E-contracting Agents Reasoning Hypothetically and Nonmonotonically

Georgios K. Giannikis and Aspassia Daskalopulu

Department of Computer and Communications Engineering,
University of Thessaly,
38221 Volos, Greece
{ggiannik,aspassia}@inf.uth.gr

Abstract. This report discusses work conducted within a broader project that is concerned with the development of an open computational environment for e-contracting. We investigate the need for hypothetical nonmonotonic reasoning in e-contracting, where, realistically, agents possess incomplete knowledge about their environment and other agents. The questions we seek to address, in such environments, are: whether it is possible for agents to identify appropriate assumptions dynamically; how these assumptions affect subsequent inferences; and what happens when new information that becomes available at some time point confirms or disproves assumptions made at previous times. We propose the representation of contract norms as default rules that are constructed dynamically from an initial contract representation in some temporal logic, e.g. in Event Calculus. We argue that a representation of contracts as Default Theories is suitable for temporal, hypothetical and nonmonotonic reasoning. Specifically, we propose a technique that can be used for theory construction and, subsequently for temporal hypothetical nonmonotonic reasoning, which enables the dynamic and ad hoc identification of candidate assumptions, without resorting to proof, and hence computationally viable. We use a full example of a contract and illustrate via examples the way this technique addresses all three issues of interest and supports the execution and performance monitoring of e-contracts.

Keywords. Agents, Open Systems, Incomplete Knowledge, Temporal Reasoning, Hypothetical Reasoning, Nonmonotonic Reasoning, E-contracts

April 2009

Table of Contents

1 Introduction

This report discusses work conducted within a broader project that is concerned with the development of an open (in the sense of Hewitt [33]) computational environment, populated by software agents, whose interactions are regulated by electronic agreements, i.e. e-contracts. Such interactions may concern business transactions, collaborative distributed problem solving, task and resource allocation problems etc. Agents in such environments will need to be able to monitor their interactions with other agents against the contracts that they are involved in, in order to determine what actions to perform and when. This means that an agent will need to establish at a given time point, the state of its transaction with other agents, in order to check and regulate its own behaviour with respect to the commitments it has assumed towards other agents, and plan its activities accordingly. Specifically, the agent will need to establish:

(i) *Factual information*, given a history of events that have occurred up to the point of its query. For instance, an agent for an e-commerce application may need to establish what facts are true of orders, payments, deliveries etc., that have occurred (who caused such events, when, whether they were carried out successfully and so on).

(ii) *Prescriptive information*, given a history of events that have occurred up to the point of its query. That is, an agent needs to know what obligations, permissions, prohibitions and legal powers are active for itself and each other agent in its environment.

To answer such queries some kind of temporal reasoning and reasoning about actions and their effects is required. Many researchers (for example, [46, 6, 20, 65] among others) have adopted Event Calculus (EC) [40] for contract representation. However, the historical information available to an agent at the time point it poses its query may be incomplete, for various reasons: Information may be lost, or distorted by noise, and in a truly open system, where agents join or leave the system at different times, information delivery from agent to agent may simply be delayed. In order to reason in the presence of incomplete historical knowledge, agents must be able to fill in information gaps, by employing assumptions about the past and the present time. When new information becomes available, possibly rendering some of these assumptions false, agents must be able to retract previously drawn conclusions, that is, conclusions inferred on the basis of these assumptions. In other words, agents need to reason nonmonotonically. Moreover, agents may need to establish potential future states of their transactions with other agents, i.e. to anticipate factual and prescriptive information, in order to plan their activities accordingly; in these cases, agents will need to be able to employ assumptions about the future. The problem of assumption-based reasoning has been explored by many researchers within the Artificial Intelligence community and the main distinction between their proposals concerns the nature of assumption identification. Static approaches to assumption identification are clearly inappropriate in the context of open systems, since it is unrealistic to expect that each agent developer will anticipate all potential situations in which the agent will find itself, in order to determine in advance potentially useful assumptions that the agent might use for its inference. Therefore, what is required is a dynamic ap-

proach to assumption identification. Finally, the identification of an appropriate assumption by an agent must be such that it is consistent with the agent's current definite knowledge about itself, the environment and the other agents that populate it. To establish whether a given assumption meets this requirement, one might be tempted to resort to proof. However, proof is notably computationally hard, and this is exacerbated by the fact that the environment is open, and hence the agent's current knowledge is subject to frequent changes.

Inspired by the syntax and semantics of Default Logic (DfL) [63], we present a way in which an agent, given a temporal representation of an electronic agreement, may construct all possible relations between what it knows and what it may assume, that are implicitly defined by the agreement. It turns out that this space is, in fact, a lattice, and at any particular time point the agent may position itself on it, given the *explicit* knowledge that it currently possesses, i.e. without resorting to proof. Once the agent has positioned itself on this lattice, it finds out what assumptions are related to the node it occupies and may employ them in its reasoning. As the agent's knowledge changes over time, and consequently as its assumption needs change, the agent repositions itself on the lattice by moving on it from node to node. We use the notion of consistency and extension derivation for Default Theories (DfT) provided by [5], which does not require logical proof. Nonmonotonic reasoning is achieved by the agent changing its position on the lattice.

In section 2 we introduce an e-contract example scenario and illustrate the need for hypothetical nonmonotonic reasoning in open computational environments, without reference to some specific temporal logic. In sections 3-5 we propose the representation of contract norms as default rules [63]. This representation may be constructed dynamically from an initial contract representation in some temporal logic, and discuss the ways in which this representation of contracts as Default Theories (DfT) is suitable [24, 26]. Section 6 discusses the full representation of an e-contract, specifically in Event Calculus. In section 7 we note the problems that are raised by previous approaches on e-contracting and hypothetical/nonmonotonic reasoning and discuss the technical aspects of our approach in more detail. Finally, section 8 summarizes our conclusions and directions for future research.

2 Preliminaries

2.1 Example Scenario

For the purposes of illustration consider a 3-party business transaction that takes place in an electronic marketplace populated by software agents. A buyer agent (BA) communicates with a seller agent (SA) and establishes an agreement with it for purchasing a certain product. Consequently, SA communicates with a carrier agent (CA) and establishes another agreement with it for the timely and safe delivery of goods to BA.

The first agreement (between BA and SA) is to be conducted on the following terms: SA should see to it that the goods be delivered to BA within 10 days from the date BA's order happens. BA, in turn, should see to it that payment be made either in cash on delivery or within 21 days from the date it receives the goods at an additional cost. The agreement may specify sanctions and possible reparations in case the two agents do not comply with their obligations, but we do not need to refer to them explicitly here. In the same spirit, the second agreement (between SA and CA) specifies obligations, deadlines and possible sanctions/reparations in case of violations.

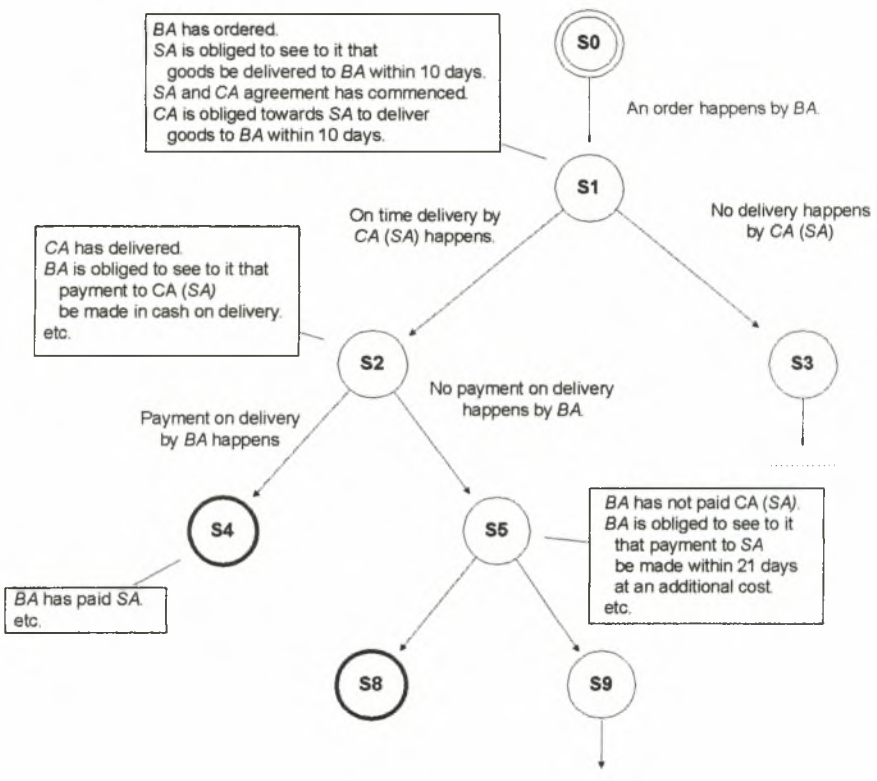

Figure 1 E-contract Transition Diagram

Following [14], we may take an informal, process view of the business transaction that is regulated by the two agreements. Each state offers a (possibly partial) description of the factual and normative propositions that hold true in it. A transition between states corresponds to an event that takes place, i.e., an action that one of the parties performs or omits to perform. Part of such a description of the business exchange as a state diagram is shown in Figure 1. Initially, at time point T0, the transaction is in state S0 where the two agreements have been established and no events have occurred yet. If BA places an order at some time after T0, the transaction will move to a

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

2.2 The need for assumptions in open environments

Note that reasoning in open environments with incomplete knowledge involves three issues of interest:

1H. What assumptions are applicable to fill in information gaps, i.e., what can be assumed by an agent to be true or false at different time points?

2H. What is the relationship between assumptions and new knowledge, i.e. how do the adopted assumptions affect future inferences, either enabling some or disabling others?

3H. What happens when new information becomes available, i.e., how does new information affect previously drawn conclusions?

To answer the first question (1H) the agent seeks to establish which norm conditions it knows about, and which it needs to assume. In an open environment, norm conditions may dynamically become known or unknown to the agent. Thus, we see that the first issue calls for a mechanism that enables agents to identify appropriate candidate assumptions dynamically. In order to answer the second question (2H) the agent needs to employ some way that commits its reasoning to specific assumptions. Finally, in order to answer the third question (3H) the agent needs to employ some kind of nonmonotonic reasoning. The need for reasoning nonmonotonically in legal domains is strongly argued in many research papers, i.e., [66, 18, 9, 32, 58] among others.

We can think of two reasons why it is useful for an agent to be able to reason hypothetically, by establishing appropriate assumptions dynamically. First, an agent can not know the future, yet it may need to plan its future activities on the basis of hypotheses that concern the future, i.e. on the assumption that certain events or other agents' actions will occur, or that certain causal relations will be effected in the environment, or that a certain normative relation (obligation, permission, prohibition, power) will obtain between itself and other agents. For example, suppose that BA orders from SA at time point T (T0<T<T1). A reasonable query that BA might have is "When will I, potentially, have to pay for this order, assuming all goes well and I receive the goods in due time, so that I plan to have adequate available funds?" To derive an answer BA needs to perform *best-guess reasoning* based on assumptions.

