Distributed Defeasible Reasoning in Ambient Intelligence

Antonis Bikakis[1], Grigoris Antoniou[1], P. Hassapis[1]

Institute of Computer Science, FO.R.T.H., Greece

Abstract

Ambient Computing environments host various agents that collect, process, change and share the available context information. The imperfect nature of context, the open and dynamic nature of ambient environments, the different viewpoints from which the ambient agents face the same context, and their heterogeneity with respect to the language and inference system that they use, have introduced new challenges in the study of Distributed AI.

The current paper presents a knowledge representation model based on the Multi-Context Systems paradigm that handles these requirements by modeling ambient agents as peers in a P2P system, local context knowledge as peer rule theories, and mapping rules, through which the ambient agents exchange context information, as defeasible rules. To resolve potential inconsistencies that may arise from the interaction of local theories through the mappings (*global conflicts*), the proposed method uses a preference relation on the system peers, which may express the trust that an agent has in the knowledge imported by other agents. On top of this model, we have developed four alternative strategies for *global conflicts* resolution, which differ in the type and extent of context knowledge that the ambient agents exchange in order to evaluate the quality of the imported context information. The four strategies have been respectively implemented in four versions of a distributed reasoning algorithm for query evaluation in Multi-Context Systems.

Key words: Knowledge Representation, Non-monotonic Reasoning, Contextual Reasoning, Ambient Intelligence, Multi-Context Systems

Email addresses: bikakis@ics.forth.gr (Antonis Bikakis), antoniou@ics.forth.gr (Grigoris Antoniou), phasap@csd.uoc.gr (P. Hassapis)

1. Introduction

1.1. Requirements and Challenges of Reasoning in Ambient Intelligence

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*.

The study of ambient computing environments and pervasive computing systems has introduced new research challenges in the field of Distributed Artificial Intelligence. These are mainly caused by the imperfect nature of the available context information and the special characteristics of the ambient environments and the agents that operate in them. Henricksen and Indulska in [24] 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 agents that operate in an ambient 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.

Contextual reasoning has a multiple role in Ambient Intelligence systems. This role 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 agents, the three main challenges of knowledge management in Ambient Intelligence are to enable:

- Reasoning with the highly dynamic and imperfect context.
- 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.
- 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

The reasoning approaches followed so far in Ambient Intelligence systems either neglect the problems caused by the imperfect nature of context (e.g. [2, 36, 38, 39, 41, 40]) or handle them by using mechanistic techniques and by building additional reasoning mechanisms on top of logic models that cannot not inherently deal with the problems of uncertainty, ambiguity and inconsistency. Representative examples of the second category include the context framework of Gaia [33] and the Middleware for Context-Aware Mobile Services (SOCAM [21]). Both of them 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 base 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 offer 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.

Regarding the distribution of the reasoning tasks, most Ambient Intelligence system have been based on fully centralized architectures. The common approach followed in such systems (e.g. [13, 14, 16, 23, 26, 32, 33, 38])

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. The need for more decentralized approaches has recently led several research teams to deploy methods and techniques from Distributed Artificial Intelligence, with the *shared memory* or *blackboard* based architectures of [25, 28, 27] being the most prominent examples. 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 seems to be a necessity.

1.3. Overview of the proposed Approach

In the current study, we present a fully distributed approach for reasoning in ambient computing environments, which is based on the Multi-Context Systems paradigm. Our approach models ambient agents as autonomous logic-based peers in a P2P system and context knowledge possessed by an ambient agent as a system peer's local rule theory, while information exchange between agents is achieved through *mapping* rules that associate their local context knowledge. Even if it is assumed that each context theory is locally consistent, the same assumption will not necessarily hold for the global knowledge base. The unification of local theories may result in inconsistencies that are caused by the mappings. For example, a local theory A may import conflicting context information from two different agents, B and C, through two competing mapping rules. In this case, even if the three different theories are locally consistent, their unification through the mappings defined by A may contain inconsistencies. To deal with this type of inconsistencies (*global conflicts*), we model mappings as defeasible rules, and use context and preference information to resolve the conflicts.

With this model, we aim to capture the three fundamental dimensions of contextual reasoning, as these were formulated in [5]:

- *Partiality.* Each agent may not have immediate access to all available information, so a local theory can be thought as a partial representation of *the world*.
- *Approximation.* Each local theory differs at the level of detail at which a portion of *the world* is represented.
- *Perspective.* Each local theory encodes a different point of view on *the world*.

Furthermore, the P2P paradigm enables us to model:

- Information flow between different ambient agents as message exchange between the system peers
- context changes using the dynamics of a P2P system
- confidence in the knowledge of other agents as trust between the system peers

The rest of the paper is structured as follows. The next section describes three motivating scenarios from the domain of Ambient Intelligence, and discusses their special characteristics. Section 3 presents background knowledge on Multi-Context Systems and discusses the limitations of prominent recent studies on contextual reasoning and peer-to-peer reasoning with regard to the special requirements of Ambient Intelligence. Section 4 presents the proposed representational model, describes four alternative strategies for using context and trust information for conflict resolution, and analyzes their properties. Section 5 demonstrates the application of the proposed reasoning methods in one of the use case scenarios presented in Section 2. Section 6 presents a prototypical implementation of the four strategies in a simulated peer-to-peer system, and the results of an experimental evaluation that we conducted. The last section summarizes and presents the next steps of this work.

2. Motivating Scenarios from the Ambient Intelligence Domain

Below, we describe three use case scenarios from the Ambient Intelligence domain. The aim of these scenarios is to highlight the special challenging characteristics of contextual reasoning in Ambient Intelligence.

