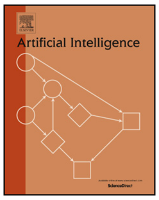
Artificial Intelligence 216 (2014) 233–274

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint



Finding explanations of inconsistency in multi-context systems [☆]

Thomas Eiter[a], Michael Fink[a,*], Peter Schüller[b], Antonius Weinzierl[a]

[a] Institute of Information Systems, Vienna University of Technology, Vienna, Austria
[b] Computer Engineering Department, Faculty of Engineering, Marmara University, Istanbul, Turkey

ARTICLE INFO

Article history:
Received 24 December 2012
Received in revised form 19 July 2014
Accepted 19 July 2014
Available online 30 July 2014

Keywords:
Multi-context systems
Inconsistency management
Interlinked knowledge
Knowledge representation formalisms
Nonmonotonic reasoning
Computational logic

ABSTRACT

Interlinking knowledge sources to enable information exchange is basic means to build enriched knowledge-based systems, which gains importance with the spread of the Internet. Inconsistency, however, arises easily in such systems, which is not least due to their heterogeneity, but also due to their independent design. This makes developing methods for consistency management of such systems a pressing issue. An important aspect is that in many relevant cases, the information at individual sources may not be amenable to change in order to resolve inconsistency, like in case of autonomous management of the sources. We thus aim at analyzing inconsistency of a system by means of the interlinking of sources and changes thereof. More concretely, we consider the powerful framework of Multi-Context Systems, in which decentralized and heterogeneous system parts interact via (possibly nonmonotonic) bridge rules for information exchange. Nonmonotonicity and potential cyclic dependencies pose additional challenges that call for suitable methods of inconsistency analysis. We thus provide two approaches for explaining inconsistency, which both characterize inconsistency in terms of bridge rules, but in different ways: by pointing out rules which need to be altered for restoring consistency, and by finding combinations of rules which cause inconsistency. We show duality and modularity properties of these notions, give precise complexity characterizations, and provide algorithms for their computation, which have been implemented in a prototype, by means of so-called HEX-programs. Our results provide a basis for inconsistency management in heterogeneous knowledge systems which, different from and orthogonal to other works, explicitly addresses the knowledge interlinks in order to restore consistency.

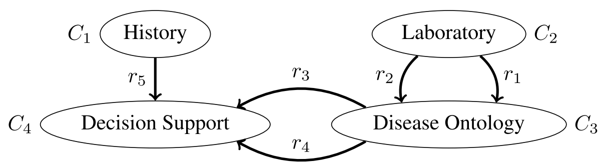
1. Introduction

In recent years, there has been increasing interest in interlinking information, driven by—and reflected in—the development of the World Wide Web. Increasing demands not only concern the large number of available sources and the degree of interlinking, but also the quality of the information. From its initial conception as a means to link textual data, the Internet has evolved to a medium for interlinking, accessing, and exchanging more structured information including rela-

☆ This article is a revised and significantly extended version of the preliminary paper [37] presented at the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR 2010), Toronto, Canada, May 9–13, 2010.

* Corresponding author.
E-mail addresses: eiter@kr.tuwien.ac.at (T. Eiter), fink@kr.tuwien.ac.at (M. Fink), peter.schuller@marmara.edu.tr (P. Schüller), weinzierl@kr.tuwien.ac.at (A. Weinzierl).

http://dx.doi.org/10.1016/j.artint.2014.07.008
0004-3702/© 2014 Elsevier B.V. All rights reserved.