Second, an agent may not know everything about the past and present, i.e., the history of its environment, other agents and itself so far, yet it may need to plan its ac-

tivities on the basis of hypotheses that concern the past and present, i.e. on the assumption that certain events or other agents' actions have occurred, or that certain normative relations have obtained between itself and other agents. Consider the case where, at time point $T2$, BA does not yet know that CA has performed delivery, still it needs to plan its business activity so that it may be able to fulfill an obligation to pay SA in due time, should it later be informed that CA delivered at T' ($T1<T'<T2$). This situation corresponds to ***no-risk reasoning***, i.e., an agent should be able to derive a conclusion even though this is based on assumptions, because alternatively it might find itself in an undesirable situation. It is, therefore, clear that an agent should be able to establish potential conclusions on the basis of assumptions.

In the next sections we present a technique to reason nonmonotonically on the basis of assumptions that are identified and employed dynamically when needed. We believe that the ability of an agent to identify candidate assumptions dynamically is inherently related to the degree to which it is autonomous (cf. [29]), hence autonomous dynamic assumption identification is central to our proposal and sets it apart from other approaches to dynamic assumption-based reasoning. However, there are application domains where full, uncontrolled, agent autonomy is not appropriate. As we shall see shortly, our proposal affords ways to control assumption identification and deployment, should this be desirable.

3 e-Contracts as Default Theories

We propose a representation of e-contracts as default theories for various reasons: Default Logic is arguably the most notable formulation for default reasoning (cf. [5, 41]) and addresses general issues, such as negation by default, the frame problem and causal reasoning, satisfactorily [41]. Also, it is suitable for prototypical, no-risk, and best-guess reasoning, all of which interest us [5]. Moreover, among all approaches to default reasoning, such as Close World Assumption, Circumscription, Default Logic, Logic Programs and Defeasible Logic, the syntax of DfL affords an intuitive way to represent separately what is known from what is assumed during the inference process, and to relate a conclusion to the knowledge and assumptions used in its inference. This is due to the fact that DfL rule schemata comprise three distinct parts, representing prerequisites, assumptions and conclusions separately.

A default rule (henceforth default) has the form:

$$P : J1, J2, \ldots Jn / C,$$

where P is the ***prerequisite***, $J=\{J_1, J_2, \ldots J_n\}$ is a set of ***justifications***, and C is the derived ***consequent***. The semantics of this rule is: If P holds and the assumptions of J are consistent with the current knowledge, then C may be inferred. A DfT is a pair of the form (W, D), where W is a set of propositional or predicate logic formulae that represent currently available knowledge, and D is a set of defaults. A default is applicable to a deductively closed set of formulae $E \supseteq W$, iff $P \in E$ and $\neg J1 \notin E, \ldots, \neg Jn \notin E$. The set E is the extension of the DfT. The notion of extension is the most complicated concept of Reiter's DfT because it is hard to determine an accurate belief set for which justifica-

tions should be consistent. In Reiter's initial paper for DfL [63] three important properties of extensions are noted. Specifically, an extension E of a default theory (W, D):

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

The requirement that the extension of a default theory be deductively closed is computationally problematic. Therefore, we derive extensions in the manner presented by Antoniou in [5]. Antoniou proposed an operational definition of extensions and an incremental technique for their computation, by maintaining syntactically consistent sets of formulae, whose conditions part (prerequisites and justifications) is interpreted conjunctively and the conclusions part (consequent) is interpreted disjunctively, as in sequent calculus. Thus, an agent that derives conclusions on the basis of assumptions, by applying defaults, constructs the extension of its grounded DfT incrementally.

Let Π represent a default reasoning process by recording the order in which defaults from D apply. At each step i of the reasoning process, i.e. after the application of each default P:J1, J2,...,Jn/C, the extension computed is a set of ground sentences $In(i)=In(i-1) \cup \{C\}$, and the set of assumptions employed, which should not turn out to be true, is $Out(i) = Out(i-1) \cup \{\neg J1\ldots, \neg Jn\}$. As a result, $\Pi(i)= \Pi(i-1) \cup \{Di \mid Di$ is the default rule which applied at step i$\}$. Initially $In(0)=W$, $Out(0)=\varnothing$ and $\Pi(0)=\varnothing$ for $i=0$. The default reasoning process $\Pi(i)$ is ***successful*** iff $In(i) \cap Out(i)= \varnothing$, otherwise it is ***failed***. Moreover, the process $\Pi(i)$ is *closed* iff every default rule that belongs in the D set and is applicable in $In(i)$ already occurs in $\Pi(i)$. According to [5] a set of formulae E is a DfT extension, if there is a closed and successful process $\Pi(i)$ of the DfT such that $E=In(i)$.

For illustration purposes, consider the default theory (W, D), with $W=\{A\}$ and D containing the following defaults:

D1 = A : B / C

D2 = true : ¬D / E

The process $\Pi(2)=\{D1,D2\}$, i.e. $In(2)=\{A,C,E\}$ and $Out(2)=\{\neg B,D\}$, is successful and closed and thus it is considered as an extension of the theory.

Now, consider the default theory (W, D), with $W=\{A\}$ and D containing the following defaults:

D1 = A : B / C

D2 = true : D / ¬B

The process $\Pi(2)=\{D1,D2\}$, i.e. $In(2)=\{A,C,\neg B\}$ and $Out(2)=\{\neg B, \neg D\}$, is closed but not successful and thus it is not considered as an extension of the theory. The process $\Pi(1)=\{D2\}$, i.e. $In(1)=\{A,\neg B\}$ and $Out(1)=\{\neg D\}$, is successful and closed, since D1 does not apply, and thus it is considered as an extension of the theory.

Hence, DfL affords a natural means to relate a conclusion to the grounds that support its inference (in the sense of argumentation [57]), that is, to the knowledge and

assumptions that are used in order to derive it. This is particularly useful in the context of questions (2H)-(3H): Extension derivation reflects the ways in which an inference made at some time point affects subsequent inferences; in the case of non-monotonic reasoning, should some assumption turn out to be false, it is possible to trace any inferences made on its basis and retract them.

For the purposes of simplicity consider an open norm-governed environment populated by agents, in which some temporal logic is employed, for instance Event Calculus [40][1]. Initially e-contract norm are represented as sentences of the form:

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

where Y and X_i $(1 \leq i \leq k)$ are first-order logic positive or negative literals and all variables are universally quantified. This representation employs the predicates of the temporal logic augmented with some special predicates to denote normative relations (obligation, prohibition, permission, power). We view normative relations as properties that are initiated or terminated by the occurrence of agents' actions or events. The norms that can be expressed in such a representation take the form, for example "agent Agent2 is obliged/permitted/prohibited towards agent Agent1 to perform action Action2 by time Time2, if agent Agent1 performs action Action1, at time Time1". The initial representation of an e-contract may be characterized as a triple (H, R, A). H corresponds to historical information and is a possibly empty or incomplete set of domain-dependent definitions for currently available information, i.e. H is a set of propositional or predicate formulae representing events that have occurred and facts that holds. R corresponds to domain-dependent causal information and is a possibly empty or incomplete set of sentences of the form (1). A is a non-empty set of sentences of the form (1) expressing the domain–independent knowledge [24, 26].

A contract norm of the form (1) may be mapped to *any one* of the following default rules [24]:

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

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

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

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

...

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

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

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

...

[1] In section 6 we see the full contract representation in Event Calculus. Here, we discuss the issues that interest us, without reference to any specific temporal logic.

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

...

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

That is, each initial norm that involves k conditions may be mapped to any one of 2^k+1 defaults. The question that arises for the agent that seeks to establish autonomously which assumptions are appropriate in order to fill in information gaps is tantamount to the question "which one of the 2^k+1 defaults should be chosen and employed in the inference procedure", for each of the norms in the initial set of norms.

An e-contract can be represented as a DfT $\equiv (W, D)$ by translating/reformulating its initial representation. For an agent constructing the DfT, the question that arises is which of the 2^k+1 defaults should be chosen and employed subsequently during its inference procedure. The construction of W and D sets is carried out as follows [26]:

The currently available knowledge W is constructed from the domain-specific part of the initial contract representation. Specifically, W is a copy of H, the possibly empty or incomplete historical information of the initial contract representation which contains all currently available knowledge about what holds and what happened.

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

The formal characterization of the construction of the DfT is as follows: An e-contract is the pair (W, D), where $W = H$ and D contains, for each definition $(Y \leftarrow X_1 \wedge \ldots \wedge X_k) \in A \cup R$, (possibly semi-grounded) defaults of the form $P_1 \wedge \ldots \wedge P_m : J_1, J_2, \ldots J_n / C$, such that $m+n=k$ and $P_i = SUBST(\theta, X_i)$ if $H \cup R \vdash SUBST(\theta, X_i)$, $J_i = SUBST(\theta, X_j)$ if $H \cup R \nvdash SUBST(\theta, X_j)$, and finally $C = SUBST(\theta, Y)$.

One may argue that the DfT construction in this way, suggests that an agent may get trapped in an endless procedure trying to prove formulae from its knowledge base, in order to decide whether to use these formulae in the prerequisite or the justification part of each default rule; in other words, the agent needs to attempt to prove formulae (and fail in doing so) in order to decide which of these are candidate assumptions. In order to overcome this limitation, we now describe a computationally viable procedure by which an agent may also determine assumptions dynamically and consequently construct the DfT. This technique does not require the agent to attempt to prove formulae from its current knowledge base, and therefore, is suitable for implementation [25, 27].

