MKNF Knowledge Bases in Multi-Context Systems

Martin Homola[1], Matthias Knorr[2], João Leite[2], and Martin Slota[2]

[1] Faculty of Mathematics, Physics and Informatics, Comenius University
[2] CENTRIA & Departamento de Informática, Universidade Nova de Lisboa

Abstract. In this paper we investigate the relationship between Multi-Context Systems and Hybrid MKNF Knowledge Bases. Multi-Context Systems provide an effective and modular way to integrate knowledge from different heterogeneous sources (contexts) through so-called bridge rules. Hybrid MKNF Knowledge Bases, based on the logic of minimal knowledge and negation as failure (MKNF), allow for a seamless combination of description logic ontology languages with non-monotonic logic programming rules. In this paper, we not only show that Hybrid MKNF Knowledge Bases can be used as particular contexts in Multi-Context Systems, but we also provide transformations from the former into the latter, without the need for an explicit Hybrid MKNF context, hence providing a way for agents to reason with Hybrid MKNF Knowledge Bases within Multi-Context Systems without the need for specialized Hybrid MKNF reasoners.

1 Introduction

In *Open Multi-Agent Systems*, interaction and cooperation is increasingly being governed by *institutions* that regulate agents' behaviour and promote desirable properties. In such systems, it is crucial for agents and institutions to make sense of knowledge obtained from different sources, not only to increase the chance of individually making the right choice, but also to potentiate the chance of agreement in negotiations.

These sources of knowledge include the increasing number of available ontologies and rule sets, to a large extent developed within initiatives such as Semantic Web and Linked Open Data, as well as the norms and policies published by the *institutions*, the information communicated by other agents, to name only a few. With such diverse sources of knowledge to deal with, agent developers have turned their attention to Multi-Context Systems (MCS) [7,8,4,5,16,17]. Within MCSs, knowledge is modularly composed of contexts, each of which possibly encapsulating a source of knowledge of a different type, while bridge rules provide effective means for integration [12,11]. With the equilibria semantics of Brewka and Eiter [6], MCSs provide an effective and modular way to integrate knowledge from different heterogeneous sources, for example, different ontologies written in some Description Logic based ontology language, such as OWL, a rule set written in Answer-Set Programming representing some business policies, or some facts written in propositional logic representing the agent's model of some other agent, to name only a few. MCSs are simple enough to allow this heterogeneous knowledge to be bridged and integrated, while keeping their distinct provenance.

M. Fisher et al. (Eds.): CLIMA XIII 2012, LNAI 7486, pp. 146–162, 2012.

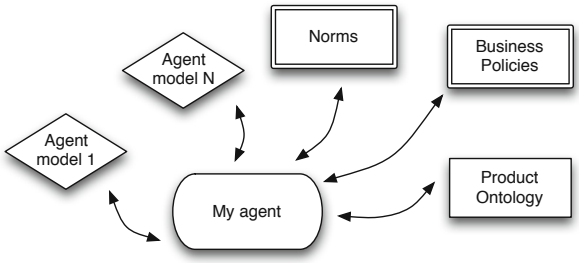

Fig. 1. Schematic depiction of a MCS representing agent's view of the system

For example, consider an open multi-agent system in which agents participate in online trading. In such a system, an agent may need to represent knowledge and reason about an ontology of products that can be purchased, models of the other existing agents (e.g. their perceived intentions based on communicated information and observed behaviour), its own business policies as well as those of the system in which it is integrated, together with existing norms. Such a system can be modelled as MCS, as depicted in Fig. 1. In such an MCS, ontologies can be modelled with DL contexts, the business policies and norms could be modelled with rule-based contexts, e.g., using logic programs, and other agents with separate contexts using propositional facts or additional rules in a more complex case. To propagate logical consequences across contexts, contexts are connected with bridge rules, illustrated by arrows in Fig. 1.

Recently, it has been shown in [2,1] that realistic norms and policies that mimic the real world require a more complex knowledge representation formalism, such as Hybrid MKNF [15] – based on the Logic of Minimal Knowledge and Negation as Failure (MKNF) [14] – that tightly combines Logic Programming (LP) and Description Logic (DL). In such scenarios, the Closed-World Assumption provided by LP rules is used e.g. to deal with defeasible knowledge, such as exceptions, while the Open-World Assumption provided by DL axioms is employed e.g. to deal with ontological knowledge and features such as reasoning with unknown individuals.

If norms and policies are to be published in such a language, it is crucial to relate them to MCSs, so that agent's imbued with the ability to deal with MCSs can also reason with them.

To this purpose, in this technical paper we investigate the relationship between MCSs and Hybrid MKNF. Taking the two-valued semantics of Hybrid MKNF [15], which is based on the Stable Model Semantics [9], we provide transformations from Hybrid MKNF into MCSs without the need for an explicit Hybrid MKNF context. This provides a way for agents to reason with Knowledge Bases written in Hybrid MKNF within MCSs, with additional contexts and bridge rules, *without the need for specialized Hybrid MKNF reasoners*.

The main contributions of this paper are three distinct ways to deal with MKNF Knowledge bases within MCSs, namely by:

– using Hybrid MKNF in the form of an MKNF context, which can then be bridged to other contexts;

– translating each Hybrid MKNF context into a First-Order context together with additional *non-monotonic* bridge rules;
– translating each Hybrid MKNF context into two contexts, a DL context and a fact base context, together with *non-monotonic* bridge rules.

The remainder of this paper is structured as follows: in Sect. 2, we review Description Logics, Logic Programs, Multi-Context Systems, and MKNF Knowledge Bases. In Sect. 3, we introduce MKNF contexts, present the two transformations into other kinds of contexts, and illustrate their use with fragments of the running example. We conclude in Sect. 4 and point to future directions.

2 Preliminaries

2.1 Description Logics

We first briefly summarise the syntax and semantics of standard function-free first-order logic with equality which forms the basis for representing both ontological and rule-based knowledge.

We assume the standard syntax of first-order *atoms*, *formulas* and *sentences*, defined inductively over disjoint sets of *constant* and *predicate symbols* $\mathbf{C}$ and $\mathbf{P}$. A first-order formula is *ground* if it contains no variables. The set of all first-order sentences is denoted by Φ. A *first-order theory* is a set of first-order sentences.

The satisfaction of a first-order sentence ϕ in a standard first-order interpretation I is denoted by $I \models \phi$; we also say that I is a *model of* ϕ if $I \models \phi$.

Description Logics (DLs) [3] are fragments of first-order logic whose reasoning tasks are usually decidable. Throughout the paper we assume that some first-order fragment is used to describe an *ontology*, i.e. to specify a shared conceptualisation of a domain of interest. Unless stated otherwise, we do not constrain ourselves to a specific DL for representing ontologies. The only assumption taken in the theoretical developments is that the ontology language is a syntactic variant of a fragment of first-order logic, covering also cases when this fragment would normally not be considered a DL. We assume that for any ontology axiom ϕ and ontology $\mathcal{O}$, $\kappa(\phi)$ and $\kappa(\mathcal{O})$ denote a first-order sentence that semantically correspond to ϕ and $\mathcal{O}$, respectively. Such translations are known for most DLs [3]. Given a first-order sentence ϕ, we say that an ontology $\mathcal{O}$ *entails* ϕ, denoted by $\mathcal{O} \models \phi$, if and only if every first-order model of $\kappa(\mathcal{O})$ is also a first-order model of ϕ.