r_1: $(3 : d{:}Pneumonia) \leftarrow (2 : xray\_pneumonia)$.
r_2: $(3 : (d, m1){:}has\_marker) \leftarrow (2 : blood\_marker)$.
r_3: $(4 : need\_ab.) \leftarrow (3 : d{:}BacterialDisease)$.
r_4: $(4 : need\_strong.) \leftarrow (3 : d{:}AtypPneumonia)$.
r_5: $(4 : allow\_strong\_ab.) \leftarrow \textbf{not}\ (1 : allergy\_strong\_ab)$.
$kb_1 = \{allergy\_strong\_ab\}$
$kb_2 = \{blood\_marker, xray\_pneumonia\}$
$kb_3 = \{Pneumonia \sqsubseteq BacterialDisease,$
$Pneumonia \sqcap \exists has\_marker.\top \sqsubseteq AtypPneumonia\}$
$kb_4 = \{give\_strong \vee give\_weak \leftarrow need\_ab.$
$give\_strong \leftarrow need\_strong.$
$\perp \leftarrow give\_strong, not\ allow\_strong\_ab.$
$give\_nothing \leftarrow not\ need\_ab, not\ need\_strong.\}$

Minimal diagnoses and explanations as defined in Section 3.
$D^{\pm}_m(M) = \{(\{r_1\}, \emptyset), (\{r_2\}, \emptyset), (\{r_4\}, \emptyset), (\emptyset, \{r_5\})\}$
$E^{\pm}_m(M) = (\{r_1, r_2, r_4\}, \{r_5\})$

Fig. 1. Example MCS with contexts C_i, bridge rules br_i, and knowledge bases kb_i.

tional and semi-structured data, and in the last years also semantically richer knowledge sources. Systems can be built in which individual information sources are connected, such that more informed and accurate answers can be given to specific user problems. Typically, these sources are expressed in different formalisms, and they are autonomously managed by third parties, such that a real integration is difficult.

Developing uniform, high-level formalisms to capture such systems has thus become a relevant issue in knowledge representation and reasoning (KRR). The Multi-Context System (MCS) framework [14,21,24,53,85,87], which evolved from MultiLanguage systems [54,55], is an expressive framework for this purpose. It is a powerful knowledge representation formalism for many application scenarios where heterogeneity and inter-contextual information exchange are essential. Multi-context systems as introduced in [21] consist of knowledge bases (in possibly heterogeneous and/or nonmonotonic logics) at nodes (called *contexts*) that formulate the exchange of information via so called bridge rules, such as

$$\begin{aligned}\big(c_1 : ok(X\_Risk)\big) \leftarrow{} & \big(c_2 : insurance(X\_Risk)\big),\\ & \big(c_2 : low\_rate(X\_Risk)\big), \textbf{not}\ \big(c_3 : black\_list(X\_Risk)\big);\end{aligned}$$

informally, this rule states that if X_Risk is a low rate insurance company according to knowledge source (context) c_2, and it is not known to be blacklisted according to context c_3, then context c_1 adds the fact $ok(X\_Risk)$ to its knowledge base.

MCS enable knowledge exchange at a general level, by interlinking possibly heterogeneous formalisms such as ontologies, databases, and logic programs. However, due to their decentralized nature, information exchange can have unforeseen effects, and in particular cause an MCS to be inconsistent. For example, consider a system for supporting health care decisions in a hospital, that comprises several components: (i) a database of laboratory test results; (ii) a patient record database; (iii) an ontology for disease classification; and (iv) a decision support system suggesting suitable treatments for patients. Modeled as an MCS, each component is a context and the information flow between them is specified by suitable bridge rules (see Fig. 1 for an overview and further details which will become relevant later on). Thanks to bridge rules existing systems might be easily incorporated. Suppose that the decision support system concludes that a patient must be given a special drug, but the patient database states that she is allergic to that drug, thus counter-indicating its use. The whole system gets inconsistent (which renders the system useless) unless such special cases were anticipated when contexts and bridge rules were modeled.