2.1. Context-Aware Mobile Phone in an Ambient Classroom

The first scenario involves a context-aware mobile phone that has been configured by professor Dr. Amber to make decisions about whether it should ring or not (in case of incoming calls) based on Dr. Amber's preferences and context. Dr. Amber has the following preferences: His mobile phone should ring in case of an incoming call, unless it is in silent mode or Dr. Amber is busy with some important activity. One such important activity is a lecture for one of the courses he teaches at the university.

Consider the case that Dr. Amber is located in the 'RA201' university classroom reading his emails on his laptop. It is Tuesday, the time is 7.50 p.m., and he has just finished with a lecture for course CS566. The context-aware mobile phone receives an incoming call, but it not in silent mode.

In this case, the mobile phone cannot decide whether it should ring or not based only on its local context knowledge, which includes knowledge about incoming calls and the mode of the mobile phone, as it is not aware of other important context parameters (e.g. Dr. Amber's current activity). Therefore, it will attempt to contact through the wireless network of the university and import context information from other ambient agents that are located nearby, and use this information to infer further knowledge about Dr Amber's current context.

In order to determine about 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 (actually his mobile phone) is located in a university classroom, then Dr. Amber 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 taking place 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 Dr. Amber's personal calendar. According to this, there is a scheduled class event for Tuesdays from 7.00 to 8.00 pm. Based on this knowledge, it can infer that at the current time (Tuesday, 7.50 p.m.), 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 'knows' that Dr. Amber is currently located in 'RA201'.

The classroom manager possesses context knowledge about the state of the classroom. Specifically, it '*knows*' that the classroom projector is off, and imports information about the presence of people in the classroom from an external person detection service; in the specific 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 in the classroom.

Eventually, the context-aware 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 currently giving a lecture. In order to resolve this conflict, the mobile phone must be able to evaluate the information it receives from the various information sources. For example, in case it is aware that the information that derives 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 it reaches to the 'ring' decision.

2.2. 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 this 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 there is not a case of 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 to 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 to the 'emergency call' decision.

2.2.1. Mushroom Hunting in an Ambient Natural Park

The scenario takes place in an ambient 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 on picking mushrooms typically do not know every specie 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. Since 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 or not. 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 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 how to describe them. 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 to describe *amanita velosa* (a formal name for *springtime amanita*), 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 to the conclusion that the mushroom is poisonous, and therefore he should not pick it. Using the knowledge of Dan, he will reach to the same decision. The third option is to use the combined knowledge of Chris and Eric. In the latter case, he will reach to 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.

2.2.2. Common Characteristics of the Three Scenarios

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

- In each case there is a communication mean through which an ambient agent may 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 reach to 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 conflicting 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 has its own vocabulary for describing the 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 reach to a decision, taking into account all the available context information that is distributed between the ambient agents. By communication load, we refer not only to the required number of messages exchanged between the involved devices, but also to the size of these messages.

3. Background

3.1. Multi-Context Systems and Contextual Reasoning

A Multi-Context System consists of a set of *contexts* and a set of inference rules (known as *mapping* or *bridge* rules) that enable information flow

between different contexts. A context can be thought of as a logical theory - a set of axioms and inference rules - that models local context knowledge. 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. Reasoning with multiple contexts requires performing two types of reasoning; (a) *local reasoning*, based on the individual context theories; and (b) *distributed reasoning*, which combines the consequences of local theories using the mappings. The most critical issues of contextual reasoning are the *heterogeneity* of local context theories, and the potential conflicts that may arise from the interaction of different contexts through the mappings.

The notions of *context* and *contextual reasoning* were first introduced in AI by McCarthy in [30], 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* [9, 31]), and the Multi-Context Systems introduced in [18], which later became associated with the Local Model Semantics proposed in [17]. Multi-Context Systems has been argued to be most adequate with respect to the three properties of contextual reasoning (*partiality*, *approximation*, *proximity*) and has been shown to be technically more general than PLC [35]. This formalism was also 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 [34], 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 [8]), which models bridge relations between different contexts as *default rules*.

Both approaches support many of the characteristics of contextual reasoning in ambient environments that we discuss in Section 2. 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 informa-

tion 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 [34], 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 problems that Contextual Default Logic cannot efficiently handle. Specifically, it does not provide ways to model the quality of the imported context information, or preference between two different information sources. In other words, it does not include any notion of priority, not allowing to resolve the potential inconsistencies that may arise while importing conflicting context information from two or more 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 [19, 37], and would add a too heavy computational overhead to the devices operating in ambient environments, which should be expected to have limited computational capabilities.

3.2. Reasoning in Peer Data Management Systems

Our study also relates to several recent studies focused on the development of formal models and methods for reasoning in peer data management systems. A key issue in formalizing data-oriented peer-to-peer systems is the semantic characterization of *mappings* (bridge rules). One approach (followed in [6, 22]) is the first-order logic interpretation of peer-to-peer systems. [11] 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 [15] 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 [10], 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 [12] extends the distributed reasoning methods of [1] to deal with conflicts caused by mutually inconsistent information sources, by detecting them and reasoning without them. The main problem is the same, once again: To perform reasoning, the conflicts are not actually resolved using some external preference or priority information; they are rather isolated. Based on the latter study, [7] proposes algorithms for inconsistency resolution in Peer-to-Peer Query Answering by exploiting a preference relation on the peers. The main drawback of this approach is that the preference ordering of the peers is essentially common for all the peers in the system, not allowing a peer to define and use a different ordering based on its own viewpoint.

The three latter approaches ([10], [12] and [7]), have some common characteristics that meet many of the requirements of ambient environments discussed in Section 2. 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 [10] assumes that all peers share a common alphabet of constants, which is not always realistic in ambient environments.
- The approaches of [10] and [12] 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 [7] assumes a global preference relation for 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 [12] and [7] 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.
- None of the approaches include the notion of privacy. All peers are expected to cooperate and disclose the same type of information dur-

ing distributed query evaluation, and use the same strategy for conflict resolution.