2.2 Logic Programs

Like Description Logics, Logic Programming has its roots in classical first-order logic. However, logic programs diverge from first-order semantics by adopting the Closed World Assumption and allowing for non-monotonic inferences. In what follows, we introduce the syntax of extended normal logic programs and define the *stable models* [9] for such programs.

Syntactically, logic programs are built from *atoms* consisting of first-order atoms without equality. An *objective literal* is an atom p or its (strong) negation $\neg p$. We denote

the set of all objective literals by $\mathbf{L}$ and the set of ground objective literals by $\mathbf{L}_\mathsf{G}$. A *default literal* is an objective literal preceded by $\sim$ denoting *default negation*. A *literal* is either an objective literal or a default literal. Given a set of literals S, we introduce the following notation: $S^+ = \{\, l \in \mathbf{L} \mid l \in S \,\}$, $S^- = \{\, l \in \mathbf{L} \mid \sim l \in S \,\}$, $\sim S = \{\, \sim L \mid L \in S \,\}$.

A *rule* is a pair $\pi = (H(\pi), B(\pi))$ where $H(\pi)$ is an objective literal, referred to as *head of* π and $B(\pi)$ is a set of literals, referred to as *body of* π. Usually, for convenience, we write π as $(H(\pi) \leftarrow B(\pi)^+, \sim B(\pi)^-.)$ We also say that $B(\pi)^+$ is the *positive body of* π and $B(\pi)^-$ the *negative body of* π. A rule is called *ground* if it does not contain variables and *definite* if it does not contain any default literal. The *grounding* of a rule π is the set of rules $\mathsf{gr}(\pi)$ obtained by replacing in π all variables with constant symbols from $\mathbf{C}$ in all possible ways. A *program* is a set of rules. A program is *ground* if all its rules are ground; *definite* if all its rules are definite. The grounding of a program $\mathcal{P}$ is defined as $\mathsf{gr}(\mathcal{P}) = \bigcup_{\pi \in \mathcal{P}} \mathsf{gr}(\pi)$.

The stable models of a program are determined by considering its first-order models in which all constant symbols are interpreted by themselves, and every ground atom p is interpreted separately of (though still consistently with) its strong negation $\neg p$. An interpretation thus corresponds to a subset of $\mathbf{L}_\mathsf{G}$ that does not contain both p and $\neg p$ for any ground atom p and *models of programs* are determined by treating rules as classical implications. A stable model is then a model of the program that can be fully derived using rules of the program assuming that literals not present in it are false by default.[1]

Definition 1 (Stable Model). *Let $\mathcal{P}$ be a ground program. An interpretation J is a* stable model of $\mathcal{P}$ *if and only if J is a subset-minimal model of the* reduct of $\mathcal{P}$ relative to J*:* $\mathcal{P}^J = \{\, H(\pi) \leftarrow B(\pi)^+. \mid \pi \in \mathcal{P} \wedge J \models \sim B(\pi)^- \,\}$.

The stable models of a non-ground program $\mathcal{P}$ are the stable models of $\mathsf{gr}(\mathcal{P})$. *The set of all stable models of a program $\mathcal{P}$ is denoted by* $[\![\mathcal{P}]\!]_{\mathsf{SM}}$.

2.3 Multi-Context Systems

Syntax of Multi-Context Systems. Following [6], a multi-context system consists of a collection of components, each of which contains knowledge represented in some logic. Abstractly, a *logic* is a triple $L = (\mathbf{KB}, \mathbf{BS}, \mathbf{ACC})$ where $\mathbf{KB}$ is the set of well-formed knowledge bases of L , $\mathbf{BS}$ is the set of possible belief sets[2] and $\mathbf{ACC} : \mathbf{KB} \rightarrow 2^{\mathbf{BS}}$ is a function describing the semantics of L by assigning to each knowledge base a set of acceptable belief sets.

In addition to the knowledge base in each component, *bridge rules* are used to interconnect the components, specifying what knowledge to assert in one component given certain beliefs held in the other components. Formally, for a collection of logics $L = \langle L_1, \ldots, L_n \rangle$, an *$L_i$-bridge rule σ over L*, $1 \leq i \leq n$, is of the form $(H(\sigma) \leftarrow B(\sigma).)$, where $B(\sigma)$ is a set of *bridge literals* of the forms $(r : p)$ and $\mathbf{not}\,(r : p)$ where $1 \leq r \leq n$ and p is an element of some belief set of L_r, and for each $kb \in \mathbf{KB}_i : kb \cup \{\, H(\sigma) \,\} \in \mathbf{KB}_i$.

[1] Note that, unlike in [10], we do not allow a program without a model to have a stable model.

[2] We assume that each element of $\mathbf{KB}$ and $\mathbf{BS}$ is a set.

Thus, putting these concepts together, a *multi-context system* (MCS) is a collection of contexts $M = \langle C_1, \ldots, C_n \rangle$ where $C_i = (L_i, kb_i, br_i)$, $L_i = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$ is a logic, $kb_i \in \mathbf{KB}_i$ a knowledge base, and br_i is a set of L_i-bridge rules over $\langle L_1, \ldots, L_n \rangle$.

In the following we present the *grounded equilibria* semantics [6], which is motivated by the stable models semantics for logic programs.

Grounded Equilibria. Given an MCS $M = \langle C_1, \ldots, C_n \rangle$, a *belief state of* M is a sequence $S = \langle S_1, \ldots, S_n \rangle$ such that each S_i is an element of $\mathbf{BS}_i$. For every bridge literal $(r : p)$ we write $S \models (r : p)$ if $p \in S_r$ and $S \models \mathbf{not}\,(r : p)$ if $p \notin S_r$; for a set of bridge literals S, $S \models S$ if $S \models L$ for every $L \in S$.

A belief state $S = \langle S_1, \ldots, S_n \rangle$ of M is an *equilibrium* if, for all i with $1 \leq i \leq n$, the following condition holds:

$$S_i \in \mathbf{ACC}_i(kb_i \cup \{\, H(\sigma) \mid \sigma \in br_i \wedge S \models B(\sigma) \,\}) \ .$$

We say that an equilibrium S is *minimal* if there is no equilibrium $S' = \langle S'_1, \ldots, S'_n \rangle$ such that $S'_i \subseteq S_i$ for all i with $1 \leq i \leq n$ and $S'_j \subsetneq S_j$ for some j with $1 \leq j \leq n$.

Now we formalise the conditions under which the minimal equilibrium is *unique*, in which case we assign it as the *grounded equilibrium* of the MCS. This can be guaranteed if the contexts can be *reduced*, using a reduction function, to monotonic ones. Formally, a logic $L = (\mathbf{KB}, \mathbf{BS}, \mathbf{ACC})$ is *monotonic* if

1. $\mathbf{ACC}(kb)$ is a singleton set for each $kb \in \mathbf{KB}$, and
2. $S \subseteq S'$ whenever $kb \subseteq kb'$, $\mathbf{ACC}(kb) = \{\, S \,\}$ and $\mathbf{ACC}(kb') = \{\, S' \,\}$.