In real world applications, system complexity tends to increase, in terms of both contexts and interconnectivity. Extensive testing that considers all possible states of a system is often infeasible, especially if legacy information systems are employed as contexts in an MCS. Therefore, and specifically due to the heterogeneity of individual system components that are linked together, inconsistency arises easily and methods for handling inconsistency are an important issue. Our work aims at *analyzing* inconsistencies in MCS, in order to understand where and why such inconsistencies occur, and how they might be removed. It thus provides a basis for the specification of concrete strategies to handle inconsistencies and to extend systems with inconsistency management mechanisms (in addition to some basic operations that can be obtained directly from our approach).

Various approaches to cope with inconsistent information have been developed for different KRR formalisms (see Section 7 for works closely related to ours and Appendix B for a more general overview). Specifically in traditional data integration scenarios, inconsistency problems also surface naturally and methods for consistency restoring or maintenance have been studied extensively. Compared to that, however, our work proposes novel concepts that *differ beyond settings and* *formalisms* considered in prior works: the important point is that we *focus on the exchange of information, its interlinking*, i.e., on *adjusting bridge rules* instead of modifying the data in the contexts. In data integration terms, we thus consider modifications of the mapping as a potential for resolving inconsistency rather than repairing or cleaning the data. While the importance of maintaining and repairing mappings has been recognized in database integration [31], progress on this has been on a slow pace.

The motivation for addressing inconsistency in MCS on the level of bridge rules stems from the fact that an MCS usually models a loose integration scenario with autonomous sources (e.g., if companies link their business logics), where changing contexts or their data to restore consistency may not be an option. Compared to data integration settings that globally materialize (at least virtually) the data of the individual sources, it may not be possible to modify (or even access all the) data that is internal to one of the knowledge bases employed in an MCS. This applies in particular to cases of cyclic information exchange. Therefore, we resort to bridge rules as the source of inconsistency, and their modification as a possibility of counteracting. On the one hand, under the reasonable assumption that each context is consistent if bridge rules are disregarded, we can fully capture reasons of inconsistency in terms of bridge rules. On the other hand, negation and potential cyclic dependencies (as opposed to acyclic mappings or cyclic mappings of limited query depth in data integration [8]) render the task of characterizing and analyzing inconsistency non-trivial.

Starting from this, our contributions are summarized as follows:

(1) In the spirit of debugging approaches used in the nonmonotonic reasoning community, especially in logic programming [20,62,72,80,90], we introduce two notions of explaining inconsistency in MCS: a *consistency-based* notion, called *diagnosis*, and an *entailment-based* notion, called *explanation*. Intuitively, a diagnosis characterizes inconsistency in terms of modified sets of bridge rules that admit equilibria (i.e., consistency of the system), while explanations point out subsets of bridge rules that are sufficient for inconsistency in the system. Potential nonmonotonicity makes these notions challenging, because both removing and adding bridge rules can cause as well as prevent inconsistency in an MCS. Despite that, our notions have appealing properties. We investigate refinements and restrictions of our notions; however, we also show that these can be expressed in terms of our basic notions.

(2) We establish useful properties of our notions. First, conversion and duality results between diagnoses and explanations show that, while representing different analytic properties, they identify the same overall set of bridge rules as relevant for inconsistency. This in fact generalizes a similar classic result by Reiter [84], who characterized the consistency-based diagnoses of system description in classical (monotonic) logic in terms of conflict sets. Furthermore, we establish modularity properties in the spirit of Splitting Sets [71], which allow for an incremental computation of diagnoses and explanations, taking the MCS topology into account.

(3) We exactly characterize the computational complexity of identifying explanations, under varying assumptions on the complexity of reasoning in contexts (note that by the underlying assumption, consistency-based explanations always exist). It turns out that this problem has for a range of context complexities no (or only mildly) higher complexity than the contexts themselves. As a consequence, computing explanations is in some cases not harder than consistency checking.

(4) Finally, we consider how consistency- and entailment-based explanations in the sense above can be computed, i.e., modified sets of bridge rules that admit equilibria respectively are sufficient for inconsistency of the system. To this end, we resort here to HEX-programs, which are a generalization of Answer Set Programming (ASP) by so-called external atoms that provide access to external sources of computation. An experimental prototype has been implemented, while more advanced implementations are underway.