4. Proposed Approach

4.1. Representational Model

Our approach models a Multi-Context System P as a collection of distributed local rule theories P_i in a peer-to-peer system:

$$P = \{P_i\}, i = 1, 2, ..., n$$

Each system peer (context) has a proper distinct vocabulary V_i and a unique identifier i. Each local theory is a set of rules that contain only local propositional literals (literals from the local vocabulary). These are of the form:

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

where i denotes the peer identifier. These rules express local context knowledge and are interpreted in the classical sense: whenever the literals in the body of a local rule $(a_i^1, a_i^2, ...a_i^{n-1})$ are consequences of the local theory, then so is the conclusion of the rule (a_i^n). Local rules with empty body are used to express local factual knowledge.

Each peer also defines mappings that associate literals from its own vocabulary (*local literals*) with literals from the vocabulary of other peers (*foreign literals*). The acquaintances of peer P_i, denoted as $ACQ(P_i)$, are the set of peers that at least one of P_i's mappings involves at least one of their local literals. Mappings are modeled as defeasible rules (rules that can be defeated in the existence of adequate contrary evidence) of the form:

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

The above mapping rule is defined by P_i, and associates some of its own local literals with some of the local literals of P_j, P_k and other system peers. a_i^n is a local literal of the theory that has defined r_i^m (P_i).

Finally, each peer P_i defines a trust level order T_i, which includes a subset of the system peers, and expresses the trust that P_i has in the other system peers. This is of the form:

$$T_i = [P_k, P_l, ..., P_n]$$

A peer P_k is considered more trusted by P_i than peer P_l if P_k precedes P_l in this list. The peers that are not included in T_i are equally trusted by P_i, but less trusted than those that are part of the list.

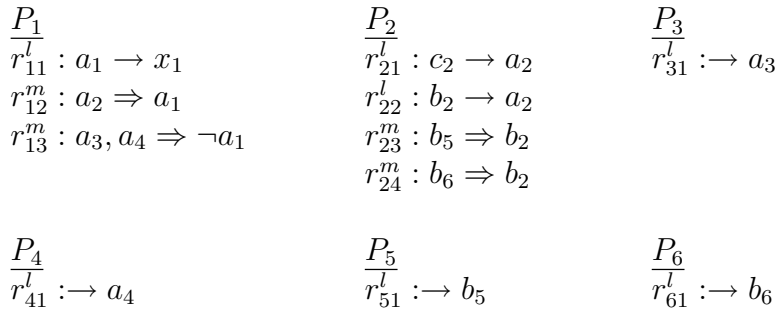

Figure 1: A MCS of Six Context Theories

4.2. Alternative Strategies for Conflict Resolution

In this section, we describe four versions of a distributed algorithm, $P2P\_DR$, for query evaluation in MCS. Each version implements a different strategy for conflict resolution and deals with the following reasoning problem: "*Given a MCS P, and a query about literal x_i issued to peer P_i, find the truth value of x_i considering P_i's local theory, its mappings and the local and mapping rules of the other system peers*".

A common characteristic of the four strategies is that they all use context knowledge and trust information from the system peers to resolve the potential conflicts. Their main difference is in the type and extent of information that the system peers exchange to evaluate the quality of imported context information, and to handle potential inconsistencies that may arise when importing information from two or more different sources. To demonstrate their differences, we will describe how each version of the algorithm is applied to a common scenario; the one depicted in Figure 1.

In this system there are six context theories and a query about literal x_1 is issued to peer P_1. To compute the truth value of x_1, according to the MCS model described in the previous section, P_1 has to import knowledge from P_2, P_3 and P_4. In case the three system peers return positive truth values for a_2, a_3 and a_4 respectively, there will be a conflict about the truth value of a_1 caused by the two conflicting mapping rules, r_{12} and r_{13}.

4.2.1. Single Answers

The *Single Answers* strategy requires each peer to return only the truth value of the literal it is queried about. When a peer receives two conflicting answers from two different peers, it resolves the conflict by comparing the

trust it has in the two peers. The algorithm that implements this strategy, $P2P\_DR^{SA}$ is called by a system peer P_i when it receives a query about one of its local literals (say x_i) and proceeds as follows:

- In the first step, the algorithm determines if the queried literal, x_i, or its negation $\neg x_i$ are consequences of P_i's local theory. If x_i derives from the local rules, the algorithm returns a positive truth value and terminates. If its negation, $\neg x_i$, derives from the local rules, it returns a negative truth value and terminates. In any other case, it proceeds with the second step.
- In the second step, the algorithm collects the local and mapping rules that support x_i. To check which of these rules can be applied, it checks the truth value of the literals in their body by issuing similar queries (recursive calls of the algorithm) to P_i or to the appropriate neighboring peers $P_j \in ACQ_{P_i}$. To avoid cycles, before each new query, it checks if the same query has been issued before, during the same algorithm call. For each applicable supportive rule r_i, the algorithm builds its Supportive Set SS_{r_i}. The Supportive Set of a rule derives from the union of the set of the *foreign literals* (literals that are defined by peers that belong in $ACQ(P_i)$) that are contained in the body of r_i, with the Supportive Sets of the local literals that belong in the body of the same rule. In the end, in case there is no applicable supportive rule, the algorithm terminates by returning a negative answer for x_i. Otherwise, it computes the Supportive Set of x_i, SS_{x_i}, as the *strongest* of the Supportive Sets of the applicable rules that support x_i, and proceeds to the next step. The *strongest* between two Supportive Sets is computed by comparing the weakest elements in each set. A literal a_k is considered stronger than literal b_l from P_i's viewpoint if P_k precedes P_l in T_i.
- In the third step, in the same way with the previous step, the algorithm collects the rules that contradict x_i and builds the Conflicting Set of x_i ($CS_{x_i} = SS_{\neg x_i}$). In case there is no applicable rule that contradicts x_i, the algorithm terminates by returning a positive answer for x_i. If the query was not posed by a foreign peer ($P_j \neq P_i$), the algorithm also returns the Supportive Set of x_i, SS_{x_i}. Otherwise, it proceeds with the last step.
- In the last step, the algorithm compares SS_{x_i} and CS_{x_i} to determine the truth value of x_i. If SS_{x_i} is *stronger*, the algorithm returns a positive answer for x_i. In case the query was not posed by a foreign peer ($P_j \neq$

P_i), the algorithm also returns the Supportive Set of x_i, SS_{x_i}. In any other case (including the case that there is not enough trust information available to give priority to one of the competing answers), it returns a negative answer.