Furthermore, $L = (\mathbf{KB}, \mathbf{BS}, \mathbf{ACC})$ is *reducible* if

1. there is $\mathbf{KB}^* \subseteq \mathbf{KB}$ such that the restriction of L to $\mathbf{KB}^*$ is monotonic,
2. there is a reduction function $red : \mathbf{KB} \times \mathbf{BS} \to \mathbf{KB}^*$ such that for each $kb \in \mathbf{KB}$ and $S, S' \in \mathbf{BS}$:
   - $red(kb, S) = kb$ whenever $kb \in \mathbf{KB}^*$,
   - red is antimonotone in the second argument, that is $red(kb, S) \subseteq red(kb, S')$ whenever $S' \subseteq S$,
   - $S \in \mathbf{ACC}(kb)$ if and only if $\mathbf{ACC}(red(kb, S)) = \{\, S \,\}$.

A context $C = (L, kb, br)$ is *reducible* if its logic L is reducible and, for all $H \subseteq \{\, H(\sigma) \mid \sigma \in br \,\}$ and all belief sets S, $red(kb \cup H, S) = red(kb, S) \cup H$.

An MCS is *reducible* if all of its contexts are. Note that a context is reducible whenever its logic L is monotonic. In this case $\mathbf{KB}^*$ coincides with $\mathbf{KB}$ and red is identity with respect to the first argument. A reducible MCS $M = \langle C_1, \ldots, C_n \rangle$ is *definite* if

1. none of the bridge rules in any context contains $\mathbf{not}$,
2. for all i and all $S \in \mathbf{BS}_i$, $kb_i = red_i(kb_i, S)$.

In a definite MCS bridge rules are monotonic, and knowledge bases are already in reduced form. Inference is thus monotonic and a unique minimal equilibrium exists. We take this equilibrium to be the grounded equilibrium:

Definition 2 (Grounded Equilibrium of a Definite MCS). *Let $M = \langle C_1, \ldots, C_n \rangle$ be a definite MCS. $S = \langle S_1, \ldots, S_n \rangle$ is the* grounded equilibrium of M*, denoted by* $\mathbf{GE}(M)$*, if S is the unique minimal equilibrium of M.*

Grounded equilibria for general MCSs are defined based on a reduct which generalises the Gelfond-Lifschitz reduct to the multi-context case:

Definition 3 (Reduct of a Reducible MCS). *Let $M = \langle C_1, \ldots, C_n \rangle$ be a reducible MCS and $S = \langle S_1, \ldots, S_n \rangle$ a belief state of M. The* S-reduct of M *is*

$$M^S = \langle C_1^S, \ldots, C_n^S \rangle$$

where, for each $C_i = (L_i, kb_i, br_i)$, we define $C_i^S = (L_i, red_i(kb_i, S_i), br_i^S)$. Here br_i^S results from br_i by deleting

1. *every rule with some* $\mathbf{not}\ (r : p)$ *in the body such that $S \models (r : p)$, and*
2. *all* $\mathbf{not}$ *literals from the bodies of remaining rules.*

For each MCS M and each belief set S, we have that the S-reduct of M is definite. We can thus check whether S is a grounded equilibrium in the usual manner:

Definition 4 (Grounded Equilibrium). *Let $M = \langle C_1, \ldots, C_n \rangle$ be a reducible MCS and $S = \langle S_1, \ldots, S_n \rangle$ a belief state of M. S is a* grounded equilibrium of M *if S is the grounded equilibrium of M^S, that is $S = \mathbf{GE}(M^S)$.*

For example, let us model a multi-agent system in which the agent b_1 aims to purchase product item i_1. For simplicity assume that there are only two agents, s_1 and s_2, offering products for sale. Agent s_1 readily offers i_1 for 30 credits and agent s_2 also currently offers i_1, but for 35 credits. Agent b_1 may model this by the following two contexts:

$$\begin{array}{lll} s_1: & \mathsf{available}(i_1) \leftarrow . & \qquad s_2: \quad \mathsf{available}(i_1) \leftarrow . \\ & \mathsf{price}(i_1, 30) \leftarrow . & \qquad\quad\ \ \mathsf{price}(i_1, 35) \leftarrow . \end{array}$$

Agent b_1 represents its business rules as context b_1. First, it imports information from s_1 and s_2 using bridge rules:[3]

$$\begin{array}{ll} b_1: & \mathsf{offers}(A, I, P) \leftarrow A : \mathsf{available}(I), A : \mathsf{price}(I, P). \\ & \mathsf{best\_price}(I, X) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X \leq Y. \\ & \mathsf{best\_price}(I, Y) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X > Y. \end{array}$$

Finally, agent b_1 uses additional rules to implement its own business logic. For instance, it can be a good strategy to go for the best price unless there is a very good reason to buy from some other agent. This is implemented by adding the rule into b_1:

$$\mathsf{purchase}(I, A) \leftarrow \mathsf{offers}(A, I, P), \mathsf{best\_price}(I, P), {\sim}\mathsf{worth\_buy}(B). \qquad (1)$$

[3] A bridge rule with a variable (A) in place of the context identifier stands for the set of ground instances obtained by replacing A with context identifiers in all possible ways. The predicates $<, \leq$ and the function $+$ (meaning less, less or equal relations and addition on numeric domains) can be formalized in logic programming and we abstract from this for space reasons. Also, due to space reasons the computation of best price is slightly simplified given the case of two selling agents, which can be easily extended in case of multiple agents.

Indeed, sometimes it might be worth not to buy for the best price in the long term. If b_1 has long standing business relations with some supplier, it may be worth purchasing some items even for a slightly higher price assuming that later on it may be rewarded with a discount or other value. Hence assuming that s_2 is the valuable supplier we add the following rules into b_1:

$$\mathsf{worth\_buy}(A) \leftarrow \mathsf{best\_price}(I,P), \mathsf{offers}(A,I,Q), \mathsf{good\_supplier}(A), Q \leq P+5. \quad (2)$$

$$\mathsf{purchase}(I,A) \leftarrow \mathsf{offers}(A,I,P), \mathsf{worth\_buy}(A). \quad (3)$$

$$\mathsf{good\_supplier}(s_2) \leftarrow . \quad (4)$$

The MCS $M = \langle b_1, s_1, s_2 \rangle$ has a single grounded equilibrium $S = \langle B_1, S_1, S_2 \rangle$ such that $\mathsf{purchase}(i_1, s_2) \in B_1$ and $\mathsf{purchase}(i_1, s_1) \notin B_1$. So the agent actually reasons according to the business strategy that we described.

2.4 MKNF Knowledge Bases

MKNF Knowledge Bases [15] are based on the logic of Minimal Knowledge and Negation as Failure (MKNF) [14], an extension of first-order logic with two modal operators: $\mathbf{K}$ and $\mathbf{not}$. We use the variant of this logic introduced in [15]. *MKNF sentences* and *theories* are defined by extending function-free first-order syntax by the mentioned modal operators in a natural way.