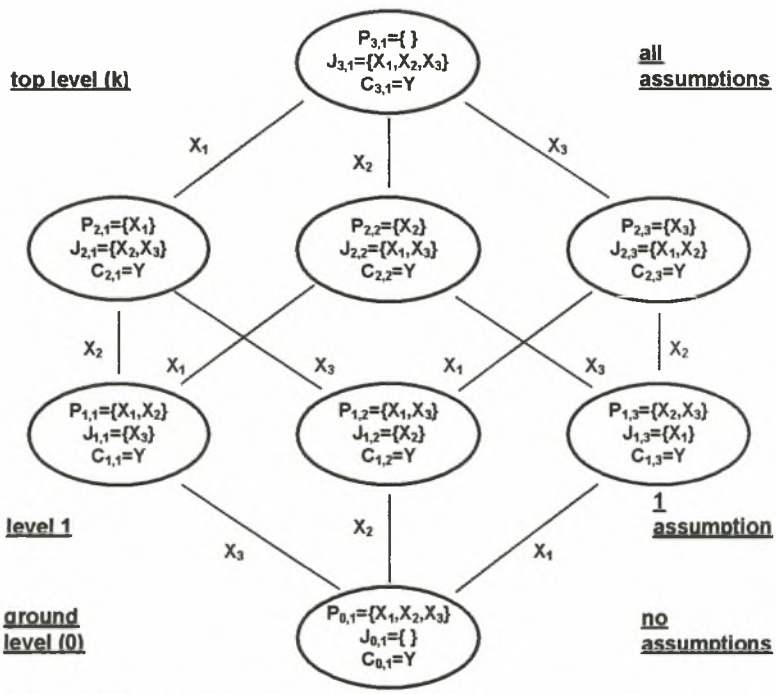

Figure 2 Expansion/contraction of assumption space and knowledge base

One may think of the 2^k possible defaults for one contract rule as organized in a lattice structure of height $(k+1)$ such as the one shown in Figure 2[2]. Each level of this structure contains one or more of the 2^k defaults, depending on the number of assumptions that these defaults employ, i.e. the binary relation that causes them to be partially ordered is the number of assumptions employed. Level 0 (the ground level) contains the single, justification-free, default, level 1 contains the k one-justification defaults, and so on, until the top level (the penthouse) which contains the single, prerequisite-free, default [27].

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

[2] For the moment we omit the normal default rule. We discuss normal defaults separately in section 4.1.

level (i-1) that $P_{i-1} = P_i \cup \{X_j \mid X_j \text{ is retracted}\}$ and $J_{i-1} = J_i - \{X_j \mid X_j \text{ is retracted}\}$, where $1 \le j \le k$ and $0 < i \le m$. If $m=k$, then $P_k = \varnothing$ and $J_k = J$.

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

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

The corresponding lattice structure of height 4 is shown in Figure 2 where each level contains the following defaults[3]:

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

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

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

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

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

In Figure 2, the assumption space expands, and the corresponding knowledge base contracts, when the agent moves upwards in the lattice structure. Conversely, the assumption space contracts and the corresponding knowledge base expands, when the agent moves downwards.

Of course, contracts (and normative systems in general) contain multiple rules, each of which may be formulated as a default, and candidate default formulations are organized in a structure, such as the one described above. Hence, the DfT e-contract representation is a pair of the form (W, D), where W is considered as already shown, and D contains the set lattices, each containing the possible formulations of a contract rule as a default. Note that, although the corresponding rule mapping is *one-to-many*, only one default for each initial contract rule may finally be employed for inference.

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

The inference process starts from the ground level, by applying as many defaults as possible given the agent's current knowledge. Each time a default applies its conclusions are included in the current extension that is being computed. When there are no further defaults that can be applied in a level, this signals that further assumptions are needed in order to proceed and inference continues by examining defaults that lie in the next level upwards. Note that the case where reasoning is possible using only

[3] Note that, in the sequent calculus setting, the comma to the prerequisites part of the default should be thought of as an "and".

rules from the ground level is identical to inference in classical logic, but here we are also able to preserve consistency of entailment if we accept variations of DfL, such as Constrained Default Logic [67].

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

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

Thus, the corresponding lattices levels contain the defaults:

Lattice for R1:

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

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

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

Lattice for R2:

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

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

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

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

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

- if $W = In(0) = \{X_1, X_2\}$ and $Out(0) = \varnothing$ then extension $In(2) = \{ X_1, X_2, , Y_1, Y_2\}$ is computed by making the assumption that X_3, X_4 and X_5 hold ($Out(2) = \{\neg X_3, \neg X_4, \neg X_5\}$) and by applying defaults $D1_{0,1}$ and $D2_{3,1}$ respectively.
- if $W = In(0) = \{X_1, X_2, X_3\}$ and $Out(0) = \varnothing$ then extension $In(2) = \{ X_1, X_2, X_3, Y_1, Y_2\}$ is computed by making the assumption that X_4 and X_5 hold ($Out(2) = \{\neg X_4, \neg X_5\}$) and by applying defaults $D1_{0,1}$ and $D2_{2,1}$ respectively.
- if $W = In(0) = \{X_1, X_3, X_4, X_5\}$ and $Out(0) = \varnothing$ then extension $In(2) = \{ X_1, X_3, X_4, X_5, Y_1, Y_2\}$ is computed by making the assumption that only X_2 holds ($Out(2) = \{\neg X_2\}$) and by applying defaults $D2_{0,1}$ and $D1_{1,1}$ respectively.

Note that although a level may contain two or more defaults that correspond to the same initial contract rule (e.g. $D2_{1,1}$ or $D2_{1,2}$ or $D2_{1,3}$) there is no need for some kind of prioritization amongst them. If two or more defaults of the same level, which are derived from the same initial rule, were to apply simultaneously, then the more general default contained in the immediately lower level should have applied. Thus, the dilemma of the form which default to apply (e.g. $D2_{1,1}$ or $D2_{1,2}$ or $D2_{1,3}$) signals an error in the reasoning procedure and that rule $D2_{0,1}$ should have applied first. Moreover, be-

cause inference involves one lattice level at a time in a step-wise manner, the agent employs the fewest possible hypotheses.

4 Discussion

4.1 Normal Default Rules

So far, we have omitted normal defaults from the discussion about the way in which an agent may construct its default theory. Normal defaults have the form $P:C/C$, i.e., their justification coincides with their consequent. Two questions seem to arise naturally [27]:

(i) Should the agent include normal defaults in the set of potential mappings that it constructs from the initial e-contract representation? And, if so,

(ii) In which level of the lattice should normal defaults be placed?

It seems to us that normal defaults are required only in order to ensure that there is at least one extension of the knowledge that the agent possesses, which may be computed by assimilating new information, provided that consistency is preserved. The normal default may be viewed syntactically as belonging to level 1 of the lattice, in that it involves one assumption, yet this assumption is different in nature from the other ones, i.e. the normal default is different semantically from the other level 1 defaults. That is, the normal default may be viewed as behaving similarly to the justification-free default, in that all its prerequisites should be satisfied by the current knowledge base; the only additional assumption made in the case of the normal default concerns the consistency of its conclusion with the current knowledge base. For this reason, although the normal default contains a single assumption, and should therefore belong to level 1 of the lattice, 'operationally' it belongs to level 0, since its assumption is not genuinely about something that holds in the world.

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

4.2 Sceptical/Credulous reasoning

DfL enables us to reason with incomplete knowledge, by deriving conclusions that are based on consistent assumptions, which may be retracted later, in the presence of new information. There are two approaches to performing such inference. In the first

one, the *sceptical reasoning mode*, a formula is entailed by a default theory, if it is derived by all its extensions. This is a strict approach and requires the computation of all possible extensions, and subsequent check to determine if a formula belongs to all of them. This mode of reasoning is useful when an agent needs to plan its future activities at some time point, on the basis of hypothetical information about the history so far. In the second approach, the *credulous reasoning mode*, a formula is entailed by a default theory, if it is derived by at least one extension. Such an approach is useful in order to support the agent's decision making when norm violation or conflicting obligations arise. To perform either sceptical or credulous reasoning, we need essentially to compute all possible extensions of a default theory. However, this is not computationally viable, and so we adopt the operational definition of extensions of [5], as explained above, which produces incrementally the extensions organized in a tree structure. To perform sceptical and credulous reasoning on this structure is tantamount to checking the offsprings of a node.

4.3 Managing the Space of Hypotheses

So far, we have argued that agents must resort to assumptions in order to reason in the presence of incomplete knowledge, and we have shown a way in which they may be able to identify candidate assumptions and employ them dynamically. We have shown that the space of possible hypotheses available to an agent is essentially infinite, since it treats any literal that it does not know about *explicitly* as a candidate assumption. Other approaches to dynamic assumption-based reasoning (eg [13, 44, 12, 60, 61, 1, 52, 53, 49, 34, 69]) rely on a finite hypotheses space, which is either pre-specified (usually in this case it is referred to as *assumption pool*) and is explored dynamically, or is identified dynamically by goal-driven generation, i.e. the agent has specific conclusions that it wants to derive and identifies what assumptions are required in order to perform its derivations. Our approach enables agents to be more 'independent' and 'open-minded' (one might say more like humans). At the same time, though, this means that we need to make provision for the management of this infinite space of assumptions.

There are three aspects to managing the space of hypotheses:

(i) How do we ensure that the set of hypotheses that an agent uses to draw a conclusion is consistent, and that the possible world models that the agent infers are rational?

(ii) How do we restrict the space of hypotheses, when this is imperative, due to constraints that arise from a particular application domain?

(iii) Is the order in which an agent identifies and employs hypotheses important, i.e. does it affect what conclusions it may draw, whether these are rational, and the extent to which the agent bases its reasoning on reality rather than wishful thinking?

We discuss each of these issues in turn, now.

4.3.1 Hypotheses Rationality and Consistency

One may argue that following Reiter's original computation of extensions within DfT we may compute possible world models that are counter-intuitive: for instance, in our example above, BA would infer an extension which suggests that BA infers a possible version of the world, in which it bears an obligation to pay SA, although no delivery from SA is explicitly recorded in this world, and similarly that SA bears an obligation to deliver, although this world does not explicitly record that BA's order is valid (*best-guess reasoning*). This view of extensions, separated from assumptions, as possible world models, is clearly undesirable. To overcome this problem we may employ Constrained Default Logic (CDfL) [67] and require joint consistency of default assumptions. The possible world model that the agent infers incrementally, for a Constrained Default Theory (CDfT), is the consistent set $In(i) \cup \neg Out(i)$. This is tantamount to saying that the possible world models inferred by the agent contain, besides previous knowledge, both the consequents and the assumptions of the applied defaults.

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

4.3.2 Hypotheses Restriction

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

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

omy. Here are some possible scenarios, for the example discussed in section 3 with the same initial knowledge (W) but with different initializations of the Out set.

- if $W = In(0) = \{X_1, X_2\}$ and $Out(0) = \{Y_2\}$ then extension $In(1) = \{X_1, X_2, Y_1\}$ is computed by applying only $D1_{0,1}$ default ($Out(1) = \{Y_2\}$). No assumptions are possible due to the initial restriction on what can be inferred.
- if $W = \{X_1, X_2, X_3\}$ and $Out(0) = \{X_4, X_5\}$ then extension $In(1) = \{X_1, X_2, X_3, Y_1\}$ is computed by applying only $D1_{0,1}$ default ($Out(1) = \{X_4, X_5\}$). Again no assumptions are possible due to the initial restriction on what can be assumed.
- if $W = \{X_1, X_3, X_4, X_6\}$ and $Out(0) = \{X_2, Y_2\}$ then no extension is computed due to the initial restriction on what can be inferred and assumed.