In the system depicted in Figure 1, the algorithm called by P_1 fails to produce a local answer for x_1. In the second step, it attempts to use P_1's mapping rules. The algorithm eventually receives positive answers for a_2, a_3 and a_4 (from the instances of the algorithm called by P_2, P_3 and P_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 its Conflicting Set, $CS_{a_1} = SS_{r_{13}} = \{a_3, a_4\}$. Assuming that the trust level order defined by P_1 is $T_1 = [P_4, P_2, P_6, P_3, P_5]$, $P2P\_DR^{SA}$ determines that SS_{a_1} is stronger than CS_{a_1} (as P_2 precedes P_3 in T_1) and returns a positive answer for a_1 and eventually for x_1 as well.

4.2.2. Strength of Answers

The *Strength of Answers* strategy requires the queried peer to return, along with the truth value of the queried literal, information about whether this value derives from its local theory or from the combination of the local theory with the queried peer's mappings. To support this feature, the second version of the algorithm, $P2P\_DR^{SWA}$, supports three types of answers: (a) a *strict* answer indicates that a positive truth value derives from the local theory only; (b) a *weak* answer indicates that a positive truth value derives from the combination of the local theory and the mapping rules; and (c) a *negative* answer indicates a negative truth value. The peer that receives the answer evaluates its quality based on the trust level of the peer that returns the answer, but also on the type of the answer. This version follows the four main steps of $P2P\_DR_{SA}$ but with the following modifications:

- An element of a Supportive Set is actually a signed literal; the sign of the literal indicates whether the truth value of the literal derives from the local theory of the peer that has defined the literal, or from the combination of the local theory and the peer's mappings.
- The strength of an element in a Supportive/Conflicting 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 peer in the trust level order of the querying peer.

Given these differences, the execution of $P2P\_DR^{SWA}$ in the system depicted in Figure 1, produces the following results: The Supportive Sets of rules r_{12} and r_{13} are respectively: $SS_{r_{12}} = \{weak_{a_2}\}$, $SS_{r_{13}} = \{strict_{a_3}, strict_{a_4}\}$ (the truth values of a_3 and a_4 derive from the local theories of P_3 and P_4 respectively, while P_2 has to use its mappings to compute the truth value of a_2) and $CS_{a_1} = SS_{r_{13}}$ is computed to be stronger than $SS_{a_1} = SS_{r_{12}}$. Eventually, the algorithm computes negative truth values for a_1 and x_1.

4.2.3. Propagating Supportive Sets

The main feature of the *Propagating Supportive Sets* strategy is that along with the truth value of the queried literal, the queried peer returns the Supportive Set of the literal. The algorithm that implements this strategy, $P2P\_DR^{PS}$, constructs the rule and literal Supportive Sets in a similar way with $P2P\_DR^{SA}$; the only difference is that in the case of $P2P\_DR^{PS}$, the Supportive Set of a rule derives from the unification of the Supportive Sets of all (local and foreign) literals in its body.

In the MCS depicted in 1, $P2P\_DR^{PS}$, when called by P_2 to compute the truth value of a_2, assuming that $T_2 = [P_5, P_6]$, returns a positive truth value and its Supportive Set $SS_{a_2} = \{b_5\}$. The answers returned for literals a_3 and a_4 are both positive truth values with empty Supportive Sets (they are locally proved), and $P2P\_DR^{PS}$ called by P_1, computes $SS_{a_1} = SS_{r_{12}} = \{a_2, b_5\}$ and $CS_{a_1} = SS_{r_{13}} = \{a_3, a_4\}$. Using $T_1 = [P_4, P_2, P_6, P_3, P_5]$, $P2P\_DR^{PS}$ determines that CS_{a_1} is stronger than SS_{a_1} (as both P_3 and P_4 precede P_5 in T_1), and computes negative truth values for a_1 and x_1.

4.2.4. Complex Supportive Sets

The *Complex Supportive Sets* strategy, similarly with *Propagating Supportive Sets*, requires the queried peer to return the Supportive Set of the queried literal along with its truth value. In the case of *Propagating Supportive Sets*, the Supportive Set is a set of literals that describes the most trustworthy (according to the trust level of the queried peer) course of reasoning that concludes in the derived truth value. In the case of Complex Supportive Sets, the Supportive Set is actually a set of sets of literals; each set describes a different course of reasoning that results in the same answer. In this case, it is the querying peer that determines the most trustworthy course using its own trust level. $P2P\_DR^{CS}$, the version of the distributed algorithm that implements this strategy, differs from $P2P\_DR^{SA}$ in the followings:

- The Supportive Set of a rule derives from the product of the Supportive Sets of the literals in the body of the rule.
- The Supportive Set of a literal derives from the unification of the Supportive Sets of the applicable rules that support it.
- Comparing the Supportive Set and the Conflicting Set of a literal requires comparing the *strongest sets* of literals of the two sets using the trust level of the peer that resolves the conflict.