As in [15], we assume that the set of constant symbols $\mathbf{C}$ is infinite and consider only *Herbrand interpretations* that interpret the equality predicate $\approx$ as a congruence relation on $\mathbf{C}$. The set of all such interpretations is denoted by $\mathbf{I}$. An *MKNF structure* is a triple $(I, \mathcal{M}, \mathcal{N})$ where $I \in \mathbf{I}$ and $\mathcal{M}, \mathcal{N} \subseteq \mathbf{I}$.[4] Intuitively, the first component is used to interpret the first-order parts of an MKNF sentence while the other two components interpret the $\mathbf{K}$ and $\mathbf{not}$ modalities, respectively. By $\phi[a/x]$ we denote the formula obtained from ϕ by replacing every unbound occurrence of variable x with the constant symbol a. The satisfaction of an MKNF sentence and an MKNF theory $\mathcal{T}$ in an MKNF structure $(I, \mathcal{M}, \mathcal{N})$ is defined as follows:

$$\begin{array}{lll}
(I,\mathcal{M},\mathcal{N}) \models p & \text{iff} & I \models p \\
(I,\mathcal{M},\mathcal{N}) \models \neg\phi & \text{iff} & (I,\mathcal{M},\mathcal{N}) \not\models \phi \\
(I,\mathcal{M},\mathcal{N}) \models \phi_1 \wedge \phi_2 & \text{iff} & (I,\mathcal{M},\mathcal{N}) \models \phi_1 \text{ and } (I,\mathcal{M},\mathcal{N}) \models \phi_2 \\
(I,\mathcal{M},\mathcal{N}) \models \exists x : \phi & \text{iff} & (I,\mathcal{M},\mathcal{N}) \models \phi[a/x] \text{ for some } a \in \mathbf{C} \\
(I,\mathcal{M},\mathcal{N}) \models \mathbf{K}\,\phi & \text{iff} & (J,\mathcal{M},\mathcal{N}) \models \phi \text{ for all } J \in \mathcal{M} \\
(I,\mathcal{M},\mathcal{N}) \models \mathbf{not}\,\phi & \text{iff} & (J,\mathcal{M},\mathcal{N}) \not\models \phi \text{ for some } J \in \mathcal{N} \\
(I,\mathcal{M},\mathcal{N}) \models \mathcal{T} & \text{iff} & (I,\mathcal{M},\mathcal{N}) \models \phi \text{ for all } \phi \in \mathcal{T}
\end{array}$$

The symbols $\top$, $\bot$, $\vee$, $\forall$ and $\supset$ are interpreted accordingly. Also, for any $\mathcal{M} \subseteq \mathbf{I}$ we write $\mathcal{M} \models \mathcal{T}$ if $(I, \mathcal{M}, \mathcal{M}) \models \mathcal{T}$ for all $I \in \mathcal{M}$. An *MKNF interpretation* $\mathcal{M}$ is a

[4] Differently from [15], we allow for empty $\mathcal{M}, \mathcal{N}$ in this definition as later on it will be useful to have satisfaction defined for this marginal case.

non-empty subset of $\mathbf{I}$.[5] By $\mathbf{M} = 2^{\mathbf{I}}$ we denote the set of all MKNF interpretations together with the empty set. The semantics of MKNF theories is defined as follows:

Definition 5 (MKNF Semantics). *Let $\mathcal{T}$ be an MKNF theory. We say that an MKNF interpretation $\mathcal{M}$ is*

- *an* S5 model of $\mathcal{T}$ *if* $\mathcal{M} \models \mathcal{T}$*;*
- *an* MKNF model of $\mathcal{T}$ *if $\mathcal{M}$ is an S5 model of $\mathcal{T}$ and for every MKNF interpretation $\mathcal{M}' \supsetneq \mathcal{M}$ there is some $I' \in \mathcal{M}'$ such that $(I', \mathcal{M}', \mathcal{M}) \not\models \mathcal{T}$.*

MKNF knowledge bases [15] consist of two components – an ontology $\mathcal{O}$ and a program $\mathcal{P}$ – and their semantics is given by translation to an MKNF theory. In the following we introduce the syntax and semantics of MKNF knowledge bases in which we constrain the program component to a normal logic program.[6]

An *MKNF knowledge base* is a set $\mathcal{K} = \mathcal{O} \cup \mathcal{P}$ where $\mathcal{O}$ is an ontology and $\mathcal{P}$ is a logic program. An MKNF knowledge base is *ground* if $\mathcal{P}$ is ground; *definite* if $\mathcal{P}$ is definite. The grounding of an MKNF knowledge base $\mathcal{K}$ is defined as $\mathsf{gr}(\mathcal{K}) = \mathcal{O} \cup \mathsf{gr}(\mathcal{P})$.

The translation function κ is defined for all literals l, default literals $\sim l$, sets of literals S, rules π with vector of free variables $\boldsymbol{x}$, programs $\mathcal{P}$ and MKNF knowledge bases $\mathcal{K} = \mathcal{O} \cup \mathcal{P}$ as follows: $\kappa(l) = \mathbf{K}\, l$, $\kappa(\sim l) = \mathbf{not}\, l$, $\kappa(S) = \bigwedge \{\, \kappa(L) \mid L \in S \,\}$, $\kappa(\pi) = (\kappa(B(\pi)) \supset \kappa(H(\pi)))$, $\kappa(\mathcal{P}) = \{\, \kappa(\pi) \mid \pi \in \mathcal{P} \,\}$, $\kappa(\mathcal{K}) = \{\, \kappa(\mathcal{O}) \,\} \cup \kappa(\mathcal{P})$.

The semantics of MKNF knowledge bases is thus defined as follows:

Definition 6 (Semantics of MKNF Knowledge Bases). *Let $\mathcal{K}$ be an MKNF knowledge base. We say that an MKNF interpretation $\mathcal{M}$ is an* S5 model of $\mathcal{K}$ *if $\mathcal{M}$ is an S5 model of $\kappa(\mathcal{K})$. Similarly, $\mathcal{M}$ is an* MKNF model of $\mathcal{K}$ *if $\mathcal{M}$ is an MKNF model of $\kappa(\mathcal{K})$.*

In the normative part of our running example, DL axioms could be used to classify different products depending on their availability on the market into available and unavailable, but also into products that can be directly purchased and special products for which licitation is required. Then, rules will be used to evaluate if a particular product can be purchased by some agent or not (e.g., normally the product can be purchased whenever it is available, but in the exceptional case when licitation is required additional conditions must be satisfied). Note that while DL axioms are required to encode the classification, non-monotonic rules are required to express exceptions. This can be encoded in the MKNF knowledge base n composed of the ontology $\mathcal{O}_n$ and the rule set $\mathcal{P}_n$:

[5] Notice that if $\mathcal{M}$ is empty, then it vacuously holds that $\mathcal{M} \models \phi$ for all sentences ϕ. For this reason, and in accordance with [15], $\emptyset$ is not considered an MKNF interpretation.

[6] Note that we do not directly include the $\mathbf{K}$ and $\mathbf{not}$ modalities in rules of the MKNF knowledge base; instead, they are introduced by the translation function κ (denoted by π in [15]) upon translation to an MKNF theory. Also, unlike in [15], κ is overridden to accept atoms, literals and sets of literals and produces an MKNF theory instead of an MKNF sentence. Thus we do not need to assume that the program is finite and can deal with infinite ground programs that result from grounding a finite but non-ground program (the same is actually done in [15] from Section 4 onwards).