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

The next question that arises is "How must we initialize the Out(i) set?". We may use a Preconstrained Default Theory (PcDfT). A PcDfT is a triple of the form (W, D, PC), where (W, D) is a CDfT and PC is set of formulae that are considered as the constraints of the theory [68]. For the first step of the process, i.e., for i=1, In(0)=W and Out(0) = PC. It is clear that, the role of preconstraints is identical to the role of the Out(i) set in initializing and adjusting agents' hypotheses. By initializing the PC set appropriately, an agent may specify and apply a certain strategy in its reasoning. Note that the formulae contained in PC define what the agent concedes or expects, and not factual knowledge. For this reason it is separated from W.

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

4.3.3 Hypotheses Sequence

A question that arises during assumption-based reasoning is what is the reasonable sequence for employing assumptions? We believe that this question and its answer are strongly related to causality.

We may use stratification for a Default Theory (DfT) [10, 11] or for its two variants (CDfT and PcDfT) [4]. A DfT is called stratified (SDfT) iff there exists a stratification function s that assigns a natural number to each default. As a result of the application of a stratification function the set of default rules is ordered into strata. If the consequent of a rule D is used by another rule D' then we apply D before D' i.e.,

$s(D) \leq s(D')$. The formal characterization of this property for any defaults D and D' is as follows:

if $prop(cons(D)) \cap prop(pre(D') \cup just(D')) \neq \varnothing$ then $s(D) \leq s(D')$

if $prop(cons(D)) \cap prop(cons(D')) \neq \varnothing$ then $s(D) = s(D')$

where $pre(D)$, $just(D)$ and $cons(D)$ denote the prerequisites, justifications and consequents of D, respectively, and $prop(D)$ denotes the set of all propositional atoms occurring in D [4]. According to this definition D and D' are mapped into different strata because $s(D) \leq s(D')$ holds. In this way, an agent ensures that D applies before D' and the causal relation between the two rules is not lost.

Note that the way in which agents construct lattice structures, as per our proposal, resembles, in a way, stratification of a DfT [10]. The possible default formulations of each initial norm are assigned to the various lattice levels, depending on the number of assumptions that each default formulation employs. That is, the number of assumptions may be regarded as somewhat similar to a stratification criterion applied to the set of possible default formulations for each initial norm.

The question that arises naturally, now, is how these two distinct ordering methods (one due to stratification and one due to the number of assumptions) relate to each other. Stratification aims at preserving causal relations between defaults, while the organization of defaults into lattice structures aims at ensuring that agents employ the fewest possible hypotheses at each step of their reasoning process, and thus base their conclusions on facts as much as possible, rather than on assumptions. The set of lattices that the agent possesses may be subjected to stratification, so that the agent chooses a reasonable order in which to apply default rules, and preserve any causal relations between defaults. Once a particular lattice, belonging to a particular stratum, is chosen, the agent establishes which node of the lattice corresponds to its current knowledge base (and therefore assumption requirements). Note that the agent may use different levels of different lattices. The precise level to be used in each lattice is determined by its current knowledge. The precise lattice to use at each point is determined by the stratification function. In this way an agent infers some knowledge, even on a (partially or totally) hypothetical basis, which causes the entailment of other knowledge in an argumentation-like manner (cf [57]) and we may characterize its conclusions in the same way that is used to characterize arguments (e.g. defeasible). To illustrate this, let us assume the previous normative system containing the norms $R1$ and $R2$ enhanced with two additional norms $R3$ and $R4$ of the form:

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

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

Lattice for $R3$:

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

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

Lattice for $R4$:

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

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

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

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

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

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

5 Examples

So far in our discussion, we have used abstract examples, in order to facilitate focusing on concepts rather than the particulars of a specific domain of application. Here we illustrate the points raised in the preceding discussion, with reference to the e-commerce example presented in section 2.1, although it is, of course, more generally applicable to other open multi-agent scenarios, where agents have to reason with incomplete or inconsistent knowledge and their behaviour is regulated by some norms (e.g. cooperative distributed problem solving, task allocation etc).

5.1 Dealing with information gaps (1H)

Single-agent Autonomous Reasoning

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

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

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

DeliversTo(agent3,agent1)

∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

}

Note that these norms are syntactically identical to the norms considered in the example presented in section 3. Thus, the corresponding lattices of candidate default formulations for each one coincide with the lattices shown in section 3, i.e. they are as follows:

Lattice for R1:

Level 0: { $D1_{0,1}$ ≡

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

Level 1: { $D1_{1,1}$ ≡

OrdersFrom(agent1,agent2) : E-shopFunctionsWell / IsObligedToDelivern(agent2,agent1),

$D1_{1,2}$ ≡

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

Level 2: { $D1_{2,1}$ ≡

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

Lattice for R2:

Level 0: { $D2_{0,1}$ ≡

OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: true

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 1: { $D2_{1,1}$ ≡

OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,2} \equiv$

OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: DeliversTo(agent3,agent1)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,3} \equiv$

DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: OrdersFrom(agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 2: { $D2_{2,1} \equiv$

OrdersFrom(agent1,agent2)

: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,2} \equiv$

DeliversTo(agent3,agent1)

: OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,3} \equiv$

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 3: { $D2_{3,1}$ ≡

true

: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1).

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

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

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

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

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

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

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

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

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

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

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

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

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

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

}

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

Multi-agent Autonomous Reasoning

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

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

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

DeliversTo(agent3,agent1) ∧

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2),

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

Pays(agent1,agent3) ∧

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

}

The corresponding lattices are as follows:

Lattice for R1:

Level 0: { $D1_{0,1}$ ≡

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

Level 1: { $D1_{1,1}$ ≡

OrdersFrom(agent1,agent2) : E-shopFunctionsWell / IsObligedToDelivern(agent2,agent1),

$D1_{1,2}$ ≡

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

Level 2: { $D1_{2,1}$ ≡

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

Lattice for R2:

Level 0: { $D2_{0,1}$ ≡

OrdersFrom(agent1,agent2) , DeliversTo(agent3,agent1),

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: true

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 1: { $D2_{1,1}$ ≡

OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)

: IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,2}$ ≡

OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: DeliversTo(agent3,agent1)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{1,3} \equiv$

DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: OrdersFrom(agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 2: { $D2_{2,1} \equiv$

OrdersFrom(agent1,agent2)

: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,2} \equiv$

DeliversTo(agent3,agent1)

: OrdersFrom(agent1,agent2), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2),

$D2_{2,3} \equiv$

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Level 3: { $D2_{3,1} \equiv$

true

: OrdersFrom(agent1,agent2), DeliversTo(agent3,agent1),

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToPayOnBehalfOf(agent1,agent3,agent2) }

Lattice for R3:

Level 0: { $D3_{0,1} \equiv$

DeliversTo(agent3,agent1), Pays(agent1,agent3))

∧ IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: true

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 1: { $D3_{1,1} \equiv$

DeliversTo(agent3,agent1), Pays(agent1,agent3)

: IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{1,2} \equiv$

DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: Pays(agent1,agent3)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{1,3} \equiv$

Pays(agent1,agent3), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: DeliversTo(agent3,agent1)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 2: { $D3_{2,1} \equiv$

DeliversTo(agent3,agent1)

: Pays(agent1,agent3), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D32_{2,2} \equiv$

Pays(agent1,agent3)

: DeliversTo(agent3,agent1), IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1),

$D3_{2,3} \equiv$

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2)

: DeliversTo(agent3,agent1), Pays(agent1,agent3)

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

Level 3: { $D3_{3,1} \equiv$

true

: DeliversTo(agent3,agent1), Pays(agent1,agent3),

IsEmpoweredToAcceptPaymentFromOnBehalfOf(agent3,agent1,agent2))

/ IsObligedToAcceptPaymenFromOnBehalfOf(agent2, agent3,agent1) }

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

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

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

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

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

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

IsEmpoweredToAcceptPaymentFromOnBehalfOf(CA,BA,SA)

}

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

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

5.2 Commitment to Assumptions (2H)

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

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

in order to infer its obligation to pay, BA needs to assume that CA is empowered to accept payment on behalf of SA, i.e. IsEmpoweredToAcceptPaymentFromOnBehalfOf(CA,BA,SA). This assumption, if employed at some time point, affects BA's subsequent inferences. For example, in order to infer SA's obligation to accept its payment via its representative (CA) as norm R3 states, BA only needs either to actually perform payment (Pays(BA, CA) will be rendered true in its knowledge base) or to assume it, because it is still committed to its previous assumption about CA's legal power. In other words, BA either needs to use the default formulation $D3_{0,1}$ or the default formulation $D3_{2,1}$, from the lattice corresponding to norm R3.

5.3 Nonmonotonic Reasoning (3H)

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

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

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

}

If the initial knowledge for agent BA is:

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

then BA may infer its obligation to pay (default formulation $D2_{1,1}$) or SA's obligation to accept its payment (default formulation $D3_{2,1}$) on the basis of the assumption that CA is

empowered to accept payment on behalf of SA (the conditions of the constraints contained in the Out set are not satisfied).

Now imagine that at a later time point BA is informed that SA is indeed a debtor, i.e. Debtor(SA) is added to its current knowledge base. This new information affects its previously drawn conclusion. Now, BA needs to traverse the lattices downwards in order to choose an alternative formulation for the norms compatible with its current knowledge and any restrictions imposed on the assumption space. For example, BA had previously used the formulation $D2_{1,1}$ for norm R2 and the formulation $D3_{2,1}$ for norm R3. Now that its knowledge base has expanded and contains new information, it traverses the corresponding lattices downwards from these nodes and reaches the formulations that lie at level 0 in each lattice, i.e. it retracts previously employed assumptions, along with any conclusions drawn on their basis. At this level, BA will not be able to continue the incremental computation of extensions.

6 The representation of e-Contracts in Event Calculus

To establish the state of a business exchange, given a history of parties' actions, we may represent the agreement that regulates this exchange in some temporal logic, as we mentioned in the Introduction. In fact, such representations have been constructed for various types of agreements by many other researchers in Event Calculus (e.g. [46, 6, 20, 65] among others). The basic elements of the language are *time points*, *fluents* and *actions* or *events*. Fluents are factual and normative propositions whose truth-value alters over time, as a result of the occurrence of an action or an event.

For the example scenario that we introduced earlier, we adapted the simple EC formalism presented in [48]. In its original form, the formalism does not distinguish between events that are brought about through agents' actions, and *force majeure* events that are brought about independently of the agents. We preserve the distinction and use the term 'action' to refer to the former, and 'event' to refer to the latter. A more detailed discussion of the representation that we constructed may be found in [24, 26]. Here, we present the main points briefly, to offer the reader an idea of what such representations look like, and to motivate the ensuing discussion about the limitations that we perceive in such representations.