In the MCS system of Figure 1, $P2P\_DR^{CS}$, when called by P_2, computes a positive truth value for a_2 and $SS_{a_2} = \{\{b_5\}, \{b_6\}\}$. The instances of the algorithm called by P_3 and P_4 return positive truth values and empty Supportive Sets for a_3 and a_4 respectively, and the instance of $P2P\_DR^{CS}$ called by P_1 computes $SS_{a_1} = SS_{r_{12}} = \{\{a_2, b_5\}, \{a_2, b_6\}\}$ and $CS_{a_1} = SS_{r_{13}} = \{\{a_3, a_4\}\}$. Using $T1 = [P_4, P_2, P_6, P_3, P_5]$, $P2P\_DR^{CS}$ determines that $\{a_2, b_6\}$ is the strongest set in SS_{a_1}, and is also stronger than $\{a_3, a_4\}$. Consequently, it returns a positive answer for a_1 and eventually a positive answer for x_1 as well.

4.3. Properties of the four Strategies

In this section we describe some formal properties of the four strategies with respect to termination, complexity, and the possibility to create an equivalent unified defeasible theory from the distributed context theories. Proposition 1 refers to termination and holds for all versions of $P2P\_DR$ as cycles are detected within the algorithm.

Proposition 1. *The algorithm is guaranteed to terminate returning either a positive or a negative answer for the queried literal.*

Proposition 2 refers to the number of messages that are exchanged between the system peers, and is a consequence of two states that we retain for each peer, which keep track of the incoming and outgoing queries of the peer. It also holds for all versions of $P2P\_DR$.

Proposition 2. *The total number of messages that are exchanged between the system peers for the computation of a single query is $O(n^2)$ (in the worst case that all peers have defined mappings that contain literals from all system peers and the evaluation of the query involves all mappings defined in the system), where n stands for the total number of system peers.*

Moving from the *Single Answers* version to the *Complex Supportive Sets*, the context information that is exchanged between the system peers becomes richer; the first strategy requires each peer to return only a single Boolean value for a given query, while the latter requires building and returning complex (sets of) sets of literals. The advantage of the latter approach is that when receiving an answer, a peer is able to evaluate its quality using its trust ratings of all the peers that are involved in the derivation of the answer, and not only of the peer that returns the answer. This advantage, of course, does not come without a cost, which is the additional computational overhead imposed by the Complex Supportive Sets strategy, and the greater size of the messages carrying the returned answers between the system peer. The following propositions are about the computational complexity of each different version of the algorithm.

Proposition 3. *The computational complexity of* $P2P\_DR^{SA}$ *and* $P2P\_DR^{SWA}$ *on a single peer for the computation of a single query is in the worst case that all peers have defined mappings that involve literals from all system peers and the evaluation of the query requires all peers to use all their mappings* $O(n^2 \times n_l^2 \times n_r)$*, where* n *stands for the total number of system peers,* n_l *stands for the number of literals a peer may define, and* n_r *stands for the total number of (local and mapping) rules that a peer theory may contain.*

Proposition 4. *The computational complexity of* $P2P\_DR^{PS}$ *on a single peer for the computation of a single query is in the worst case that the truth value of the queried literal depends on the truth value of all literals in the system* $O(n^2 \times n_l^2 \times n_r)$*, where* n *stands for the total number of system peers,* n_l *stands for the number of literals a peer may define, and* n_r *stands for the total number of (local and mapping) rules that a peer theory may contain.*

At first glance, the first three versions seem to have equal computational complexity. However, the *worst cases* described in Propositions 3 and 4 are much different. The *worst case* in Proposition 4 occurs when for the evaluation of a single query, the algorithm has to evaluate the truth values of all literals in the system. The *worst case* in Proposition 3 further requires that all peers have defined mappings that involve literals from all other system peers. Obviously, the *worst case* of Proposition 3 is more restrictive and is a sub-case of the worst case described in Proposition 4.

Proposition 5. *The computational complexity of $P2P\_DR^{CS}$ on a single peer for the computation of a single query is in the worst case that the truth value of the queried literal depends on the truth value of literals in the system $O((n \times n_l)^{n \times n_l})$, where n stands for the total number of system peers, n_l stands for the number of literals a peer may define, and n_r stands for the total number of (local and mapping) rules that a peer theory may contain.*

The last proposition shows that the cost of exchanging rich context information between the system peers in order to better evaluate the imported answers is actually too heavy, making the fourth version of the algorithm inapplicable in cases of very dense Multi-Context Systems.

The last property, described in Proposition 6, holds for all versions of $P2P\_DR$, and refers to the soundness and completeness of the algorithm with respect to Defeasible Logic.

Proposition 6. *Using a standard process, it is possible to unify the local context theories into a global defeasible theory, which produces the same results with $P2P\_DR$, under the proof theory of [3]. In this theory, local rules are modeled as strict rules, mapping rules are modeled as defeasible rules, and trust information from the system peers is used to derive priorities between conflicting rules.*

The latter property, which shows the equivalence with a defeasible theory, 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. [1]). Via Proposition 6, the four versions of $P2P\_DR$ have a precise semantic characterization. Defeasible Logic has a proof-theoretic [3], an argumentation-based [20] and a model-theoretic semantics [29]. The proof of Proposition 6 is particularly complex, as it has to take into account matters as how provability in the case of $P2P\_DR$ is interpreted in defeasible provability of Defeasible Logic, and how trust information from the distributed system peers is translated into the priority relation of Defeasible Logic. Details about Proposition 6, as well as the proofs for Propositions 1-5 will be presented elsewhere due to space limitations.

5. Application of the proposed Methods in a use case scenario

In this section, we describe how the proposed representational model and conflict resolution strategies can be applied in one of the Ambient Intelligence use case scenarios described in Section 2, and specifically to the "*Context-Aware Mobile Phone in an Ambient Classroom Scenario*" described in 2.1. The aim, here, is to highlight the suitability of our methods to the special characteristics of Ambient Intelligence, and to demonstrate the practical differences of the four conflict resolution strategies.

