1, 2

[2] PHILIPPE ADJIMAN, PHILIPPE CHATALIC, FRANCOIS GOASDOUE', MARIE-CHRISTINE ROUSSET, AND LAURENT SIMON. **Distributed Reasoning in a Peer-to-Peer Setting: Application to the Semantic Web**. *Journal of Artificial Intelligence Research*, **25**:269–314, 2006. 24, 32, 66

[3] ALESSANDRA AGOSTINI, CLAUDIO BETTINI, AND DANIELE RIBONI. **Experience Report: Ontological Reasoning for Context-aware Internet Services**. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, Washington, DC, USA, 2006. IEEE Computer Society. 4, 8

[4] LEILA AMGOUD AND CLAUDETTE CAYROL. **On the Acceptability of Arguments in Preference-based Argumentation**. In *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 1–7, 1998. 31, 105

[5] LEILA AMGOUD AND CLAUDETTE CAYROL. **A Reasoning Model Based on the Production of Acceptable Arguments**. *Annals of Mathematic and Artificial Intelligence*, **34**(1-3):197–215, 2002. 31, 105

[6] LEILA AMGOUD, CLAUDETTE CAYROL, AND DANIEL LE BERRE. **Comparing Arguments using Preference Orderings for Argument-based Reasoning**.

In *IEEE International Conference on Tools with Artificial Intelligence ICTAI'96*, pages 400–403, Los Alamitos, California, 1996. IEEE Computer Society Press. 31, 105

[7] Leila Amgoud, Simon Parsons, and Laurent Perrussel. **An Argumentation Framework based on contextual Preferences**. In *International Conference on Formal and Applied and Practical Reasoning (FAPR'00)*, pages 59–67, 2000. 31, 32, 105

[8] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J. Maher. **Representation results for defeasible logic**. *ACM Transactions on Computational Logic*, **2**(2):255–287, 2001. 29, 34, 49, 66, 71, 102, 104, 137, 138

[9] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J. Maher. **Embedding defeasible logic into logic programming**. *Theory Pract. Log. Program.*, **6**(6):703–735, 2006. 94, 104

[10] Grigoris Antoniou, David Billington, Guido Governatori, and Michale J. Maher. **A Flexible Framework for Defeasible Logics**. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 405–410. AAAI Press / The MIT Press, 2000. 29

[11] Grigoris Antoniou and Frank vanHarmelen. *A Semantic Web Primer*. MIT Press, Cambridge, MA, USA, 2004. 20

[12] G. Attardi and M. Simi. **A formalization of viewpoints**. *Fundamenta Informaticae*, **23**:149–173, 1995. 21

[13] Franz Baader and Bernhard Hollunder. **Priorities on Defaults with Prerequisites, and Their Application in Treating Specificity in Terminological Default Logic**. *Journal of Automated Reasoning*, **15**(1):41–68, 1995. 106

[14] T.J.M. Bench-Capon. **Persuasion in Practical Argument Using Value-based Argumentation Frameworks**. *Journal of Logic and Computation*, **13**:429–448, June 2003. 31, 105

[15] M. Benerecetti, F. Giunchiglia, and L. Serafini. **Model checking multiagent systems**. *Journal of Logic and Computation*, **8**:8–3, 1998. 20

[16] Massimo Benerecetti, Paolo Bouquet, and Chiara Ghidini. **Formalizing Belief Reports - The Approach and a Case Study**. In *AIMSA*, pages 62–75, 1998. 21

[17] Massimo Benerecetti, Paolo Bouquet, and Chiara Ghidini. **Contextual reasoning distilled**. *JETAI*, **12**(3):279–305, 2000. 19

[18] Philip A. Bernstein, Fausto Giunchiglia, Anastasios Kementsietsidis, John Mylopoulos, Luciano Serafini, and Ilya Zaihrayeu. **Data Management for Peer-to-Peer Computing : A Vision**. In *WebDB*, pages 89–94, 2002. 24

[19] Antonis Bikakis and Grigoris Antoniou. **Defeasible Contextual Logic and its Application to Ambient Intelligence**. *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, accepted. 17

[20] Antonis Bikakis and Grigoris Antoniou. **Defeasible Contextual Reasoning in Ambient Intelligence**. *Knowledge and Information Systems*, under review. 17