$\mathcal{O}_n$: $\leqslant 0\,\mathsf{offered\_by.supplier} \sqsubseteq \mathsf{unavailable\_product}$
$=1\,\mathsf{offered\_by.supplier} \sqsubseteq \mathsf{purchasable\_product}$
$\geqslant 2\,\mathsf{offered\_by.supplier} \sqsubseteq \mathsf{licitable\_product}$

$\mathcal{P}_n$: $\mathsf{purchase\_allowed}(I, A) \leftarrow \mathsf{offered\_by}(I, A), {\sim}\mathsf{licitable\_product}(I)$
$\mathsf{purchase\_allowed}(I, A). \leftarrow \mathsf{licitable\_product}(I), \mathsf{offered\_by}(I, A), \mathsf{best\_price}(I, A).$

3 Reducing an MKNF Context

MKNF knowledge bases can be used within Multi-Context Systems by specifying an *MKNF context* that uses the *MKNF logic*. We formalise these notions below. Subsequently, we show that every MKNF context can be transformed into a *first-order context*. The transformed MCS has the same grounded equilibria as the original one, showing that instead of a specialised MKNF reasoner, a first-order reasoner can be used to obtain equivalent results. Subsequently, we show that this result can be strengthened even further, resulting in a multi-context system that requires only a DL reasoner instead of an MKNF reasoner provided all bridge literals referring to that context are expressible in the DL in consideration. In this case, a DL reasoner is basically all that is necessary to handle reasoning with MKNF contexts within a multi-context system.

3.1 MKNF Contexts and First-Order Contexts

We start by formalising what an *MKNF context* actually is. The *MKNF logic* is the logic $L_{\text{MKNF}} = (\mathbf{KB}_{\text{MKNF}}, \mathbf{BS}_{\text{MKNF}}, \mathbf{ACC}_{\text{MKNF}})$ where

- $\mathbf{KB}_{\text{MKNF}}$ is the set of MKNF knowledge bases,
- $\mathbf{BS}_{\text{MKNF}}$ is the set of deductively closed sets of first-order sentences,
- $\mathbf{ACC}_{\text{MKNF}}(\mathcal{K})$ contains $\{\, \phi \in \Phi \mid \mathcal{M} \models \phi \,\}$ for every MKNF model $\mathcal{M}$ of $\mathcal{K}$ and also the inconsistent belief set Φ in case $\mathcal{K}'$, obtained from $\mathcal{K}$ by removing all rules with default negation, has no MKNF model.

The latter condition is required to adhere to the formal framework of multi-context systems. An *MKNF context* can now be established as follows:

Definition 7 (MKNF Context). *A context $C = (L, kb, br)$ is an* MKNF context *if $L = L_{\text{MKNF}}$, kb is an MKNF knowledge base, and, for every $\sigma \in br$, $H(\sigma)$ is either an ontology axiom or a definite rule. We also say that such σ is an* MKNF bridge rule.

An MKNF context $C = (L_{\text{MKNF}}, \mathcal{K}, br)$ is ground *if $\mathcal{K}$ is ground and all rules in heads of MKNF bridge rules from br are ground;* finite *if both $\mathcal{K}$ and br are finite.*

In order to talk about grounded equilibria of multi-context systems with MKNF contexts, their reducibility must be guaranteed. Given our definition above, this is indeed the case:

Proposition 8. *Every MKNF context is reducible.*

We are now able to plug the normative MKNF KB n into our example MCS using the bridge rule set br_n:

$$\begin{aligned} br_n\colon\ & \mathsf{supplier}(A) \leftarrow A : \mathsf{available}(I). \\ & \mathsf{offered\_by}(I, A) \leftarrow A : \mathsf{available}(I). \\ & \mathsf{best\_price}(I, s_1) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X \leq Y. \\ & \mathsf{best\_price}(I, s_2) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X > Y. \end{aligned}$$

Agent b_1 is now required to reason with the norms. To implement this, the following bridge rule is added to b_1:

$$\mathsf{allowed}(I, A) \leftarrow n : \mathsf{purchase\_allowed}(I, A).$$

Additionally, the business rules (1) and (3) need to be altered (in the respective order):

$$\mathsf{purchase}(I, A) \leftarrow \mathsf{offers}(A, I, P), \mathsf{allowed}(I, A), \mathsf{best\_price}(I, P), {\sim}\mathsf{worth\_buy}(B). \quad (5)$$

$$\mathsf{purchase}(I, A) \leftarrow \mathsf{offers}(A, I, P), \mathsf{allowed}(I, A), \mathsf{worth\_buy}(A). \quad (6)$$

The agent will now take the norms into account during reasoning. The updated MCS $M^n = \langle b_1, s_1, s_2, n\rangle$ has a single grounded equilibrium $S^n = \langle B_1^n, S_1^n, S_2^n, N^n\rangle$, however, neither $\mathsf{purchase}(i_1, s_1)$ nor $\mathsf{purchase}(i_1, s_2)$ belong to B_1^n. Indeed the agent's business strategy is now in conflict with the norms, it cannot favour the valuable supplier any longer if there is a better momentary offer.

To fully accommodate the norms while keeping the agent operational, we would have to drop ${\sim}\mathsf{worth\_buy}(B)$ from the rule (5) and completely drop the rule (6) which is now obsolete, it represents an invalid strategy. Or if in turn we chose to ignore the norms (and bear eventual consequences) we might drop $\mathsf{allowed}(I, A)$ from both rules and return to the previous purchasing strategy.

To reduce an MKNF context to a simpler one, we define *first-order contexts*. The *first-order logic* is the logic $L_{\text{FO}} = (\mathbf{KB}_{\text{FO}}, \mathbf{BS}_{\text{FO}}, \mathbf{ACC}_{\text{FO}})$ where

- $\mathbf{KB}_{\text{FO}}$ is the set of first-order theories,
- $\mathbf{BS}_{\text{FO}}$ is the set of deductively closed sets of first-order sentences,
- $\mathbf{ACC}_{\text{FO}}(\mathcal{T}) = \{\, \phi \in \Phi \mid \mathcal{T} \models \phi \,\}$.

A *first-order context* is henceforth defined as follows:

Definition 9 (First-Order Context). *A context $C = (L, kb, br)$ is a* first-order context *if $L = L_{\text{FO}}$, kb is a first-order theory, and, for every $\sigma \in br$, $H(\sigma)$ is a first-order sentence. We also say that such σ is a* first-order bridge rule.

The first-order logic is monotonic, so every first-order context is reducible.