5.1. Local Context and Information Exchange Modeling

According to the representational model presented in section 3, the following 6 rules are used to express Dr. Amber's preferences as they are defined in his mobile phone (r^l_{11}, r^l_{12}), the local context knowledge of Dr. Amber's mobile phone (r^l_{13}, r^l_{14}), and the associations of the local knowledge of the mobile phone with the context knowledge of Dr. Amber's laptop (P_3), the localization service (P_4), and the classroom manager (P_4) (r^m_{15}, r^m_{16}).

$r^l_{11} : incoming\_call, \neg silent\_mode, \neg important\_activity \Rightarrow ring$
$r^l_{12} : lecture \rightarrow important\_activity$
$r^l_{13} :\rightarrow incoming\_call$
$r^l_{14} :\rightarrow normal\_mode$
$r^m_{15} : scheduled\_CS566_2, location\_RA201_3 \Rightarrow lecture$
$r^m_{16} : \neg class\_activity_4 \Rightarrow \neg lecture$

In the same way we model the local context knowledge of Dr. Amber's laptop using rules r^l_{21}-r^l_{23}; the knowledge possessed by the localization service through rule r^l_{31}; the knowledge and the mappings of the classroom manager with the person detection service (P_5) using rules r^l_{41}-r^m_{42}; and the local context knowledge of the person detection service using rule r^l_{51}:

$r^l_{21} :\rightarrow day(tuesday)$
$r^l_{22} :\rightarrow time(19.50)$
$r^l_{23} : day(tuesday), time(X), 19.00 \leq X \leq 20.00 \rightarrow scheduled\_CS566$
$r^l_{31} :\rightarrow location\_RA201$
$r^l_{41} :\rightarrow projector(off)$
$r^m_{42} : persons\_detected(X)_5, X < 2, projector(off) \Rightarrow \neg class\_activity$
$r^l_{51} :\rightarrow persons\_detected(1)$

The trust level of P_1 is $T_1 = [P_3, P_4, P_5, P_2]$, which means that the information imported from the localization service is considered more trusted

than that imported from the classroom manager, which is more trusted than the information that derives from the person detection service and so on.

5.2. Conflict Resolution

Upon receiving an incoming call, P_1 will call $P2P\_DR$ to determine about whether it should ring or not. In the case of the *Single Answers* strategy, the algorithm proceeds as follows:

- Using the local context theory of the mobile phone, it cannot reach to a decision as it has no knowledge about Dr Amber's current activity. It can derive a positive value for *incoming_call* and a negative truth value for *silent_mode*, but it cannot determine about the applicability of rules r^l_{11} and r^l_{12}. To do that, it has to determine the truth value of *lecture*.
- For *lecture*, the mobile phone contains two conflicting mapping rules (r^m_{15} and r^m_{16}). Through r^m_{15}, the algorithm accesses the local knowledge of P_2 (the laptop) and P_3 (localization service), computes positive truth values for both *scheduled_CS566* and *location_RA201*, and determines that r^m_{15} is applicable.
- Through rule r^m_{16}, the algorithm accesses the local knowledge of P_4 (classroom manager) and through P_4's mapping rule r^m_{42}, it also uses the knowledge of P_5 (person detection service) to determine that $\neg class\_activity$ is true and r^m_{16} is applicable.
- The algorithm uses the trust level order of P_1 to determine which of the two conflicting mapping rules (r^m_{15} and r^m_{16}) is *stronger*. P_4 precedes P_2 in T_1, so the algorithm determines that r^m_{16} is stronger, and computes a negative truth value for *lecture* and consequently for *important_activity* as well.
- The algorithm determines that r^l_{11} is applicable and reaches to the *ring* decision.

In the case of the *Strength of Answers* strategy, the mobile phone receives, strict positive answers for *scheduled_CS566* and *location_RA201* from P_2 and P_3 respectively (they derive from the local theories of the laptop and the localization service), and a weak positive answer for $\neg class\_activity$ from P_4 (it derives from the combination of the local theory of the classroom manager with its mappings). As a result, the Supportive Set of *lecture* will only contain strict positive values, while the Conflicting Set will contain a weak positive one, the algorithm will compute a positive truth value for

lecture and consequently for *important_activity* as well, and will not reach to the *ring* decision.

In the case of the *Propagating Supportive Sets* and the *Complex Supportive Sets* strategies, the algorithm, compared to the *Single Answers* strategy, will additionally take into account the rank of the person detection service (P_5) in the trust level order of the mobile phone (T_1) for the evaluation of the answer it receives from the classroom manager (P_4). As both P_4 and P_5 precede P_3 in T_1, the algorithm will compute a negative truth value for *lecture*, and eventually it will reach to the *ring* decision.

6. Prototypical Implementation and Experimental Evaluation

6.1. Implementation

In order to evaluate the four strategies, we implemented the respective versions of the $P2P\_DR$ algorithm 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, from personal computers to mobile phones. This adds an extra advantage when creating applications that involve multiple types of devices.

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* 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 algorithms. The *logic* package contains the classes the represent (in memory) the literals and rules. The *knowledge* package includes the *KnowledgeBase* class (a Singleton class that stores the local and mapping rules, the trust level order and

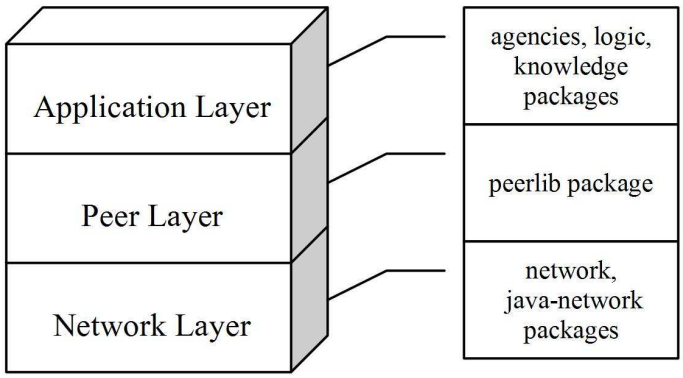

Figure 2: System Layered Architecture.