We use terms, such as Order(agent1, agent2), for fluents that become true as a result of specific actions (here ordering AOrder(agent1, agent2)). We use terms of the form Op(agent1, agent2, action, time) for fluents that describe normative propositions and their intended reading is "agent1 is in legal relation Op towards agent2 to perform action by time". The legal relation Op may be obligation, prohibition or permission; although these notions are typically formalized in some system of Deontic Logic, we merely use them as descriptive names for fluents, and do not adopt any specific Deontic Logic axiomatization.

As [45, 6] note, the effects of an action apply only when the action is considered valid, and this, in turn depends on whether its agent has the legal and practical ability to perform it. An agent's legal and practical ability with respect to certain actions may be time-dependent, so we use the fluents IPower(agent, action) and PAbility(agent, action) respectively, and the fluent Valid(agent, action) to denote that an action performed by an

agent is valid. We employ the six basic predicates of [48], shown in Table 1; of those, Initiates and Terminates are used along with Happens in the specific description of a particular contract, to represent causal relations between fluents and actions/events. The other three are defined in a domain-independent manner. We have modified the original definition of the HoldsAt predicate to take into account action validity, and have, consequently, extended the Happens predicate to include the agent of an action as an argument (for events, though, we use the original form of Happens).

Table 1 Basic Event Calculus Predicates

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

For illustration purposes, some domain-independent definitions are shown below:

Clipped(time1, fluent, time2) ← (Happens(agent, action, time) ∧ Terminates(action, fluent, time)
∧ time1≤ time<time2 ∧ HoldsAt(Valid(agent, action), time))

Declipped(time1, fluent, time2) ← (Happens(agent, action, time) ∧ Initiates(action, fluent, time)
∧ time1≤ time<time2∧ HoldsAt(Valid(agent, action), time))

HoldsAt(fluent, time2) ← (Happens(agent, action, time1) ∧ Initiates(action, fluent, time1) ∧ time1<time2
∧ ¬Clipped(time1, fluent, time2) ∧ HoldsAt(Valid(agent, action), time1))

¬HoldsAt(fluent, time2) ← (Happens(agent, action, time1) ∧ Terminates(action, fluent, time1) ∧ time1<time2
∧ ¬Declipped(time1, fluent, time2) ∧ HoldsAt(Valid(agent, action), time1))

HoldsAt(fluent, time2) ← (HoldsAt(fluent, time1) ∧ time1<time2 ∧ ¬Clipped(time1, fluent, time2))

¬HoldsAt(fluent, time2) ← (HoldsAt(fluent, time1) ∧ time1<time2 ∧ ¬Declipped(time1, fluent, time2))

Note that the first definition for HoldsAt above reflects the establishment of a fluent as a result of an action, while the second one reflects the common sense law of inertia. We do not show here the definitions of all the predicates with respect to event

(rather than action) occurrence, but a reader familiar with EC may see what form these take easily.

As stated in section 3, the EC representation of an e-contract may be characterized as a triple (H, R, A). Specifically, H corresponds to historical information and is a (possibly empty/incomplete) set of definitions for predicates H_L = {Happens, Holds}, R corresponds to causal information and is a (possibly empty/incomplete) set of definitions for R_L = {Initiates, Terminates}, and A is the (non-empty) set of definitions for the domain–independent predicates A_L = {HoldsAt, ¬HoldsAt, Clipped, Declipped}, that is, $A = \{Y \leftarrow X_1 \wedge \ldots \wedge X_k \mid Y \in A_L \text{ and } X_i \in A_L \cup H_L \cup R_L \cup T_L\}$, where T_L contains the first-order-logic predicates used to express temporal relations, i.e., $T_L = \{<, =, >, \geq, \leq\}$. So, the complete language is $H_L \cup R_L \cup A_L \cup T_L$.

6.1 Examples

Let us return to our example scenario, and consider that BA orders from SA at time point T. Here is an extract of the H, R and A sets of the EC representation for the agreement between BA and SA. Note that this information may be incomplete, i.e., an agent may possess only partial historical knowledge (here, BA knows that it ordered from SA at time point T) and partial causal knowledge (here, BA knows that placing an order imposes an obligation on the recipient of the order to deliver; it knows that this obligation is terminated/discharged successfully when delivery actually takes place; and it also knows that the occurrence of delivery imposes an obligation on itself for payment, which is terminated when payment is actually made):

H^{BA} = { Happens(BA, AOrder(BA, SA), T) }

R^{BA} = {

R1 ≡

Initiates(AOrder(agent1,agent2),Obligation(agent2,agent1,ADelivery(agent2,agent1), time1+10),time1) ←

Happens(agent1, AOrder(agent1, agent2), time1)

R2 ≡

Initiates(ADelivery(agent1,agent2),Obligation(agent2,agent1,APayment(agent2,agent1),time1), time1) ←

(Happens(agent1, ADelivery(agent1, agent2), time1)

HoldsAt(Obligation(agent1, agent1, ADelivery(agent1, agent2), time2), time1)

time1≤ time2)

R3 ≡

Terminates(ADelivery(agent1,agent2),Obligation(agent1,agent2,ADelivery(agent1,agent2),time2),time1) ←

(Happens(agent1, ADelivery(agent1, agent2), time1)

HoldsAt(Obligation(agent1, agent2, ADelivery(agent1, agent2), time2), time1)

time1≤ time2)

R4 ≡

Terminates(APayment(agent1,agent2),Obligation(agent1,agent2,APayment(agent1,agent2),time2),time) ←

(Happens(agent1, APayment(agent1, agent2), time1)

HoldsAt(Obligation(agent1, agent2, APayment(agent1, agent2), time2), time1)

time1≤ time2)

}

A^{BA} = {

A1 ≡

HoldsAt(Obligation(agent2, agent1, ADelivery(agent2, agent1), time1+10), time2) ←

Happens(agent1, AOrder(agent1, agent2), time1)

Initiates(AOrder(agent1, agent2),Obligation(agent1, agent1,ADelivery(agent2, agent1),time1+10),time1)

¬Clipped(time1, Obligation(agent2, agent1, ADelivery(agent2, agent1), time1+10), time2)

HoldsAt(Valid(agent1, AOrder(agent1, agent2)), time1)

time1<time2

A2 ≡

HoldsAt(Obligation(agent2, agent1, APayment(agent2, agent1), time1+21), time2) ←

Happens(agent1, ADelivery(agent1, agent2), time1)

Initiates(ADelivery(agent1, agent2),Obligation(agent2, agent1,APayment(agent2, agent1),time1),time1)

¬Clipped(time1, Obligation(agent2, agent1, APayment(agent2, agent1), time1+21), time2)

HoldsAt(Valid(agent1, ADelivery(agent1, agent2)), time1)

time1<time2

}

With reference to this representation, and given BA's current knowledge, only rule R1 may actually be used for inference, since its conditions are satisfied, and so BA may only infer that:

Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T)

But what if BA needs to perform *best-guess* or *non-risk* reasoning? In this case BA needs to identify rule conditions that it may use as assumptions, and we proposed that this is possible, if the initial set of contract rules are reformulated as default rules. Since many different formulations are possible for each contract rule, the agent need not commit (statically) to some specific one, and instead it may construct the lattice of possible default formulations, as we argued earlier. In this way, the agent will be able to use any of the possible formulations, depending on its currently available knowledge, which changes over time; essentially the agent will be identifying candidate assumptions dynamically.

As a result the agent constructs the following DfT (W^{BA} , D^{BA}):

W^{BA} = { Happens(BA, AOrder(BA, SA), T) },

that is, W^{BA} contains the historical information available to the agent, and D^{BA} is the set containing the corresponding lattices of default formulations for each rule contained in the R^{BA} and A^{BA} sets.

Now, BA is able to perform both *no-risk* and *best-guess* reasoning by employing some of these defaults. For example, in the absence of information to the contrary, it may assume that its order is a valid action and that SA's obligation to deliver is not unexpectedly terminated, in order to infer that SA bears an obligation to deliver the ordered goods. BA may come to this conclusion by employing in its inference the defaults DR1 and DA1, respectively, and by computing the In and Out sets as shown below[4]:

DR1 ≡

$$\frac{\text{Happens(BA, AOrder(BA,SA), T)} \quad : \quad \text{true}}{\text{Initiates(AOrder(BA, SA), Obligation(SA,BA,ADelivery(SA,BA), T+10), T)}}$$

[4] Note that in our example time is discrete. So agents may generate past or future time points in order to make their assumptions. The only requirement for agents when assuming the existence of time points is to position each new time point in the overall time sequence, by introducing their temporal relation to other, known or assumed, time points.

DA1 ≡

$$\frac{\begin{array}{l}\text{Happens(BA, AOrder(BA, SA), T),}\\ \text{Initiates(AOrder(BA,SA),}\\ \text{Obligation(SA,BA,ADelivery(SA,BA),T+10),T)}\\ :\\ \neg\text{Clipped(T, Obligation(SA, BA, ADelivery(SA, BA), T+10),T1),}\\ \text{HoldsAt(Valid(BA, AOrder(BA, SA)), T), T < T1}\end{array}}{\text{HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T1)}}$$

$In(2)^{BA} = W^{BA} \cup \{$ Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T),

HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10),T1),

$\}$

$Out(2)^{BA} = W^{BA} \cup \{$ Clipped(T, Obligation(SA, BA, ADelivery(SA, BA), T+10), T1),

$\neg$ HoldsAt(Valid(BA, AOrder(BA, SA)), T), $\neg$ (T< T1)

$\}$

In the same spirit, and on the assumptions that: SA's delivery will happen at some time point; such delivery will be valid; the effect of such delivery will be an obligation for BA to pay; and, finally, that such obligation will not be terminated by some other action, BA may infer what its potential payment period will be, relative to the time point of its assumptions. BA may come to this conclusion by employing in its inference the defaults DR2 and DA2, respectively, and by computing the In and Out sets as shown below[5]:

DR2 ≡

$$\frac{\begin{array}{l}\text{true}\\ :\\ \text{Happens(SA, ADelivery(SA, BA), T'),}\\ \text{HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T'), T' ≤ T'}\end{array}}{\text{Initiates(ADelivery(SA,BA),Obligation(BA,SA,APayment(BA,SA), T'), T')}}$$

[5] Note that the use of time point T' is possible under the assumptions that T1< T' < T2 and T' ≤ T+10.