[21] Antonis Bikakis and Grigoris Antoniou. **Defeasible Contextual Reasoning with Arguments in Ambient Intelligence**. *IEEE Transactions on Knowledge and Data Engineering*, under review. 17

[22] Antonis Bikakis and Grigoris Antoniou. **Alternative Strategies for Contextual Reasoning with Conflicts in Ambient Computing**. In *Proceedings of Second International Conference on Web Reasoning and Rule Systems (RR 2008)*, **5341** of *Lecture Notes in Computer Science*, pages 234–235. Springer, 2008. 17

[23] Antonis Bikakis and Grigoris Antoniou. **Distributed Defeasible Contextual Reasoning in Ambient Computing**. In *Proceedings of European Conference on Ambient Intelligence (AmI-08)*, **5355** of *Lecture Notes in Computer Science*, pages 308–325. Springer, 2008. 16

[24] Antonis Bikakis and Grigoris Antoniou. **Distributed Defeasible Reasoning in Multi-Context Systems**. In *NMR-08*, 2008. 17

[25] Antonis Bikakis and Grigoris Antoniou. **Distributed Reasoning with Conflicts in a Multi-Context Framework**. In *AAAI*, pages 1778–1779. AAAI Press, 2008. 16

[26] Antonis Bikakis and Grigoris Antoniou. **Local and Distributed Defeasible Reasoning in Multi-Context Systems**. In *Proceedings of International Symposium on Rule Representation, Interchange and Reasoning on the Web (RuleML 2008)*, **5321** of *Lecture Notes in Computer Science*, pages 135–149. Springer, 2008. 17

[27] Antonis Bikakis and Grigoris Antoniou. **Contextual Argumentation in Ambient Intelligence**. In *10th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'09)*, 2009. accepted. 16

[28] Antonis Bikakis, Grigoris Antoniou, and Panayiotis Hassapis. **Alternative Strategies for Conflict Resolution in Multi-Context Systems**. In *Proceedings of the 5th IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI'2009)*, **296** of *IFIP Advances in Information and Communication Technology*. Springer, 2009. 17

[29] Antonis Bikakis, Theodoros Patkos, Grigoris Antoniou, and Dimitris Plexousakis. **A Survey of Semantics-based Approaches for Context Reasoning in Ambient Intelligence**. In *Constructing Ambient Intelligence, AmI-07 Workshops Proceedings*, **3774** of *CCIC*, pages 14–23. Springer-Verlag, 2008. 16

[30] Arnold Binas and Sheila A. **Peer-to-Peer Query Answering with Inconsistent Knowledge**. In *KR*, pages 329–339, 2008. 32

[31] Arnold Binas and Sheila A. McIlraith. **Exploiting Preferences over Information Sources to Efficiently Resolve Inconsistencies in Peer-to-peer Query Answering**. In *AAAI 2007 Workshop on Preference Handling for Artificial Intelligence*, 2007. 24, 25, 32

[32] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. **An Abstract, Argumentation-Theoretic Approach to Default Reasoning**. *Artificial Intelligence*, **93**:63–101, 1997. 26

[33] Alexander Borgida and Luciano Serafini. **Distributed Description Logics: Assimilating Information from Peer Sources**. *Journal of Data Semantics*, **1**:153–184, 2003. 20

[34] Paolo Bouquet and Fausto Giunchiglia. **Reasoning about Theory Formulation and Reformulation: A New Solution to the Qualification Problem**. In *Proceedings of the Second World Conference on the Fundamentals of Artificial Intelligence*, pages 39–50, 1995. 21

[35] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. **C-OWL: Contextualizing Ontologies**. In *International Semantic Web Conference*, pages 164–179, 2003. 20

[36] Gerhard Brewka, Floris Roelofsen, and Luciano Serafini. **Contextual Default Reasoning**. In *IJCAI*, pages 268–273, 2007. 23, 105

[37] Sasa Buvac and Ian A. Mason. **Propositional Logic of Context.** In *AAAI*, pages 412–419, 1993. 19

[38] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. **Inconsistency Tolerance in P2P Data Integration: an Epistemic Logic Approach**. In *DBPL-05*, **3774** of *LNCS*, pages 90–105. SV, 2005. 24