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 2 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 trust and rule files as well as the initialization of the network libraries and knowledge base. After that, it waits for pending queries. Finally, there is another class named *Client*, which can be used to connect to a *Node* specified by IP address, so that one can manually make specific queries for the rules of that peer instance.

6.2. Experimental Evaluation

The goal of the experiments that we conducted 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 for the four strategies using MCS with various peer populations.

6.2.1. Setup of the Experiments

The experiments that we conducted required test theories that correspond to the worst cases that we described in Section 4. Using a tool that we built for the needs of the experiments, we created theories that correspond to the worst case that the computation of a single query requires computing the

truth value of all literals from all system nodes. The test theories that we created 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 P_0 and associate the truth value of its local literal a_0 with the truth value of the literals from n other system peers. Half of them support a_0 as their conclusion, while the remaining rules contradict a_0. In case the truth values returned for all foreign literals a_1, a_2,...,a_n are all positive then all mapping rules are applicable and are involved in the computation of the truth value of a_0.

In order to exclude the communication overhead from the total time spent by P_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:

- $P2P\_DR^{SA}$: Positive truth values for all literals
- $P2P\_DR^{SWA}$: Positive strict/weak answers (chosen randomly) for all literals.
- $P2P\_DR^{PS}$: Positive truth values with Supportive Sets that contain all the other foreign literals:

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

- $P2P\_DR^{PS}$: 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 made six experiments with a variant number of system peers (n): 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 1: Computation Time for the Four Strategies

♯ peers (n)	SA	SWA	PSS	CSS
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	-

6.2.2. Results

Table 1 shows in milliseconds the computation time for each version of $P2P\_DR$. For the case of $P2P\_DR^{CS}$, we were able to measure the computation time only for the cases n = 10, 20, 40; in the other cases the test machine ran out of memory.

As it is obvious from Table 1, 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 Propositions 4, 5. 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.

7. Conclusions and Future Work

This paper proposed a totally distributed approach for reasoning in Ambient Computing environments based on the Multi-Context Systems paradigm. To handle inconsistency in the distributed context knowledge, we added nonmonotonic features in Multi-Context Systems. The proposed model uses local rule theories to express local context knowledge, defeasible rules for the definition of mappings, and a preference relation to resolve conflicts that derive from the interaction of distributed theories through the mappings. We also described four strategies that use context and preference information for conflict resolution, and analyzed their formal properties with respect to

termination, complexity and the possibility to create an equivalent global defeasible theory from the distributed contexts. We demonstrated the use of the algorithm in a use case scenario from the Ambient Intelligence domain. Finally, we described the implementation of the four strategies in a simulated peer-to-peer environment, which we used to evaluate the strategies with respect to their computational overhead. The obtained results highlight the tradeoff between the extent of context information exchanged between the ambient agents to evaluate the quality of the imported context and the computational complexity of the algorithms that implement the four strategies. Part of our ongoing work includes:

- Implementing the algorithm in Logic Programming, using the equivalence with Defeasible Logic, and the well-studied translation of defeasible knowledge into logic programs under Well-Founded Semantics [4].
- Implementing the described scenarios in a real ambient peer-to-peer environment with peers lying on a variety of stationary and mobile devices (such as PDAs or cell phones).
- Adding non-monotonic features in the local context theories to support uncertainty in the local context knowledge.
- Extending the algorithm to support overlapping vocabularies, enabling different contexts to use elements of common vocabularies (e.g. URIs).
- Studying more applications in the Ambient Intelligence and Semantic Web domains, where the theories may represent ontological context knowledge, policies and regulations.

References

[1] Adjiman, P., Chatalic, P., Goasdoue', F., Rousset, M.-C., Simon, L., 2006. Distributed Reasoning in a Peer-to-Peer Setting: Application to the Semantic Web. Journal of Artificial Intelligence Research 25, 269–314.

[2] Agostini, A., Bettini, C., Riboni, D., 2005. Loosely Coupling Ontological Reasoning with an Efficient Middleware for Context-awareness. In: Proceedings of MobiQuitous 2005. pp. 175–182.

[3] Antoniou, G., Billington, D., Governatori, G., Maher, M. J., 2001. Representation results for defeasible logic. ACM Transactions on Computational Logic 2 (2), 255–287.

[4] Antoniou, G., Billington, D., Governatori, G., Maher, M. J., 2006. Embedding defeasible logic into logic programming. Theory and Pracice of Logic Programming 6 (6), 703–735.

[5] Benerecetti, M., Bouquet, P., Ghidini, C., 2000. Contextual reasoning distilled. JETAI 12 (3), 279–305.

[6] Bernstein, P. A., Giunchiglia, F., Kementsietsidis, A., Mylopoulos, J., Serafini, L., Zaihrayeu, I., 2002. Data Management for Peer-to-Peer Computing : A Vision. In: WebDB. pp. 89–94.

[7] Binas, A., McIlraith, S. A., 2007. 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.

[8] Brewka, G., Roelofsen, F., Serafini, L., 2007. Contextual Default Reasoning. In: IJCAI. pp. 268–273.

[9] Buvac, S., Mason, I. A., 1993. Propositional Logic of Context. In: AAAI. pp. 412–419.

[10] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R., 2005. Inconsistency Tolerance in P2P Data Integration: an Epistemic Logic Approach. In: DBPL-05. Vol. 3774 of LNCS. SV, pp. 90–105.