Proposition 10. *Every first-order context is reducible.*

3.2 Reduction to a First-Order Context

We are now ready to introduce a transformation of an MKNF context to a corresponding first-order context. This transformation is based on transforming the rules from

the MKNF knowledge base to bridge rules, leaving us with only the ontology component which can already be handled by a first-order context. For example, if the MKNF knowledge base in the MKNF context C_j contains the rule

$$p \leftarrow q, {\sim}r.$$

the corresponding first-order bridge rule is of the form

$$p \leftarrow (j : q), \mathbf{not}\,(j : r).$$

Furthermore, since bridge rules in an MKNF context may have definite rules in their heads, we also need to transform them in some way so that the resulting bridge rules are compatible with the first-order context. This is done by moving the body of the definite rule π in the head of a bridge rule σ to the body of σ. For example, if the MKNF context C_j contains the MKNF bridge rule

$$(p \leftarrow q.) \leftarrow (i_1 : q), \mathbf{not}\,(i_2 : r).$$

then the transformed first-order bridge rule will be

$$p \leftarrow (j : q), (i_1 : q), \mathbf{not}\,(i_2 : r).$$

In case of MKNF bridge rules that have an ontology axiom ϕ in their head, this axiom needs to be translated to its first-order counterpart $\kappa(\phi)$. For all ground rules these transformations can be formalised as follows:

Definition 11 (Transformation to First-Order Bridge Rules). *Let j be an integer. We introduce the following notation for every $l \in \mathbf{L_G}$ and $S \subseteq \mathbf{L_G}$: $\beta_j(l) = (j : l)$, $\beta_j({\sim}l) = \mathbf{not}\,(j : l)$, $\beta_j(S) = \{\, \beta_j(L) \mid L \in S \,\}$.*

Let $C = (L_{\text{MKNF}}, \mathcal{O} \cup \mathcal{P}, br)$ be a ground MKNF context. For every rule $\pi \in \mathcal{P}$, $\beta_j(\pi)$ denotes the first-order bridge rule

$$H(\pi) \leftarrow \beta_j(B(\pi)).$$

Furthermore, for every MKNF bridge rule $\sigma \in br$ of the form $(\pi \leftarrow B(\sigma).)$, where π is a definite rule, $\beta_j(\sigma)$ denotes the first-order bridge rule

$$H(\pi) \leftarrow \beta_j(B(\pi)) \cup B(\sigma).$$

and for every MKNF bridge rule $\sigma \in br$ of the form $(\phi \leftarrow B(\sigma).)$, where ϕ is an ontology axiom, $\beta_j(\sigma)$ denotes the first-order bridge rule

$$\kappa(\phi) \leftarrow B(\sigma).$$

Also, for any set of rules or MKNF bridge rules S, $\beta_j(S) = \{\, \beta_j(\pi) \mid \pi \in S \,\}$.

The definition of the first-order context that corresponds to an MKNF context is now straightforward – it suffices to apply the above transformation β_j to all rules and MKNF bridge rules of the MKNF context:

Definition 12 (First-Order Context Corresponding to MKNF Context). *Let* $C_j = (L_{\text{MKNF}}, \mathcal{O} \cup \mathcal{P}, br_j)$ *be a ground MKNF context. The* first-order context corresponding to C_j *is* $C_j^{\text{FO}} = (L_{\text{FO}}, \{\, \kappa(\mathcal{O}) \,\}, \beta_j(br_j) \cup \beta_j(\mathcal{P}))$.

Due to the properties of the MKNF semantics, when we consider a finite ground MKNF context, which is usually the most interesting case in applications, we find that it can be substituted by the corresponding first-order context without affecting the grounded equilibria of the multi-context system. Formally:

Theorem 13 (Reduction Into First-Order Context). *Let* $M = \langle C_1, \ldots, C_n \rangle$ *be a multi-context system such that for some* j *with* $1 \leq j \leq n$, C_j *is a finite ground MKNF context and put*

$$M' = \langle C_1, \ldots, C_{j-1}, C_j^{\text{FO}}, C_{j+1}, \ldots, C_n \rangle \ .$$

The grounded equilibria of M *and* M' *coincide.*

This transformation can be repeated for all MKNF contexts in the multi-context system, yielding an equivalent system that does not require to use the MKNF logic. Formally:

Corollary 14. *For every multi-context system* M *with some finite ground MKNF contexts there exists a multi-context system* M' *such that the grounded equilibria of* M *and* M' *coincide and* M' *uses first-order contexts instead of the original MKNF contexts.*

Revisiting our running example, the MCS M^n is reduced into $M^{n\prime} = \langle b_1, s_1, s_2, n^{\text{FO}} \rangle$ where $n^{\text{FO}} = (L_{\text{FO}}, \{\, \kappa(\mathcal{O}_n) \,\}, \beta_n(br_n) \cup \beta_n(\mathcal{P}_n))$:

$$\begin{aligned}
\kappa(\mathcal{O}_n)\colon\ & \forall X \forall Y\, \neg \mathsf{offered_by}(X,Y) \vee \neg \mathsf{supplier}(Y) \implies \mathsf{unavailable_product}(X) \\
& \forall X \exists Y\, \mathsf{offered_by}(X,Y) \wedge \mathsf{supplier}(Y) \wedge (\forall Z (Z = Y) \vee \neg \mathsf{offered_by}(X,Z) \\
& \qquad \vee \neg \mathsf{supplier}(Z)) \implies \mathsf{purchasable_product}(X) \\
& \forall X \exists Y \exists Z (Y \neq Z) \wedge \mathsf{offered_by}(X,Y) \wedge \mathsf{supplier}(Y) \wedge \mathsf{offered_by}(X,Z) \\
& \qquad \wedge \mathsf{supplier}(Z) \implies \mathsf{licitable_product}(X) \\
\beta_n(\mathcal{P}_n)\colon\ & \mathsf{purchase_allowed}(I,A) \leftarrow n : \mathsf{offered_by}(I,A), \mathbf{not}\ (n : \mathsf{licitable_product}(I)). \\
& \mathsf{purchase_allowed}(I,A) \leftarrow n : \mathsf{licitable_product}(I), n : \mathsf{offered_by}(I,A), \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad n : \mathsf{best_price}(I,A).
\end{aligned}$$

and $\beta_n(br_n) = br_n$.

3.3 Translation Into Two Contexts

To be able to translate an MKNF context into a *DL context* we need to define DL contexts first. The *DL logic* is the logic $L_{\text{DL}} = (\mathbf{KB}_{\text{DL}}, \mathbf{BS}_{\text{DL}}, \mathbf{ACC}_{\text{DL}})$ where

- $\mathbf{KB}_{\text{DL}}$ is the set of all ontologies;
- $\mathbf{BS}_{\text{DL}}$ is the set of all deductively closed sets of first-order sentences;
- $\mathbf{ACC}_{\text{DL}}(\mathcal{O})$ returns the set of first-order sentences ϕ such that $\mathcal{O} \models \phi$.

A *DL context* is obtained as follows:

Definition 15 (DL Context). *A context $C = (L, kb, br)$ is a* DL context *if $L = L_{\text{DL}}$, kb is an ontology $\mathcal{O}$, and, for every $\sigma \in br$, $H(\sigma)$ is a DL-axiom. We also say that such σ is a* DL bridge rule.

DLs as fragments of first-order logic are monotonic, so DL contexts are reducible.

Proposition 16. *Every DL context is reducible.*

Since DLs only make use of unary and binary predicates, and an MKNF context is not limited to these due to the presence of atoms over predicates whose arity is greater than 2, we need an additional context here that serves as a means to store and retrieve facts. For simplicity, we introduce an abstract *fact base context* and its associated logic. The *fact base logic* is the logic $L_{\text{FB}} = (\mathbf{KB}_{\text{FB}}, \mathbf{BS}_{\text{FB}}, \mathbf{ACC}_{\text{FB}})$ where

- $\mathbf{KB}_{\text{FB}}$ is the set of subsets of $\mathbf{L}_{\mathsf{G}}$;
- $\mathbf{BS}_{\text{FB}}$ is the set of subsets of $\mathbf{L}_{\mathsf{G}}$;
- $\mathbf{ACC}_{\text{FB}}(kb)$ is the identity function.