DA2 ≡

$$\frac{\begin{array}{c}\text{true}\\ :\\ \text{Happens}(SA, \text{ADelivery}(SA, BA), T'),\\ \text{Initiates}(\text{ADelivery}(SA,BA),\\ \text{Obligation}(BA,SA,\text{APayment}(BA,SA), T'), T'),\\ \neg\text{Clipped}(\text{time3}, \text{Obligation}(BA, SA, \text{APayment}(BA, SA), T'), T2),\\ \text{HoldsAt}(\text{Valid}(SA, \text{ADelivery}(SA, BA)), T'),\\ T' < T2\end{array}}{\text{HoldsAt}(\text{Obligation}(BA, SA, \text{APayment}(BA,SA), T'), T2)}$$

$In(2)^{BA} = W^{BA} \cup \{$ Initiates(ADelivery(SA,BA), Obligation(BA, SA, APayment(BA, SA), T'), T'),

HoldsAt(Obligation(BA, SA, APayment(BA, SA), T', T2)

$\}$

$Out(2)^{BA} = W^{BA} \cup \{$ ¬ Happens(SA, ADelivery(SA, BA), T'),

¬ HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T'), ¬ (T'≤ T'),

¬ Happens(SA, ADelivery(SA, BA), T'),

¬ Initiates(ADelivery(SA,BA),Obligation(BA,SA,APayment(BA,SA), T'), T'),

Clipped(time3, Obligation(BA, SA, APayment(BA, SA), T'), T2),

¬ HoldsAt(Valid(SA, ADelivery(SA, BA)), T'), ¬ (T'< T2)

$\}$

We may wish to restrict the assumption space. Suppose we wanted our agent BA to avoid assuming the validity of actions, and use information about action validity only when it explicitly knows about it. In this case, the Out(0) set (the set of preconstraints) must be initialized to contain the forbidden assumption. For example, in this case BA constructs a PcDfT where the PC set contains the following formula:

$PC^{BA} = Out(0)^{BA} = \{$ HoldsAt(Valid(agent, action), time) $\}$

Now, neither DA1 nor DA2 defaults may be employed in the inference process.

Consider, again the same initial knowledge and the same set of rules in H, R and A sets, for the agreement between BA and SA, but, now, with the restriction that if some agent is a known debtor, then all obligations that hold towards it are terminated, from the time point at which it becomes known that the agent is a debtor, onwards.

$PC^{BA} = Out(0)^{BA} = \{ Clipped(time1, Obligation(agent1, agent2, action(agent1, agent2), time2), time2) \leftarrow$

$HoldsAt(IsADebtor(agent2), time1) \wedge time1<time2$

}

With $W^{BA} = \{ Happens(BA, AOrder(BA, SA), T) \}$ BA may, initially, perform both *no-risk* and *best-guess* reasoning by employing assumptions as shown above. Now imagine that a later time point T', BA is informed that SA is a debtor, i.e. HoldsAt(IsADebtor(SA), T') is added in its knowledge base. This new information affects its previously drawn conclusion. In this case, BA needs to traverse the lattices downwards in order to retract its earlier assumptions and conclusions, and, if necessary, to choose alternative default formulations, compatible with its current, updated, knowledge.

As noted earlier, in section 4.3, following Reiter's original computation of extensions of a DfT we may compute possible world models that are counter-intuitive: for instance, in our example above, BA would infer, after applying all of DR1, DR2, DA1 and DA2, the extension:

$In(4) = W^{BA} \cup \{$ Initiates(AOrder(BA, SA), Obligation(SA, BA, ADelivery(SA, BA), T+10), T),

HoldsAt(Obligation(SA, BA, ADelivery(SA, BA), T+10), T1),

Initiates(ADelivery(SA,BA), Obligation(BA, SA, APayment(BA, SA), T'), T'),

HoldsAt(Obligation(BA, SA, APayment(BA, SA), T'), T2)

}

This extension seems to suggest that BA infers a possible version of the world, in which it bears an obligation to pay SA, although no delivery from SA is explicitly recorded in this world, and similarly that SA bears an obligation to deliver, although this world does not explicitly record that BA's order is valid. As we explained earlier, in section 4.3, if we employ Constrained Default Logic the assumptions employed by an agent at some time point constrain its future inferences, as we require joint consistency of assumptions. Moreover, if we use Stratified Default Logic, we ensure that assumptions are employed in a rational sequence, knowledge about causal relations between rules is preserved, and the agent resorts to assumptions only when it really has to do so.

7 Related Work

We noted in the introduction that in addressing assumption-based reasoning in open agent systems, one must essentially address the following questions:

1H. What assumptions are applicable to fill in information gaps, i.e., what can be assumed by an agent to be true or false at different time points?
2H. What is the relationship between assumptions and new knowledge, i.e. how do the adopted assumptions affect future inferences, either enabling some or disabling others?
3H. What happens when new information becomes available, i.e., how does new information affect previously drawn conclusions?

Work within the assumption-based reasoning community focuses on issue (1H), only, and we review some of the main proposals in this section, in order to put forward the main difference between this work and ours. There is also work within the nonmonotonic reasoning community, which is predominantly focused on questions (2H) and (3H), and does not address the issue of dynamic identification of assumptions satisfactorily.

7.1. Assumption-based reasoning

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

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

One of the most notable approaches that fall into the static assumptions category is Doyle's Truth Maintentance System [19], which was subsequently revised and extended by de Kleer's Assumption-Based Truth Maintenance System [15, 16], which dealt with Horn clauses and tagged each inference with the assumptions and justifications that support it. Reiter and de Kleer later extended this proposal to deal with clauses that are more general than Horn clauses ([64], [17]). In another study, Kohals *et al.* in [37, 2] extended the propositional assumption-based model with probabilities and produced the Assumption-based Evidential language. Additionally, Poole's Theorist is implemented in Prolog and relies on a modification to classical logic to achieve default reasoning [54, 55, 56]. Bondarenko *et al*, [7] take an argumentation view. Kowalski and Sadri [38, 39] compare the Situation Calculus, which was primarily designed for reasoning about hypothetical actions, and the Event Calculus, which is primarily intended for reasoning about actual events. In the same line, but with considerable differences, Provetti [59] also deals with the problem of actual and

hypothetical actions in the context of the Situation Calculus and the Event Calculus. Florea's [21] proposal is based on the TLI logic, which is a first-order logic enhanced with special notation to describe Reiter's original default rules and assists in the derivation of extensions. Finally, Tahara [70] addresses the issue of inconsistency that arises in the knowledge base due to inconsistent hypotheses and uses a preference ordering between hypotheses to resolve contradictions.

The most notable approaches that fall into the second category, where assumptions are supposed to be identified and employed dynamically, include those of Cox and Pietrzykowski [13], Reichgelt and Shadbolt [60, 61], Abe [1], Pellier and Fiorino [52, 53], Jago [34] and Stamate [69]. Our work is, obviously, related mostly to this second category. However, it seems to us that assumption identification in these approaches is not truly dynamic. Before we discuss briefly each of these approaches, we make some general remarks on this issue:

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

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

Cox and Pietrzykowski in [13] explore the problem of the derivation of hypotheses to explain observed events. This is equivalent to finding what assumptions together with some axioms imply a given formula. This is similar to what we refer to as no-risk reasoning, i.e. the identification and usage of assumptions about the past. In this work, the identification of assumptions is essentially goal-driven, and it requires proof, in order to establish that the observed event is implied by what is known (the axioms) and what is assumed.

Reichgelt and Shadbolt in [60, 61] present a way to analyze planning as a form of theory extension. Theory extension enables an agent to add further assumptions to its knowledge base, in order to derive potential plans towards goal achievement. This is similar to what we refer to as best-guess reasoning, i.e. the identification and usage of

assumptions about the future. Their approach requires the use of a pre-specified assumption pool, where candidate assumptions are defined in advance, along with preconditions for their usage. The selection of an appropriate assumption from this pool is conducted in a goal-driven manner and requires that the preconditions associated with the assumption may be deductively proved from the knowledge base. If multiple assumptions have preconditions that are satisfied, selection amongst them is performed by checking them against pre-specified criteria, e.g. parsimony (the assumption with the fewest preconditions is selected) or generality (the more general assumption is preferred).

Abe [1], also, deals with the problem of missing hypotheses for the explanation of an observation. He proposes a way to generate analogous hypotheses from the knowledge base when the latter lacks the necessary ones. His work extends [64]. Hypothesis generation is done in two distinct steps: i) using first abduction and then deduction, candidate hypotheses are searched in the knowledge base, and ii) in case where such candidate assumptions do not exist in the knowledge base, analogous hypotheses are generated, by examining clauses in the knowledge base and the assumption requirements that were identified in the previous step. Hypotheses are generated ad hoc during the inference process, by exploiting predefined analogy relationships between clauses. This is an attractive approach, but it requires caution: in some applications it is difficult to define analogy relations between clauses, in advance; if no such definition for analogy is provided *a priori*, counterintuitive results may be produced: For instance, suppose that a buyer agent is obliged to pay a seller agent by some deadline, and that it actually proceeds to do so by cash deposit into the seller's bank account. Although the action of paying via a cash deposit is analogous to the action of paying in cash (in the sense that they have the same practical effect, the seller agent ends up possessing the required funds), the contract that regulates the exchange between the two agents may dictate that only payment in some specific form is deemed as acceptable. The two distinct forms of payment that seem analogous in terms of practical effects, may have different legal effects: one will result in the successful discharge of the buyer's obligation to pay the seller, while the other will result in a (technical) violation of this obligation.

Pellier and Fiorino in [52, 53] address Assumption-based Planning, and propose a mechanism by which an agent can produce "reasonable" conjectures, i.e. assumptions, based on its current knowledge. Any action precondition that cannot be proved from the knowledge base is considered to be a candidate assumption. A tentative plan (i.e. one that involves assumptions) becomes firm, and can be employed by the agent in order to achieve a specific goal, only when the agent can satisfy all of the conjectures, and this requires the agent to regard them as sub-goals and produce plans for them in turn.

Jago [34] uses the notion of context in making assumptions. A context is the current set of the agent's beliefs. Nested contexts are used to model nested assumptions, and temporally ordered contexts are used to represent the agent's set of beliefs as it changes over time. Assumptions are not identified *a priori*, but rather during the reasoning process, either by guessing or in a goal-driven manner.

Finally, Stamate [69] uses a three-valued logic to express uncertainty in logic programs; this may be attributed to uncertain information, or to missing information, i.e. information that is not derivable using the current knowledge and program rules.

7.2. Nonmonotonic reasoning

The EC representation of a contract allows us to establish what factual and normative fluents are true at a given time point, through appropriate queries on the HoldsAt predicate. This representation, though, does not allow us to reason with incomplete knowledge dynamically.

There are various approaches to reasoning with incomplete knowledge, such as the Closed World Assumption [62], Circumscription [47], Logic Programs [22, 23], and Defeasible Logic [50], and in fact these have been explored by many researchers (for example, [46, 6, 20, 65, 71, 31, 50, 51] among others).