[39] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. **Logical Foundations of Peer-To-Peer Data Integration**. pages 241–251. ACM, 2004. 24

[40] Philippe Chatalic, Gia Hien Nguyen, and Marie-Christine Rousset. **Reasoning with Inconsistencies in Propositional Peer-to-Peer Inference Systems**. In *ECAI*, pages 352–356, 2006. 24, 25, 32

[41] Harry Chen, Tim Finin, and Anupam Joshi. **Semantic Web in a Pervasive Context-Aware Architecture**. *Artificial Intelligence in Mobile System 2003*, pages 33–40, October 2003. 5, 8

[42] Alessandro Cimatti. **Multi-agent reasoning with belief contexts: the approach and a case study**. In *ECAI-94: Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents*, pages 71–85, New York, NY, USA, 1995. Springer-Verlag New York, Inc. 20

[43] James P. Delgrande and Torsten Schaub. **Expressing preferences in default logic**. *Artificial Intelligence*, **123**(1-2):41–87, 2000. 105

[44] Phan Minh Dung. **An Argumentation Semantics for Logic Programming with Explicit Negation**. In *ICLP*, pages 616–630, 1993. 36

[45] Phan Minh Dung. **On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games**. *Artificial Intelligence*, **77**:321–357, 1995. 26, 36, 102

[46] Michael Fisher and Chiara Ghidini. **Programming Resource-Bounded Deliberative Agents**. In *IJCAI*, pages 200–205, 1999. 21

[47] J. Forstadius, O. Lassila, and T. Seppanen. **RDF-based model for context-aware reasoning in rich service environment**. In *PerCom 2005 Workshops*, pages 15–19, 2005. 5, 8

[48] Enrico Franconi, Gabriel M. Kuper, Andrei Lopatenko, and Luciano Serafini. **A Robust Logical and Computational Characterisation of Peer-to-Peer Database Systems**. In *DBISP2P*, pages 64–76, 2003. 24

[49] Charalampos Fretzagias and Maria Papadopouli. **Cooperative Location Sensing for Wireless Networks**. In *Second IEEE International conference on Pervasive Computing and Communications*, Orlando, Florida, 2004. 106

[50] Fabien L. Gandon and Norman M. Sadeh. **Semantic web technologies to reconcile privacy and context awareness**. *Journal of Web Semantics*, **1**:241–260, 2004. 5, 8

[51] Alejandro Javier García and Guillermo Ricardo Simari. **Defeasible Logic Programming: An Argumentative Approach**. *Theory and Practice of Logic Programming*, **4**(1-2):95–138, 2004. 30

[52] David Gelernter. **Generative Communication in Linda**. *ACM Transactions on Programming Languages and Systems*, **7**(1):80–112, 1985. 6

[53] Chiara Ghidini. **Modelling (Un)Bounded Beliefs**. In *CONTEXT '99: Proceedings of the Second International and Interdisciplinary Conference on Modeling and Using Context*, pages 145–158, London, UK, 1999. Springer-Verlag. 21

[54] Chiara Ghidini and Fausto Giunchiglia. **Local Models Semantics, or contextual reasoning=locality+compatibility**. *Artificial Intelligence*, **127**(2):221–259, 2001. 19

[55] Fausto Giunchiglia. **Non-Omniscient Belief as Context-Based Reasoning**. In *Proc. of the 13th International Joint Conference on Artificial Intelligence*, pages 548–554, 1993. 21, 22

[56] Fausto Giunchiglia and Enrico Giunchiglia. **Ideal and real belief about belief: some intuitions**. In *MAAMAW '96: Proceedings of the 7th European workshop on Modelling autonomous agents in a multi-agent world : agents breaking away*, pages 1–12, Secaucus, NJ, USA, 1996. Springer-Verlag New York, Inc. 21

[57] Fausto Giunchiglia and Luciano Serafini. **Multilanguage hierarchical logics, or: how we can do without modal logics**. *Artificial Intelligence*, **65**(1), 1994. 19, 101

[58] Li Gong. **JXTA: A Network Programming Environment**. *IEEE Internet Computing*, **5**(3):88–95, 2001. 88

[59] Georg Gottlob. **Complexity Results for Nonmonotonic Logics**. *Journal of Logic and Computation*, **2**(3):397–425, 1992. 23