[11] Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R., 2004. Logical Foundations of Peer-To-Peer Data Integration. ACM, pp. 241–251.

[12] Chatalic, P., Nguyen, G. H., Rousset, M.-C., 2006. Reasoning with Inconsistencies in Propositional Peer-to-Peer Inference Systems. In: ECAI. pp. 352–356.

[13] Chen, H., Finin, T., Joshi, A., October 2003. Semantic Web in a Pervasive Context-Aware Architecture. Artificial Intelligence in Mobile System 2003, 33–40.

[14] Forstadius, J., Lassila, O., Seppanen, T., 2005. RDF-based model for context-aware reasoning in rich service environment. In: PerCom 2005 Workshops. pp. 15–19.

[15] Franconi, E., Kuper, G. M., Lopatenko, A., Serafini, L., 2003. A Robust Logical and Computational Characterisation of Peer-to-Peer Database Systems. In: DBISP2P. pp. 64–76.

[16] Gandon, F. L., Sadeh, N. M., 2004. Semantic web technologies to reconcile privacy and context awareness. Journal of Web Semantics 1, 241–260.

[17] Ghidini, C., Giunchiglia, F., 2001. Local Models Semantics, or contextual reasoning=locality+compatibility. Artificial Intelligence 127 (2), 221–259.

[18] Giunchiglia, F., Serafini, L., 1994. Multilanguage hierarchical logics, or: how we can do without modal logics. Artificial Intelligence 65 (1).

[19] Gottlob, G., 1992. Complexity Results for Nonmonotonic Logics. Journal of Logic and Computation 2 (3), 397–425.

[20] Governatori, G., Maher, M. J., Billington, D., Antoniou, G., 2004. Argumentation Semantics for Defeasible Logics. Journal of Logic and Computation 14 (5), 675–702.

[21] Gu, T., Pung, H. K., Zhang, D. Q., May 2004. A Middleware for Building Context-Aware Mobile Services. In: Proceedings of the IEEE Vehicular Technology Conference (VTC 2004). Milan, Italy.

[22] Halevy, A. Y., Ives, Z. G., Suciu, D., Tatarinov, I., 2003. Schema Mediation in Peer Data Management Systems. In: ICDE. p. 505.

[23] Hatala, M., Wakkary, R., Kalantari, L., 2005. 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.

[24] Henricksen, K., Indulska, J., 2004. Modelling and Using Imperfect Context Information. In: Proceedings of PERCOMW '04. IEEE Computer Society, Washington, DC, USA, pp. 33–37.

[25] Khushraj, D., Lassila, O., Finin, T., August 2004. sTuples: Semantic Tuple Spaces. In: First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous04). pp. 267–277.

[26] Kofod-Petersen, A., Mikalsen, M., 2005. Representing and Reasoning about Context in a Mobile Environment. Revue d'Intelligence Artificielle 19 (3), 479–498.

[27] Korpipaa, P., Mantyjarvi, J., Kela, J., Keranen, H., Malm, E.-J., 2003. Managing Context Information in Mobile Devices. IEEE Pervasive Computing 02 (3), 42–51.

[28] Krummenacher, R., Kopecký, J., Strang, T., 2005. Sharing Context Information in Semantic Spaces. In: OTM Workshops. pp. 229–232.

[29] Maher, M. J., 2002. A Model-Theoretic Semantics for Defeasible Logic. In: Paraconsistent Computational Logic. pp. 67–80.

[30] McCarthy, J., 1987. Generality in Artificial Intelligence. Communications of the ACM 30 (12), 1030–1035.

[31] McCarthy, J., Buvač, S., 1998. Formalizing Context (Expanded Notes). In: Aliseda, A., van Glabbeek, R., Westerståhl, D. (Eds.), Computing Natural Language. CSLI Publications, Stanford, California, pp. 13–50.

[32] Patkos, T., Bikakis, A., Antoniou, G., Plexousakis, D., Papadopouli, M., 2007. A Semantics-based Framework for Context-Aware Services: Lessons Learned and Challenges. In: Proceedings of 4th International Conference on Ubiquitous Intelligence and Computing (UIC-2007). Vol. 4611 of LNCS. Springer, pp. 839–848.

[33] Ranganathan, A., Campbell, R. H., 2003. An infrastructure for context-awareness based on first order logic. Personal Ubiquitous Comput. 7 (6), 353–364.

[34] Roelofsen, F., Serafini, L., 2005. Minimal and Absent Information in Contexts. In: IJCAI. pp. 558–563.

[35] Serafini, L., Bouquet, P., 2004. Comparing formal theories of context in AI. Artificial Intelligence 155 (1-2), 41–67.

[36] Sinner, A., Kleemann, T., von Hessling, A., 2004. Semantic User Profiles and their Applications in a Mobile Environment. In: Artificial Intelligence in Mobile Systems 2004.

[37] Stillman, J., 1992. The Complexity of Propositional Default Logics. In: AAAI. pp. 794–799.

[38] Toninelli, A., Montanari, R., Kagal, L., Lassila, O., November 2006. A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments. In: Proc. of 5th International Semantic Web Conference. pp. 5–9.

[39] Turhan, A.-Y., Springer, T., Berger, M., 2006. 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. IEEE Computer Society, Washington, DC, USA.

[40] Wang, X. H., Dong, J. S., Chin, C. Y., Hettiarachchi, S. R., Zhang, D., 2004. Semantic Space: an infrastructure for smart spaces. IEEE Pervasive Computing 3 (3), 32–39.

[41] Wang, X. H., Zhang, D. Q., Gu, T., Pung, H. K., 2004. Ontology Based Context Modeling and Reasoning using OWL. In: PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops. IEEE Computer Society, Washington, DC, USA, p. 18.