Our results provide a basis for building enhanced MCS systems which are capable of analyzing and reasoning about emerging inconsistencies. Rather than automatically resolving inconsistency, as e.g. in [13], we envisage a (semi-)automatic approach with user support for locating and tracking parts that cause inconsistency. Indeed, user invention may be needed as often no automatic solution is feasible. For instance in our healthcare example, giving the special drug would resolve the inconsistency, but this should only be done after approval by a medical doctor.

Structure. The remainder of this article is organized as follows. Section 2 provides necessary preliminaries on the MCS framework and introduces a running example. In Section 3, diagnoses and explanations are introduced, while Section 4 contains conversion results and modularity properties that depend on the interlinking of MCS. Section 5 contains a detailed analysis of the computational complexity of identifying both diagnoses and explanations, and in Section 6 we elaborate on how they can be computed using HEX programs. A comprehensive discussion of related work is given in Section 7, followed by concluding remarks and directions for future work in Section 8. We give more details to some of our examples in Appendix A, discuss a broad range of related work in Appendix B, and provide proofs for all results in Appendix C.

2. Preliminaries

In this section, we recall nonmonotonic MCS from [21]. Further background is given in [22,32], which discuss extensions of MCS, compare them to other related formalisms, and survey computational issues.

Loosely speaking, a nonmonotonic MCS consists of *contexts*, each composed of a knowledge base with an underlying abstract *logic*, and a set of *bridge rules* which control the information flow between contexts. For a running example we use an MCS shown in Fig. 1 whose details are fully explained in Example 2 and in the remainder of this section. The MCS framework uses a minimalistic, abstract model of logics, which consists of possible sets of formulas, possible sets of beliefs, and a satisfiability relation.

Definition 1. A logic $L = (\mathbf{KB}_L, \mathbf{BS}_L, \mathbf{ACC}_L)$ consists, in an abstract view, of the following components:

- $\mathbf{KB}_L$ is the set of well-formed knowledge bases of L. We assume each element of $\mathbf{KB}_L$ is a set (of "formulas").
- $\mathbf{BS}_L$ is the set of possible belief sets, where the elements of a belief set are statements that possibly hold, given a knowledge base.
- $\mathbf{ACC}_L : \mathbf{KB}_L \to 2^{\mathbf{BS}_L}$ is a function describing the "semantics" of the logic by assigning to each knowledge base a set of acceptable belief sets.

Intuitively, a belief set is a set of statements (beliefs) that a reasoner may jointly hold, and $\mathbf{ACC}_i(kb)$ singles out, given a knowledge base kb, the belief sets that are justifiably acceptable. To accommodate nonmonotonic formalisms where knowledge base may have multiple such acceptable belief sets (e.g., answer sets of logic program; extensions of a default theory; expansions of an autoepistemic theory), $\mathbf{ACC}_i(kb)$ is designed as a multi-valued function. Note that the possible belief sets in $\mathbf{BS}_L$ are abstract without imposing any conditions a priori. This allows to flexibly capture a host of concrete applications and arbitrary context reasoning capabilities viewing belief sets as an abstract interface. E.g., for classical logic, belief sets might contain only atoms, only literals, or formulas of certain structure or nesting depth; they may be logically closed or not, depending on the particular capability of belief set formation.

This abstract notion of a purely functional "logic" captures many monotonic and nonmonotonic logics, e.g., classical logic, description logics, modal, default, and autoepistemic logics, circumscription, and logic programs under answer set semantics. Moreover, it allows also to consider formalisms not based on logic (e.g., Dung-style argumentation systems) under a suitable representation.

Example 1. We illustrate how this abstraction of a logic captures some well-known knowledge-representation formalisms. The introduced logics are also used in formalizing our running example.