[60] Guido Governatori and Michael J. Maher. **An Argumentation-Theoretic Characterization of Defeasible Logic**. In *ECAI*, pages 469–473, 2000. 29, 30, 31

[61] Guido Governatori, Michael J. Maher, David Billington, and Grigoris Antoniou. **Argumentation Semantics for Defeasible Logics**. *Journal of Logic and Computation*, **14**(5):675–702, 2004. 29, 32, 36, 102, 105, 139

[62] Philip D. Gray and Daniel Salber. **Modelling and Using Sensed Context Information in the Design of Interactive Applications**. In *EHCI '01: Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction*, pages 317–336, London, UK, 2001. Springer-Verlag. 1

[63] Tao Gu, Hung Keng Pung, and Da Qing Zhang. **A Middleware for Building Context-Aware Mobile Services**. In *Proceedings of the IEEE Vehicular Technology Conference (VTC 2004)*, Milan, Italy, May 2004. 5, 8

[64] Ramanathan Guha. *Contexts: a formalization and some applications*. PhD thesis, Stanford, CA, USA, 1992. 20

[65] V. Haarslev, R. Möller, and M. Wessel. **RacerPro User Guide**. http://www.racer-systems.com/products/racerpro/users-guide-1-9.pdf, 2005. 4

[66] Volker Haarslev and Ralf Möller. **RACER System Description.** In *IJCAR*, pages 701–706, 2001. 4

[67] Alon Y. Halevy, Zachary G. Ives, Dan Suciu, and Igor Tatarinov. **Schema Mediation in Peer Data Management Systems**. In *ICDE*, page 505, 2003. 24

[68] M. Hatala, R. Wakkary, and L. Kalantari. **Ontologies and Rules in Support of Real-time Ubiquitous Application**. *Journal of Web Semantics, Special Issue on "Rules and ontologies for Semantic Web"*, **3**(1):5–22, 2005. 5, 8

[69] Karen Henricksen and Jadwiga Indulska. **Modelling and Using Imperfect Context Information**. In *Proceedings of PERCOMW '04*, pages 33–37, Washington, DC, USA, 2004. IEEE Computer Society. 2

[70] ISTAG. **Ambient Intelligence: from vision to reality**. Technical report, Information Society Technologies Advisory Group (ISTAG), September 2003. 1

[71] John F. Horty and Richmond H. Thomason and David S. Touretzky. **A Skeptical Theory of Inheritance in Nonmonotonic Semantic Networks**. *Artificial Intelligence*, **42**(2-3):311–348, 1990. 27

[72] Souhila Kaci and Leendert van der Torre. **Preference-based argumentation: Arguments supporting multiple values**. *International Journal of Approximate Reasoning*, **48**(3):730–751, 2008. 31, 105

[73] Antonis C. Kakas and Pavlos Moraitis. **Argumentation based decision making for autonomous agents**. In *AAMAS*, pages 883–890, 2003. 30, 31

[74] Yarden Katz and Jennifer Golbeck. **Social Network-based Trust in Prioritized Default Logic**. In *AAAI*, 2006. 105

[75] Deepali Khushraj, Ora Lassila, and Tim Finin. **sTuples: Semantic Tuple Spaces**. In *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitousŏ04)*, pages 267–277, August 2004. 6, 8

[76] Anders Kofod-Petersen and Marius Mikalsen. **Representing and Reasoning about Context in a Mobile Environment**. *Revue d'Intelligence Artificielle*, **19**(3):479–498, 2005. 5, 8

[77] Panu Korpipaa, Jani Mantyjarvi, Juha Kela, Heikki Keranen, and Esko-Juhani Malm. **Managing Context Information in Mobile Devices**. *IEEE Pervasive Computing*, **02**(3):42–51, 2003. 6, 8

[78] Reto Krummenacher, Jacek Kopecký, and Thomas Strang. **Sharing Context Information in Semantic Spaces.** In *OTM Workshops*, pages 229–232, 2005. 6, 8

[79] Douglas B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems; Representation and Inference in the Cyc Project*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989. 20

[80] Fangzhen Lin and Yoav Shoham. **Argument Systems: A Uniform Basis for Nonmonotonic Reasoning**. In *KR*, pages 245–255, 1989. 28