Closed World Assumption

Under the Closed World Assumption (CWA) [62], an atomic formula is assumed false, unless it is explicitly known to be true. When an agent that uses a (possibly incomplete) EC contract representation, coupled with CWA, makes assumptions, these concern the *falsity* of certain formulae, rather than their truth. That is, in addressing question (1H), CWA dictates that an information gap be treated as negative information (one might call this the 'atheist' stance). In many realistic scenarios, however, it is important to be able to make assumptions about the *truth* (rather than the falsity) of certain formulae, i.e., to treat information gaps for what they are (absence of definite information) as potentially positive information (one might call this the 'agnostic' stance). For instance, suppose that a buyer agent has a rule determining whether it bears an obligation to pay a seller agent. Some of the rule conditions may be whether the buyer placed an order with the seller, whether the goods that were ordered were, in fact, delivered, and if so, whether such delivery met all possible requirements (e.g. the right quantity and quality of goods were delivered, to the right place of delivery, using the right delivery method, at the right time and so on). The buyer agent will assume that anything it does not know about explicitly is false, so if does not possess explicit information about one or more of these conditions, it will not infer that it bears the obligation to pay the seller agent through the application of the rule, and may infer, through CWA, that it does not bear such an obligation. The buyer agent, in this case, cannot exploit assumptions in order to perform no-risk or best-guess reasoning. Since any assumptions employed at some point of the inference process are not retained for future reference, there is no way to relate them to future inferences. Hence with CWA we cannot address (2H) satisfactorily. When new information becomes available, possibly refuting some of the assumptions that were employed at earlier points in the inference process, there is no way to retract previously drawn conclusions, that is CWA does not address (3H) satisfactorily. Of course, one may argue that such questions can be addressed, in a domain-specific manner, via the use of special purpose predicates (e.g. by recording assumptions used during the inference of each specific conclusion). However, we argued that by resort-

ing to Default Logic we obtain a more general-purpose solution to the problem of dynamic assumption identification, which is also compatible with our common intuitions.

Circumscription

Circumscription [47] is a generalization of the CWA, and might be used instead of it (work described in [71] is in this direction). Here, special predicates are used, in order to denote abnormal (unexpected) events and effects of actions, and the inference strategy attempts to minimize abnormality. The agent possesses explicit information about abnormality, and the conclusions derived are those contained in the minimal models of the (augmented with special predicates about abnormality) knowledge base. It is now possible for the seller agent of our example above to perform best-guess and no-risk reasoning, for if it does not explicitly know that delivery was 'abnormal' in some sense (e.g. it never happened, or it happened at the wrong time, or the wrong quantity or quality of goods were delivered an so on), it will use its rule to infer that it has an obligation to pay. That is, the agent is able to treat an information gap as potentially positive (true) information. However, Circumscription poses some other problems, acknowledged by other researchers, as well: First, it requires that we define abnormal events, effects of actions and the like, ***explicitly***, and, also, that we distinguish each abnormal individual from other individuals, explicitly [8] (page 222)]. Second, in order to decide which individuals to characterize as abnormal, we are required to anticipate the conclusions that we want to be able to derive [3] (page 149)]. Finally, in addressing (2H) and (3H), Circumscription suffers from the same problems as CWA.

Logic Programs

The correspondence between Logic Programs (with stable model or answer set semantics) and default theories has been established in [43]. We might consider the EC contact representation as a (general) Logic Program, with stable model semantics [22], or as an extended Logic Program, with answer sets [23] – in fact, work described in [31] and [51] adopt the former view. In both cases, entailment is goal-driven. In stable model semantics, given a logic program LP we define its reduction LP_M with respect to a set of goal atoms M. A stable model may be computed following two steps. First, by removing all ground instances of rules contained in LP, that have in their body negative literals $\neg B$, where $B \in M$. Second, by removing all ground negative literals in the bodies of the rules that remain in LP. In answer sets semantics a similar procedure is used to compute the answer set of LP. Note that the elimination steps, described above, for the computation of a stable model or an answer set, presuppose the rejection of all rules that either contradict the set of goal atoms, or are irrelevant with the goal and, furthermore, these steps presuppose the falsity of all assumptions. The absence of an atom A from a stable model of LP is taken to signal that A is false. The absence of an atom A from an answer set of an extended LP is taken to mean that A is unknown. In the light of the comments we made earlier, when discussing CWA, it seems to us that answer set semantics are preferable to stable model semantics, for they enable us to treat information gaps in a more open-minded way (the agnostic vs. the atheist stance). What we find problematic in both cases though,

for the purposes of assumption-based reasoning (and specifically in relation to question (1H)), is the fact that potential assumptions can only be spotted in a goal-driven manner. The agent needs to decide *a priori* what conclusion it wants to derive, in order to be able to identify which assumptions are essential to make, in order to be able to actually derive it.

Defeasible Logic

Finally, there is another approach to default reasoning with e-contracts, namely Defeasible Logic [50], which is used by [30] and [51]. Defeasible Logic allows us to define which conclusions are retractable, by making a distinction between strict and defeasible rules. Knowing that some information is defeasible enables an agent to treat it as a potential assumption. A question that arises is whether it is possible to determine, *a priori*, during the construction of the rule base, what is and what is not defeasible. In some situations (such as the examples shown in [30] and [51]) we are, indeed, able to determine this on the basis of some specific domain information. In this case though, the agent does not discover potentially useful assumptions for itself; rather it uses an implicitly pre-specified pool of assumptions. We can see a way out of this problem: we may adopt a more general view and consider all derived conclusions as defeasible. Rule conditions that are themselves defined through defeasible rules, are defeasible. Rule conditions that are not defined through rules must be provided either as strict facts or as defeasible facts. The agent will treat anything that it does not know about as a potential assumption. However, in order to establish whether some information gap exists, it will need to carry out proof on its knowledge base (to determine which defeasible rules fire), which is computationally expensive.

8 Conclusions and Future Work

The work in this report is motivated from the need for hypothetical nonmonotonic reasoning in an e-commerce setting, which is the application area of our project, and more generally in normative systems, where realistically agents will have incomplete knowledge about their environment, and about other agents. The questions we seek to address is whether it is possible for agents to identify appropriate assumptions dynamically, how these assumptions affect subsequent inferences and what happens when new information that becomes available at some time point disproves employed assumptions. We argued that e-contracts could be represented as Default Theories and, at first, we proposed a theoretical way in which such theories could be constructed automatically from initial contract representations, e.g. Event Calculus-based contract representations. That proposal relied on determining what information could be proved from the agent's knowledge base, in order to decide whether it would serve as an assumption or not. Afterwards, we proposed an incremental technique that can be used for this construction that enables the dynamic and ad hoc identification of candidate assumptions without resorting to proof. We presented a full Event Calculus-based contract representation and illustrate via examples the way the technique presented in this work addresses all three issues of interest (1H)-(3H) and supports e-contracts execution and performance monitoring. Moreover, note that, a contract

representation as a Default Theory enables agents to manage normative conflicts, first by detecting conflicts in an extension or among distinct extensions and, second by applying dynamically priorities among rules that are based on various criteria. It is out of our scope to discuss normative conflict management in this work. A detailed discussion is available in our previous work [24, 26, 28]. Finally, we noted some limitations of EC representations with respect to reasoning with incomplete knowledge and discussed why various approaches to nonmonotonic reasoning, such as CWA, Logic Programs, Circumscription and Defeasible Logic, that could be used in conjunction with an EC contract representation seem unsatisfactory. In the same spirit, we discussed other approaches to assumption-based reasoning and the need for the ad hoc, dynamic, derivation of hypotheses.

We have developed a prototype implementation based on our previous idea to hypothetical reasoning [25, 27], which translates initial propositional representations into propositional Default Theories. The prototype follows the specifications listed below: (i) e-contract rules are represented as sentences of the form (1), (ii) extensions are computed in the manner presented in [5], i.e., by maintaining syntactically consistent sets of formulae, (iii) normal defaults are not considered, (iv) the hierarchical structure off all possible default formulations is constructed incrementally during the inference process, (v) the applicability of defaults is checked in the same order that initial rules are given, and (vi) in levels that contain more than one corresponding defaults for the same initial rule, i.e. level 1 to level $max(k_i)$-1 ($1 \leq i \leq r$), the applicability of defaults is checked in the order that the defaults are placed in this level.

Our current work focuses on employing the ideas discussed in this report in our existing prototype implementation. Naturally we are, also, interested in extending our implementation so that it may translate FOL representations into Default Theories. Note that in an initial (FOL) Event Calculus representation, all variables are implicitly assumed to be universally quantified. So the question that arises for the translation is what is the appropriate quantification for the variables that appear in the resulting default rules. There are four major approaches (cf. [63, 42, 55, 35, 36]) to the semantics of open Default Theories, and we have yet to investigate which one might be appropriate for computational purposes. Moreover, it is our intention to examine how the dynamic and ad hoc identification of candidate assumptions, as it is presented in this report, can be applied to other approaches to nonmonotonic reasoning such as Logic Programs and Defeasible Logic.

Acknowledgments

This work was supported by the European Social Fund and the Greek Secretariat for Research and Technology through the 3rd Community Support Programme - Measure 8.3 (PENED2003-03EΔ466).

References

[1] Akinori Abe. Two-sided hypotheses generation for abductive analogical reasoning. In *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99)*, pages 145–152, Washington, DC, USA, 1999. IEEE Computer Society.

[2] Bernhard Anrig, Rolf Haenni, Jörg Kohlas, and Norbert Lehmann. Assumption-based modeling using ABEL. In Dov M. Gabbay, Rudolf Kruse, Andreas Nonnengart, and Hans Jörgen Ohlbach, editors, *ECSQARU-FAPR*, volume 1244 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 1997.

[3] Grigoris Antoniou. *Nonmonotonic Reasoning*. MIT Press, 1997.

[4] Grigoris Antoniou. Stratification for default logic variants. *International Journal of Intelligent Systems*, 13(9):785–799, 1998.

[5] Grigoris Antoniou. A tutorial on default logics. *ACM Computer Surveys*, 31(4):337–359, 1999.

[6] Alexander Artikis, Jeremy Pitt, and Marek J. Sergot. Animated specifications of computational societies. In *1st International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1053–1061, Bologna, Italy, July 15-19 2002. ACM.

[7] Andrei Bondarenko, Francesca Toni, and Robert A. Kowalski. An assumption-based framework for non-monotonic reasoning. In *LPNMR*, pages 171–189, 1993.

[8] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[9] Robert L. Causey. EVID: a system for interactive defeasible reasoning. *Decis. Support Syst.*, 11(2):103–131, 1994.

[10] Pawel Cholewinski. Stratified default theories. In Leszek Pacholski and Jerzy Tiuryn, editors, *CSL*, volume 933 of *Lecture Notes in Computer Science*, pages 456–470. Springer, 1994.

[11] Pawel Cholewinski. Reasoning with stratified default theories. In V. Wiktor Marek and Anil Nerode, editors, *LPNMR*, volume 928 of *Lecture Notes in Computer Science*, pages 273–286. Springer, 1995.

[12] Philip T. Cox and Tomasz Pietrzykowski. A complete, nonredundant algorithm for reversed skolemization. *Theor. Comput. Sci.*, 28:239–261, 1984.