Classical propositional logic. To capture classical (propositional) logic over a set At of propositional atoms, we may define:

- $\mathbf{KB}^c = 2^{\Sigma}$ is the set of all subsets of Σ, where Σ is the set of well-formed formulas over At built using the connectives $\wedge, \vee, \neg, \to$;
- $\mathbf{BS}^c = 2^{\Sigma}$, i.e., each set of formulas is a possible belief set; and
- $\mathbf{ACC}^c$ returns for each set $kb \in \mathbf{KB}^c$ of well-formed formulas a singleton set that contains the set of formulas entailed by kb; if $\models_c$ denotes classical entailment, then $\mathbf{ACC}^c(kb) = \{\{F \in \Sigma \mid kb \models_c F\}\}$.

The resulting logic $L^c_{\Sigma} = (\mathbf{KB}^c, \mathbf{BS}^c, \mathbf{ACC}^c)$ captures entailment in classical logics.

In practice, the formulas in knowledge bases and belief sets might be restricted to particular forms, e.g., to literals; we denote the respective logic by $L^{pl}_{\Sigma} = (\mathbf{KB}^{pl}, \mathbf{BS}^{pl}, \mathbf{ACC}^{pl})$. Note that

$$\mathbf{ACC}^{pl}(kb) = \big\{\{A \in At \mid kb \models_c A\} \cup \{\neg A \mid A \in At \text{ and } kb \models_c \neg A\}\big\}.$$

For our running example we employ two contexts, C_1 and C_2, using signatures $\Sigma_1 = \{allergy\_strong\_ab\}$ and $\Sigma_2 = \{blood\_marker, xray\_pneumonia\}$, and logics $L^{pl}_{\Sigma_1}$ and $L^{pl}_{\Sigma_2}$, respectively. Their knowledge bases are as follows:

$$kb_1 = \{allergy\_strong\_ab\},$$

$$kb_2 = \{blood\_marker, xray\_pneumonia\}.$$

Those knowledge bases provide information that the patient is allergic to strong antibiotics (kb_1), respectively that a certain blood marker is present and that pneumonia was detected in an X-ray examination (kb_2).

The corresponding semantics is given by $\mathbf{ACC}(kb_1) = \{\{allergy\_strong\_ab\}\}$ for C_1, and $\mathbf{ACC}(kb_2) = \{\{blood\_marker, xray\_pneumonia\}\}$ for C_2.

Description logic. For ontologies with syntax and semantics of the description logic $\mathcal{ALC}$ (see [2]), we use the abstract logic $L_{\mathcal{A}}$ (for details see Example 13 in Appendix A). An $L_{\mathcal{A}}$-knowledge base contains both A-Box and T-Box axioms. An accepted belief set of such a knowledge base is the set of atomic assertions that follow from the knowledge base.

For a running example, we use an ontology about diseases, given by context C_3 using $L_{\mathcal{A}}$. Its knowledge base, kb_3, consists of two axioms, where the first states that pneumonia is a bacterial disease and the second states that pneumonia which has an associated blood-marker implies an atypical pneumonia (that is a severe form of pneumonia). The corresponding knowledge base is:

$$\begin{aligned} kb_3 = \{&Pneumonia \sqsubseteq BacterialDisease, \\ &Pneumonia \sqcap \exists has\_marker.\top \sqsubseteq AtypPneumonia\}. \end{aligned}$$

As kb_3 is satisfiable and contains only terminological knowledge, no assertions follow from this knowledge base, thus $\mathbf{ACC}(kb_3) = \{\emptyset\}$. Adding the assertions that d is pneumonia and that the role has_marker holds between d and $m1$ results in the conclusion that d is also a bacterial disease and atypical pneumonia, i.e.,

$$\mathbf{ACC}\big(kb_3 \cup \{d{:}Pneumonia, (d, m1){:}has\_marker\}\big)$$
$$= \big\{\{d{:}Pneumonia,\ d{:}BacterialDisease, d{:}AtypPneumonia, (d, m1){:}has\_marker\}\big\}.$$