[81] Michael J. Maher. **A Model-Theoretic Semantics for Defeasible Logic**. In *Paraconsistent Computational Logic*, pages 67–80, 2002. 139

[82] Frederick Maier and Donald Nute. **Ambiguity Propagating Defeasible Logic and the Well-Founded Semantics**. In *Logics in Artificial Intelligence, 10th European Conference, JELIA 2006*, pages 306–318, 2006. 105

[83] Frederick Maier and Donald Nute. **Relating Defeasible Logic to the well-founded semantics for normal logic programs**. In *Proceedings of NMR'06: Eleventh International Workshop on Non-monotonic Reasoning*, 2006. 105

[84] John McCarthy. **Generality in Artificial Intelligence**. *Communications of the ACM*, **30**(12):1030–1035, 1987. 19

[85] John McCarthy and Saša Buvač. **Formalizing Context (Expanded Notes)**. In Atocha Aliseda, Rob van Glabbeek, and Dag Westerståhl, editors, *Computing Natural Language*, pages 13–50. CSLI Publications, Stanford, California, 1998. 19

[86] Sanjay Modgil. **Reasoning about preferences in argumentation frameworks**. *Artificial Intelligence*, 2009. 32

[87] B. Mrohs, M. Luther, R. Vaidya, M. Wagner, S. Steglich, W. Kellerer, and S. Arbanowski. **OWL-SF - A Distributed Semantic Service Framework**. In *Workshop on Context Awareness for Proactive Systems (CAPS 2005)*, 2005. 6, 8

[88] Donald Nute. *Defeasible Deontic Logic*, chapter Apparent Obligation, pages 288–315. Kluwer, 1997. 104

[89] Donald Nute. **Defeasible Logic**. In *Proceedings of the 14th International Conference on Applications of Prolog*, pages 87–114, 2001. 104

[90] Bijan Parsia and Evren Sirin. **Pellet: An OWL DL Reasoner**. In *3rd International Semantic Web Conference (ISWC2004)*, 2004. 4

[91] S. Parsons, C. Sierra, and N. R. Jennings. **Agents that reason and negotiate by arguing**. *Journal of Logic and Computation*, **8**(3):261–292, 1998. 20, 21

[92] Theodore Patkos, Antonis Bikakis, Grigoris Antoniou, Maria Papadopouli, and Dimitris Plexousakis. **Design and challenges of a semantics-based framework for context-aware services**. *International Journal on Reasoning-based Intelligent Systems*, accepted. 5, 8

[93] John L. Pollock. **Defeasible Reasoning**. *Cognitive Science*, **11**(4):481–518, 1987. 26, 27

[94] Henry Prakken. *Logical Tools for Modelling Legal Argument. A Study of Defeasible Reasoning in Law.* Kluwer Academic Publishers, 1997. 29

[95] Henry Prakken and Giovanni Sartor. **A dialectical model of assessing conflicting arguments in legal reasoning**. *Artificial Intelligence and Law*, **4**:331–368, 1996. 29

[96] Henry Prakken and Giovanni Sartor. **Argument-Based Extended Logic Programming with Defeasible Priorities**. *Journal of Applied Non-Classical Logics*, **7**(1), 1997. 29, 30, 31

[97] Anand Ranganathan and Roy H. Campbell. **An infrastructure for context-awareness based on first order logic**. *Personal Ubiquitous Comput.*, **7**(6):353–364, 2003. 5, 8

[98] Floris Roelofsen and Luciano Serafini. **Minimal and Absent Information in Contexts**. In *IJCAI*, pages 558–563, 2005. 23

[99] Ronald Prescott Loui. **Defeat among arguments: a system of defeasible inference**. *Computational Intelligence*, **3**:100–106, 1987. 26

[100] N. S. Ryan, J. Pascoe, and D. R. Morse. **Enhanced Reality Fieldwork: the Context-aware Archaeological Assistant**. In V. Gaffney and M. van Leusen and S. Exxon, editor, *Computer Applications in Archaeology 1997*, British Archaeological Reports, Oxford, October 1998. Tempus Reparatum. 2

[101] Dominik Ryzko and Henryk Rybinski. **Distributed Default Logic for Multi-agent System**. In *IAT '06: Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology*, pages 204–210, Washington, DC, USA, 2006. IEEE Computer Society. 105