A *fact base context* is established as follows:

Definition 17 (Fact Base Context). *A context $C = (L, kb, br)$ is a* fact base context *if $L = L_{\text{FB}}$, kb is a subset of $\mathbf{L}_{\mathsf{G}}$, and, for every $\sigma \in br$, $H(\sigma)$ is $l \in \mathbf{L}_{\mathsf{G}}$. We also say that such σ is a* fact base bridge rule.

A fact base context is obviously reducible.

Proposition 18. *Every fact base context is reducible.*

Essentially, the idea is to translate an MKNF context C_j into a pair of contexts, namely a DL context and a fact base context, which is possible under certain restrictions as specified below. This can be achieved by first considering the reduction into a first-order context, which contains only bridge rules and $\kappa(\mathcal{O})$. The bridge rules can be divided between the two contexts depending on whether the literal in the head appears only in the rules or not. In the former case such a head can never be used for reasoning in the DL context, hence, all these rules are added to the fact base context, while in the latter case, they are added to the DL context. Additionally, in all bridge literals that do not appear in $\mathcal{O}$, the pointer is changed to the fact base context.

We formalize this, by first defining this division of literals appearing in the MKNF context, i.e. in the corresponding MKNF KB. Given an MKNF knowledge base $\mathcal{K} = \mathcal{O} \cup \mathcal{P}$, we define that $l \in \mathbf{L}_{\mathsf{G}}$ is a DL-literal if the predicate of l appears in $\mathcal{O}$. Otherwise, l is a non-DL literal.

Now we can define an abstract function that can be used to transform the bridge literals in a given first-order context, with the intention that bridge rules with a non-DL-atom in the head point to a different context k.

Definition 19 (Transformation to Two-Context Bridge Rules). *Let j and k be integers and S a set of bridge rules. We define $\beta_j^k(S) = \{\, \beta_j^k(\pi) \mid \pi \in S \,\}$. For every $\pi \in S$ $\beta_j(\pi)$ denotes the bridge rule $H(\pi) \leftarrow \beta_j^k(B(\pi))$. We define for sets of bridge literals S that $\beta_j^k(S) = \{\, \beta_j^k(L) \mid L \in S \,\}$. Moreover, we define for single bridge literals $(j : l)$ and $\mathbf{not}\,(j : l)$:*

- $\beta_j^k((j:l)) = (k:l)$ *and* $\beta_j^k(\mathbf{not}\,(j:l)) = \mathbf{not}\,(k:l)$ *if* l *is a non-DL-atom;*
- $\beta_j^k((j:l)) = (j:l)$ *and* $\beta_j^k(\mathbf{not}\,(j:l)) = \mathbf{not}\,(j:l)$ *otherwise.*

Finally, $\beta_j^k((i:l)) = (i:l)$ *and* $\beta_j^k((i:l)) = (i:l)$ *for* $1 \leq i \neq j \leq n$.

Note that the second case not only handles DL-atoms, but also arbitrary first-order sentences, which also covers translations of DL axioms.

We define the two-context DL MCS corresponding to an MKNF context.

Definition 20 (Two-Context DL MCS Corresponding to an MKNF Context). *Let* $C_j = (L_{\text{MKNF}}, \mathcal{O} \cup \mathcal{P}, br_j)$ *be a ground MKNF context.The* two-context DL MCS corresponding to C_j, $\langle C_j^{\text{DL}}, C_k^{\text{FB}} \rangle$, *is defined as follows:*

- $C_j^{\text{DL}} = (L_{\text{DL}}, \mathcal{O}, br_j)$;
- $br_j = \{H(\pi) \leftarrow \beta_j^k(B(\pi)) \mid \pi \in (\beta_j(br_j) \cup \beta_j(\mathcal{P})) \wedge H(\pi)$ *is a DL-atom fact*$\} \cup \{\phi \leftarrow \beta_j^k(B(\pi)) \mid \pi \in br_j \wedge H(\pi)$ *is an ontology axiom* $\phi\}$;
- $C_k^{\text{FB}} = (L_{\text{FB}}, \emptyset, br_k)$;
- $br_k = \{H(\pi) \leftarrow \beta_j^k(B(\pi)) \mid \pi \in (\beta_j(br_j) \cup \beta_j(\mathcal{P})) \wedge H(\pi)$ *is a non-DL-atom fact*$\}$.

Bridge rules in $(\beta_j(br_j) \cup \beta_j(\mathcal{P}))$ are divided between the two contexts as outlined. The only exception are bridge rules in br_j with an ontology axiom in the head. These are added to the DL context. Note that the index k for the fact base context allows us to add C_k^{FB} to an MCS with n contexts at a position of choice, which is $n+1$.

Given a belief set S, S^* denotes the deductive closure of S. We can substitute a finite, ground MKNF context with a two-context MCS without affecting the grounded equilibria provided that certain conditions hold.

Theorem 21 (Translation Two-Context DL MCS). *Let* $M = \langle C_1, \ldots, C_n \rangle$ *be a reducible multi-context system such that, for some* j *with* $1 \leq j \leq n$, C_j *is a finite ground MKNF context, and, for all* i *with* $1 \leq i \leq n$, *all* $\pi \in br_i$, *and each* $(j:l) \in \pi$ *and* $\mathbf{not}\,(j:l) \in \pi$, l *is an objective literal. Let* k *be* $n+1$, *and set*

$$M' = \langle C_1', \ldots, C_{j-1}', C_j^{\text{DL}}, C_{j+1}', \ldots, C_n', C_{n+1}^{\text{FB}} \rangle$$

where, for all i *with* $1 \leq j \neq i \leq n$, C_i' *results from* C_i *by applying* β_j^{n+1} *to* br_i.

The grounded equilibria of M *and* M' *are equivalent, i.e., for all* i *with* $1 \leq i \neq j \leq n$, $S_i = S_i'$, *and* $S_j = (S_j^{\text{DL}} \cup S_{n+1}^{\text{FB}})^*$.

The restriction on bridge literals in MCS given in Theorem 21 could be seen as being too severe, i.e. it is possible to come up with a less restrictive result that only limits these bridge literals to a degree such that they are not arbitrary first-order sentences, but either non-DL atoms or formulas that are in the belief set of the DL in consideration. We claim that these more restrictive conditions are sufficient here, in particular in light of the kind of rules we allow in MKNF knowledge bases, and leave lifting the result to more expressive MKNF knowledge bases (and more expressive bridge rules) for future work.