Disjunctive answer set programming. For normal disjunctive logic programs under answer set semantics over a non-ground signature Σ (cf. [45] and [83]), we use the abstract logic L^{asp}_{Σ}, which is detailed in the appendix in Example 14. We employ L^{asp}_{Σ} for a context, C_4, suggesting proper treatments where $\Sigma = \{give\_strong, give\_weak, need\_ab, allow\_strong\_ab, give\_nothing\}$. The knowledge base for C_4 is:

$$\begin{aligned} kb_4 = \{&give\_strong \lor give\_weak \leftarrow need\_ab.\\ &give\_strong \leftarrow need\_strong.\\ &\bot \leftarrow give\_strong, not\ allow\_strong\_ab.\\ &give\_nothing \leftarrow not\ need\_ab, not\ need\_strong.\}.\end{aligned}$$

C_4 suggests a treatment which is either a strong antibiotics, a weak antibiotics, or no medication at all. Without further information, kb_4 thus concludes that nothing is required, i.e., $\mathbf{ACC}(kb_4) = \{\{give\_nothing\}\}$. If $need\_ab$ is added, however, kb_4 results in two answer sets, i.e., $\mathbf{ACC}(kb_4 \cup \{need\_ab.\}) = \{A_1, A_2\}$ where $A_1 = \{give\_strong, need\_ab\}$ and $A_2 = \{give\_weak, need\_ab\}$. □

A *bridge rule* can add information to a context, depending on the belief sets which are accepted at other contexts. Let $L = (L_1, \ldots, L_n)$ be a sequence of logics. An L^k-bridge rule r over L is of the form

$$(k : s) \leftarrow (c_1 : p_1), \ldots, (c_j : p_j), \mathbf{not}\ (c_{j+1} : p_{j+1}), \ldots, \mathbf{not}\ (c_m : p_m).,\tag{1}$$

where for each $i \in \{1, \ldots, m\}$ we have $1 \le c_i \le n$ (using integers as context identifiers), p_i is an element of some belief set of L_{c_i}, and s is a knowledge-base formula of L_k (i.e. $s \in \bigcup \mathbf{KB}_{L_k}$).

We denote by $\varphi(r)$ the formula s in the head of r and by $C_h(r)$ the context k where r belongs to. The full head of r is denoted by $head(r) = (k : s)$, thus $head(r) = (C_h(r) : \varphi(r))$. The literals in the body of r are referred to by $body(r)$, $body^+(r)$, $body^-(r)$, and $Body(r)$ which respectively denote the sets $\{(c_1 : p_1), \ldots, (c_m : p_m)\}$, $\{(c_1 : p_1), \ldots, (c_j : p_j)\}$, $\{(c_{j+1} : p_{j+1}), \ldots, (c_m : p_m)\}$, and $\{(c_1 : p_1), \ldots, (c_j : p_j), \mathbf{not}\ (c_{j+1} : p_{j+1}), \ldots, \mathbf{not}\ (c_m : p_m)\}$. Furthermore, $C_b(r)$ denotes the set of contexts referenced in r's body, i.e., $C_b(r) = \{c_i \mid (c_i : p_i) \in body(r)\}$. For technical use later, we denote by $cf(r)$ the *condition-free* bridge rule stemming from r by removing all elements in its body, i.e., $cf(r)$ is $(k : s) \leftarrow .$, and for any set of bridge rules R, we let $cf(R) = \bigcup_{r \in R} cf(r)$.

Definition 2. A multi-context system $M = (C_1, \ldots, C_n)$ is a collection of contexts $C_i = (L_i, kb_i, br_i)$, $1 \le i \le n$, where $L_i = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$ is a logic as above, $kb_i \in \mathbf{KB}_i$ a knowledge base, and br_i is a set of L^k-bridge rules over $L = (L_1, \ldots, L_n)$. Furthermore, for each $H \subseteq \{\varphi(r) \mid r \in br_i\}$ it holds that $kb_i \cup H \in \mathbf{KB}_{L_i}$, i.e., adding bridge rule heads to a knowledge base again yields a knowledge base.