[102] Jordi Sabater, Carles Sierra, Simon Parsons, and Nicholas R. Jennings. **Engineering Executable Agents using Multi-context Systems**. *Journal of Logic and Computation*, **12**(3):413–442, 2002. 20

[103] Bill Schilit and M. Theimer. **Disseminating Active Map Information to Mobile Hosts**. *IEEE Network*, **8**(5):22–32, 1994. 1

[104] Luciano Serafini and Paolo Bouquet. **Comparing formal theories of context in AI**. *Artificial Intelligence*, **155**(1-2):41–67, 2004. 19

[105] Guillermo Ricardo Simari and Ronald Prescott Loui. **A Mathematical Treatment of Defeasible Reasoning and its Implementation**. *Artificial Intelligence*, **53**(2-3):125–157, 1992. 30

[106] A. Sinner, T. Kleemann, and A. von Hessling. **Semantic User Profiles and their Applications in a Mobile Environment**. In *Artificial Intelligence in Mobile Systems 2004*, 2004. 4, 8

[107] Jonathan Stillman. **The Complexity of Propositional Default Logics**. In *AAAI*, pages 794–799, 1992. 23

[108] Frieder Stolzenburg, Alejandro Javier García, Carlos Iván Chesñevar, and Guillermo Ricardo Simari. **Computing Generalized Specificity**. *Journal of Applied Non-Classical Logics*, **13**(1):87–113, 2003. 30

[109] Matthias Thimm. In *Proceedings of the Sixth International Workshop on Argumentation in Multi-Agent Systems (ArgMAS'09)*, pages 155–172, May. 30

[110] Matthias Thimm, Alejandro J. Garcia, Gabriele Kern-Isberner, and Guillermo R. Simari. **Using Collaborations for Distributed Argumentation with Defeasible Logic Programming**. In M. Pagnucco and

M. Thielscher, editors, *Proceedings of the 12th International Workshop on Non-Monotonic Reasoning (NMR'08)*, pages 179–188. University of New South Wales, September 2008. 30

[111] Matthias Thimm and Gabriele Kern-Isberner. **A Distributed Argumentation Framework using Defeasible Logic Programming**. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *Proceedings of the 2nd International Conference on Computational Models of Argument (COMMA'08)*, number 172 in Frontiers in Artificial Intelligence and Applications, pages 381–392. IOS Press, May 2008. 30

[112] Tim Berners-Lee and James Hendler and Ora Lassila. **The Semantic Web**. *Scientific American*, May 2001. 20

[113] Alessandra Toninelli, Rebecca Montanari, Lalana Kagal, and Ora Lassila. **A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments**. In *Proc. of 5th International Semantic Web Conference*, pages 5–9, November 2006. 4, 5, 8

[114] Anni-Yasmin Turhan, Thomas Springer, and Michael Berger. **Pushing Doors for Modeling Contexts with OWL DL a Case Study**. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, Washington, DC, USA, 2006. IEEE Computer Society. 4, 8

[115] Ugo Chirico. **Java Internet Prolog**. http://www.ugosweb.com/jiprolog, 2008. 106

[116] Gerard Vreeswijk. *Studies in Defeasible Argumentation*. PhD thesis, Dept. of Mathematics and Computer Science, Vrije Universiteit Amsterdam, 1993. 28

[117] Xiao Hang Wang, J. S. Dong, C. Y. Chin, S. R. Hettiarachchi, and D. Zhang. **Semantic Space: an infrastructure for smart spaces**. *IEEE Pervasive Computing*, **3**(3):32–39, 2004. 4, 8

[118] Xiao Hang Wang, Da Qing Zhang, Tao Gu, and Hung Keng Pung. **Ontology Based Context Modeling and Reasoning using OWL**. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, page 18, Washington, DC, USA, 2004. IEEE Computer Society. 4, 8

[119] Mark Weiser. **The Computer for the 21st Century**. *Scientific American*, **265**(3):94–104, September 1991. 1

[120] Xenophon Zabulis, Thomas Sarmis, Dimitris Grammenis, and Antonis Argyros. **A multicamera vision system supporting the development of wide-area exertainment applications**. In *IAPR Conference on Machine Vision and Applications (MVA'09)*, 2009. 106