Again, this transformation can be repeated for all MKNF contexts in the multi-context system, yielding an equivalent system that does not require to use the MKNF logic if the bridge literals to the MKNF context are appropriately restricted. Formally:

Corollary 22. *For every multi-context system M with some finite ground MKNF contexts such that all bridge literals to these contexts are objective literals, there exists a multi-context system M' such that the grounded equilibria of M and M' are equivalent (in the sense of Theorem 21) and M' uses pairs of DL contexts and fact base contexts instead of the original MKNF contexts.*

Turning again to our running example, if we choose the two-context translation instead of the previous one, Translating M^n is reduced into $M^{n''} = \langle b_1, s_1, s_2, n^{\text{DL}}, m^{\text{FB}} \rangle$ where $n^{\text{DL}} = (L_{\text{DL}}, \mathcal{O}_n, br_n)$ and $m^{\text{FB}} = (L_{\text{FB}}, \emptyset, br_m)$ where $\mathcal{O}_n$ is as defined above and br_n and br_m are as follows:

$$
\begin{aligned}
br_n\colon\ & \mathsf{supplier}(A) \leftarrow A : \mathsf{available}(I).\\
& \mathsf{offered\_by}(I, A) \leftarrow A : \mathsf{available}(I).\\
br_m\colon\ & \mathsf{best\_price}(I, s_1) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X \leq Y.\\
& \mathsf{best\_price}(I, s_2) \leftarrow s_1 : \mathsf{price}(I, X), s_2 : \mathsf{price}(I, Y), X > Y.\\
& \mathsf{purchase\_allowed}(I, A) \leftarrow n : \mathsf{offered\_by}(I, A), \mathbf{not}\ (n : \mathsf{licitable\_product}(I)).\\
& \mathsf{purchase\_allowed}(I, A) \leftarrow n : \mathsf{licitable\_product}(I), n : \mathsf{offered\_by}(I, A),\\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad m : \mathsf{best\_price}(I, A).
\end{aligned}
$$

4 Conclusions

Open multi agent systems can be modelled in MCSs, and we have considered MKNF knowledge bases as one such context, which allows the usage of a highly expressive knowledge representation and reasoning formalism in such MCSs. One immediate result of our work is that the Hybrid MKNF [15] can indeed be used in MCSs, namely in the form of MKNF contexts, which can then be interlinked with other contexts in the MCS.

Since not all agents may necessarily be able to reason with such a context, we investigated the possibility of using the expressiveness of MKNF knowledge bases without having to use an actual MKNF context. We showed that an MKNF context can be reduced to an associated first-order context without any effects on the semantics, i.e. the grounded equilibria of the considered MCS. Hence, a first order reasoner can be used instead. Moreover, restricting the form of bridge literals to objective literals, we have shown that we can even use a combination of a DL reasoner and a simple store for (rule) facts to achieve the same result. In the former case, the resulting MCS is more general, while in the latter case we are enabled to use a decidable and faster reasoner.

Future work includes loosening the restriction in Theorem 21 such that bridge literals may contain more expressive formulas w.r.t. the DL context. In line with this lies the extension of the results to more general MKNF KBs, i.e. where objective literals in MKNF rules may not just be atoms and their (classical) negations. Another line of work would be to consider substituting an MKNF context with two contexts where one context is a DL-context and the other one in ASP. In this case, contrary to our translation into two contexts, the treatment of non-monotonic rules would be literally hidden in the context and it would be interesting to compare these different two-context translations. In [6], also a well-founded semantics is defined for MCSs and considering this

semantics and investigating its correlation with the well-founded semantics for Hybrid MKNF [13] would also be interesting possibly enabling us to use a semantics in MCSs that is, due to its nature, of a lower computational complexity.

Acknowledgments. Matthias Knorr, João Leite and Martin Slota were partially supported by Fundação para a Ciência e a Tecnologia under project "ERRO – Efficient Reasoning with Rules and Ontologies" (PTDC/EIA-CCO/121823/2010). Martin Homola was partially supported by the Slovak national project VEGA no. 1/1333/12. The collaboration between the co-authors resulted from the Slovak–Portuguese bilateral project "ReDIK – Reasoning with Dynamic Inconsistent Knowledge", supported by the APVV agency under SK-PT0-0028-10 and by Fundação para a Ciência e a Tecnologia (FCT/2487/3/6/2011/S).

References

1. Alberti, M., Gomes, A.S., Gonçalves, R., Knorr, M., Leite, J., Slota, M.: Normative systems require hybrid knowledge bases (extended abstract). In: Conitzer, V., Winikoff, M., Padgham, L., van der Hoek, W. (eds.) Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012), June 4-8. IFAAMAS, Valencia, Spain (2012)
2. Alberti, M., Gomes, A.S., Gonçalves, R., Leite, J., Slota, M.: Normative Systems Represented as Hybrid Knowledge Bases. In: Leite, J., Torroni, P., Ågotnes, T., Boella, G., van der Torre, L. (eds.) CLIMA XII 2011. LNCS, vol. 6814, pp. 330–346. Springer, Heidelberg (2011)
3. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications, 2nd edn. Cambridge University Press (2007)
4. Benerecetti, M., Cimatti, A., Giunchiglia, E., Giunchiglia, F., Serafini, L.: Formal Specification of Beliefs in Multi-Agent Systems. In: Jennings, N.R., Wooldridge, M.J., Müller, J.P. (eds.) ECAI-WS 1996 and ATAL 1996. LNCS, vol. 1193, pp. 117–130. Springer, Heidelberg (1997)
5. Benerecetti, M., Giunchiglia, F., Serafini, L.: Model checking multiagent systems. Journal of Logic and Computation 8(3), 401–423 (1998)
6. Brewka, G., Eiter, T.: Equilibria in heterogeneous nonmonotonic multi-context systems. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, July 22-26, pp. 385–390. AAAI Press, Vancouver (2007)
7. Casali, A., Godo, L., Sierra, C.: Graded BDI Models for Agent Architectures. In: Leite, J., Torroni, P. (eds.) CLIMA 2004. LNCS (LNAI), vol. 3487, pp. 126–143. Springer, Heidelberg (2005)
8. Cimatti, A., Serafini, L.: Multi-Agent Reasoning with Belief Contexts: The Approach and a Case Study. In: Wooldridge, M.J., Jennings, N.R. (eds.) ECAI 1994 and ATAL 1994. LNCS, vol. 890, pp. 71–85. Springer, Heidelberg (1995)
9. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Kowalski, R.A., Bowen, K.A. (eds.) Proceedings of the 5th International Conference and Symposium on Logic Programming (ICLP/SLP 1988), August 15-19, pp. 1070–1080. MIT Press, Seattle (1988)
10. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Computing 9(3-4), 365–385 (1991)

11. Ghidini, C., Giunchiglia, F.: Local models semantics, or contextual reasoning=locality+compatibility. Artificial Intelligence 127(2), 221–259 (2001)
12. Giunchiglia, F.: Contextual reasoning. Epistemologia - Special Issue on I Linguaggi e le Macchine XVI, 345–364 (1993)
13. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artificial Intelligence 175(9-10), 1528–1554 (2011)
14. Lifschitz, V.: Nonmonotonic databases and epistemic queries. In: Mylopoulos, J., Reiter, R. (eds.) Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI 1991), pp. 381–386. Morgan Kaufmann, Sydney, Australia (1991)
15. Motik, B., Rosati, R.: Reconciling description logics and rules. Journal of the ACM 57(5), 93–154 (2010)
16. Parsons, S., Sierra, C., Jennings, N.R.: Agents that reason and negotiate by arguing. Journal of Logic and Computation 8(3), 261–292 (1998)
17. Sabater, J., Sierra, C., Parsons, S., Jennings, N.R.: Engineering executable agents using multi-context systems. Journal of Logic and Computation 12(3), 413–442 (2002)