[13] Philip T. Cox and Tomasz Pietrzykowski. Causes for events: Their computation and applications. In Jǿrg H. Siekmann, editor, *CADE*, volume 230 of *Lecture Notes in Computer Science*, pages 608–621, New York, NY, USA, 1986. Springer-Verlag New York, Inc.

[14] Aspassia Daskalopulu. Modeling legal contracts as processes. In *Legal Information Systems Applications, 11th International Workshop on Database and Expert Systems Applications (DEXA'00)*, pages 1074–1079. IEEE Computer Society, 2000.

[15] Johan de Kleer. An assumption-based TMS. *Artif. Intell.*, 28(2):127–162, 1986.

[16] Johan de Kleer. Extending the ATMS. *Artif. Intell.*, 28(2):163–196, 1986.

[17] Johan de Kleer. A general labeling algorithm for Assumption-based Truth Maintenance. In *Proceedings of the 7th National Conference on Artificial Intelligence. St. Paul, MN, August 21-26, AAAI Press / The MIT Press*, pages 188–192, 1988.

[18] Sandra K. Dewitz, Young Ryu, and Ronald M. Lee. Defeasible reasoning in law. *Decis. Support Syst.*, 11(2):133–155, 1994.

[19] Jon Doyle. A truth maintenance system. *Artif. Intell.*, 12(3):231–272, 1979.

[20] Andrew D. H. Farrell, Marek J. Sergot, Mathias Salli, and Claudio Bartolini. Using the event calculus for tracking the normative state of contracts. *Int. J. Cooperative Inf. Syst.*, 14(2-3):99–129, 2005.

[21] Adina Magda Florea. Assumption-based reasoning of intelligent agents. In *11th International Conference on Control Systems and Computer Science (CSCS'97)*, pages 126–133, 1997.

[22] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. pages 1070–1080, 1988.

[23] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Comput.*, 9(3/4):365–386, 1991.

[24] Georgios K. Giannikis and Aspassia Daskalopulu. Defeasible reasoning with e-contracts. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2006)*, pages 690–694, Hong Kong, China, 2006. IEEE Computer Society.

[25] Georgios K. Giannikis and Aspassia Daskalopulu. E-contracting agents reasoning with assumptions. Technical Report 5343/1, A006.3ΓΙΑ, Department of Computer and Communications Engineering, University of Thessaly, 5343/1, A006.3ΓΙΑ, May 2007.

[26] Georgios K. Giannikis and Aspassia Daskalopulu. The representation of e-Contracts as default theories. In H.G. Okuno and M. Ali, editors, *Proceedings of 19th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2007)*, LNAI 4570, pages 963–973, Kyoto, Japan, 2007. Springer-Verlag Berlin Heidelberg.

[27] Georgios K. Giannikis and Aspassia Daskalopulu. How agents know what to assume when? In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2008)*, pages 538–545, Sydney, Australia, 9 - 12 December 2008. IEEE Computer Society.

[28] Georgios K. Giannikis and Aspassia Daskalopulu. Normative conflicts: Patterns, detection and resolution. In *International Conference Web Information Systems and Technologies (WEBIST 2009)*, pages 527–532, Lisbon, Portugal, 23-26 March 2009.

[29] Georgios K. Giannikis and Aspassia Daskalopulu. The role of assumption identification in autonomous agent reasoning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Budapest, Hungary, 10-15 May 2009.

[30] Guido Governatori and Duy Hoang. A semantic web based architecture for e-contracts in defeasible logic. In *1st International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 145–159, 2005.

[31] Benjamin N. Grosof. Representing e-commerce rules via situated courteous logic programs in RuleML. *Electronic Commerce Research and Applications*, 3(1):2–20, 2004.

[32] Jaap Hage. Law and defeasibility. *Artif. Intell. Law*, 11(2-3):221–243, 2003.

[33] Carl Hewitt. The challenge of open systems: current logic programming methods may be insufficient for developing the intelligent systems of the future. *BYTE*, 10(4):223–242, 1985.

[34] Mark Jago. Modelling assumption-based reasoning using contexts. In *Workshop on Context Representation and Reasoning (CRR'05)*, 2005.

[35] Michael Kaminski. A comparative study of open default theories. *Artif. Intell.*, 77(2):285–319, 1995.

[36] Michael Kaminski, Johann A. Makowsky, and Michael L. Tiomkin. Extensions for open default theories via the domain closure assumption. *J. Log. Comput.*, 8(2):169–187, 1998.

[37] Jörg Kohlas and Paul-Andri Monney. Probabilistic assumption-based reasoning. In David Heckerman and E. H. Mamdani, editors, *UAI*, pages 485–491. Morgan Kaufmann, 1993.

[38] Robert A. Kowalski and Fariba Sadri. The situation calculus and event calculus compared. In *Maurice Bruynooghe (Ed.): Logic Programming, Proceedings of the 1994 International Symposium, November 13-17, MIT Press*, pages 539–553, 1994.

[39] Robert A. Kowalski and Fariba Sadri. Reconciling the event calculus with the situation calculus. *J. Log. Program.*, 31(1-3):39–58, 1997.

[40] Robert A. Kowalski and Marek J. Sergot. A logic-based calculus of events. *New Generation Comput.*, 4(1):67–95, 1986.

[41] V. Lifschitz. Success of default logic. In *Hector Levesque and Fiora Pirri editors, Logical Foundations for Cognitive Agents: Contributions in Honour of Ray Reiter*, pages 208–212. Springer Verlag, 1999.

[42] Vladimir Lifschitz. On open defaults. In J. Lloyd, editor, *Proceedings of the symposium on computational logic*, pages 80–95. Berlin: Springer-Verlag, 1990.

[43] Fangzhen Lin and Yoav Shoham. Argument systems: A uniform basis for nonmonotonic reasoning. In *KR*, pages 245–255, 1989.

[44] Donald W. Loveland. A linear format for resolution. *Lecture Notes in Mathematics 125, Springer-Verlag, Berlin*, pages 147 – 162, 1970.

[45] David Makinson. On the formal representation of rights relations. *Journal of Philosophical Logic*, 15(4):403–425, 1986.

[46] Rafaul Hernandez Marvn and Giovanni Sartor. Time and norms: a formalisation in the event-calculus. In *7th International Conference on Artificial Intelligence and Law*, pages 90–99, New York, NY, USA, 1999. ACM Press.

[47] John McCarthy. Circumscription - a form of non-monotonic reasoning. *Artif. Intell.*, 13(1-2):27–39, 1980.

[48] Rob Miller and Murray Shanahan. The event calculus in classical logic - alternative axiomatisations. *Electron. Trans. Artif. Intell.*, 3(A):77–105, 1999.

[49] Dana S. Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman. Shop2: An htn planning system. *J. Artif. Intell. Res. (JAIR)*, 20:379–404, 2003.

[50] Donald Nute. Defeasible logic. In Dov Gabbay, Christopher J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Nonmonotonic Reasoning and Uncertain Reasoning*, volume 3, pages 353–395. Oxford University Press, 1994.

[51] Adrian Paschke, Martin Bichler, and Jens Dietrich. ContractLog: An approach to rule based monitoring and execution of service level agreements. In *1st International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 209–217, 2005.

[52] Damien Pellier and Humbert Fiorino. Assumption-based planning. In *AISTA'04: International Conference on Advances in Intelligence Systems Theory and Applications*, pages 367–376, Luxembourg-Kirchberg, Luxembourg, November 2004.

[53] Damien Pellier and Humbert Fiorino. Multi-agent assumption-based planning. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 1717–1718. Professional Book Center, 2005.

[54] D. Poole, R. Goebel, and R. Aleliunas. *Theorist: a logical reasoning system for defaults and diagnosis*, pages 331–352. Knowledge Frontier: Essays in the Representation of Knowledge, Springer Verlag, New York, 1987.

[55] David Poole. A logical framework for default reasoning. *Artif. Intell.*, 36(1):27–47, 1988.

[56] David Poole. *Who chooses the assumptions?* Abductive Reasoning, Cambridge: MIT Press., 1996.

[57] Henry Prakken. An argumentation framework in default logic. *Ann. Math. Artif. Intell.*, 9(1-2):93–132, 1993.

[58] Henry Prakken and Giovanni Sartor. The three faces of defeasibility in the law. *Ratio Juris*, 17:118–139, 2004.

[59] Alessandro Provetti. Hypothetical reasoning about actions: From situation calculus to event calculus. *Computational Intelligence*, 12:478–498, 1996.

[60] Han Reichgelt and Nigel Shadbolt. Planning as theory extension. In *Proceedings of the Seventh Conference on Artificial Intelligence and Simulation of Behaviour (AISB89)*, pages 191–199, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[61] Han Reichgelt and Nigel Shadbolt. A specification tool for planning systems. In *European Conference on Artificial Intelligence (ECAI90)*, pages 541–546, 1990.

[62] Raymond Reiter. On closed world data bases. In *Logic and Data Bases*, pages 55–76, 1977.

[63] Raymond Reiter. A logic for default reasoning. *Artif. Intell.*, 13(1-2):81–132, 1980.

[64] Raymond Reiter and Johan de Kleer. Foundations of assumption-based truth maintenance systems: Preliminary report. In *Proceedings AAAI-87 Sixth National Conference on Artificial Intelligence, July 13-17, Seattle, WA. AAAI Press*, pages 183–189, 1987.

[65] Mohsen Rouached, Olivier Perrin, and Claude Godart. A contract-based approach for monitoring collaborative web services using commitments in the event calculus. In Anne H. H. Ngu, Masaru Kitsuregawa, Erich J. Neuhold, Jen-Yao Chung, and Quan Z. Sheng, editors, *6th International Conference on Web Information Systems Engineering*, volume 3806 of *Lecture Notes in Computer Science*, pages 426–434. Springer, 2005.

[66] Giovanni Sartor. A simple computational model for nonmonotonic and adversarial legal reasoning. In *ICAIL '93: Proceedings of the 4th international conference on Artificial intelligence and law*, pages 192–201, 1993.

[67] Torsten Schaub. On constrained default theories. In *17th European Conference on Artificial Intelligence (ECAI'92)*, pages 304–308, 1992.

[68] Torsten Schaub. Variations of constrained default logic. In Michael Clarke, Rudolf Kruse, and Serafn Moral, editors, *ICSQARU*, volume 747 of *Lecture Notes in Computer Science*, pages 310–317. Springer, 1993.

[69] Daniel Stamate. Assumption based multi-valued semantics for extended logic programs. In *ISMVL*, page 10. IEEE Computer Society, 2006.

[70] Ikuo Tahara. Computing scenario from knowledge with preferentially ordered hypotheses. *Systems and Computers in Japan*, 35(4):19–26, 2004.

[71] Pinar Yolum and Munindar P. Singh. Reasoning about commitments in the event calculus: An approach for specifying and executing protocols. *Ann. Math. Artif. Intell.*, 42(1-3):227–253, 2004.