By $br(M) = \bigcup_{i=1}^{n} br_i$ and $C(M) = \{C_1, \ldots, C_n\}$ we denote the set of all bridge rules, resp. the set of all contexts of M. We write $br_i(M)$ to denote the set of bridge rules of context i of M, i.e., $br_i(M) = \{r \in br(M) \mid C_h(r) = i\}$. In the following, we formally introduce our running example.

Example 2. Consider an MCS M embodying a health care decision support system that contains the following contexts: a patient history database (C_1), a blood and X-Ray analysis database (C_2), a disease ontology (C_3), and a decision support system (C_4) which suggests proper treatments. The corresponding abstract logics and knowledge bases are those in Example 1. The knowledge bases and bridge rules of M are shown in Fig. 1, where each bridge rule $r \in \{r_1, \ldots, r_5\}$ with $C_h(r) = j$ and $i \in C_b(r)$ is depicted as an arrow from C_i to C_j.

Rules r_1 and r_2 (belonging to context C_3) provide input for disease classification to the ontology; they assert facts about a new individual 'd' corresponding to the patient. Rules r_3 and r_4 (belonging to context C_4) link disease information with medication requirements, while r_5 (also belonging to context C_4) relates acceptance of strong antibiotics with an allergy check on the patient database. □

The semantics of MCS is defined over locally accepted belief sets as follows. A *belief state* of an MCS $M = (C_1, \ldots, C_n)$ is a sequence $S = (S_1, \ldots, S_n)$ of belief sets $S_i \in \mathbf{BS}_i$, $1 \le i \le n$. A bridge rule r of form (1) is *applicable* in S, denoted $S \models r$, iff for all $(j : p) \in body^+(r)$ it holds that $p \in S_j$, and for all $(j : p) \in body^-(r)$ it holds that $p \notin S_j$. For a set R of bridge rules, $app(R, S) = \{r \in R \mid S \models r\}$, denotes the set of applicable bridge rules.

The equilibrium semantics defines acceptable belief states of an MCS. Intuitively, it selects a belief state S of an MCS M as acceptable, if each context C_i takes the heads of all bridge rules that are applicable in S into account, and accepts its belief set S_i.

Definition 3. A belief state $S = (S_1, \dots, S_n)$ of M is an equilibrium iff for every belief set S_i, $1 \le i \le n$, it holds that $S_i \in \mathbf{ACC}_i(kb_i \cup \{\varphi(r) \mid r \in app(br_i, S)\})$.

Example 3. Imagine that we make a small modification to our running example MCS M (Fig. 1), such that

$$kb_2 = \{\neg blood\_marker, xray\_pneumonia\},$$

i.e., imagine that the blood marker is not present. Then M has a single equilibrium $S = (S_1, S_2, S_3, S_4)$ where

$$S_1 = \{allergy\_strong\_ab\},$$

$$S_2 = \{\neg blood\_marker, xray\_pneumonia\},$$

$$S_3 = \{d{:}Pneumonia, d{:}BacterialDisease\},$$

$$S_4 = \{need\_ab, give\_weak\}.$$

The only rules applicable in S are r_1 and r_3, as $app(br_1(M), S) = app(br_2(M), S) = \emptyset$, $app(br_3(M), S) = \{r_1\}$, and $app(br_4(M), S) = \{r_3\}$. Note that if we replace S_4 with $\{need\_ab, give\_strong, allow\_strong\_ab\}$, then the resulting belief state is not an equilibrium: C_4 uses answer set semantics, therefore *allow_strong_ab* cannot be part of S_4 unless it is added by a bridge rule. The only bridge rule with this head is r_5, and its applicability is blocked by the presence of *allergy_strong_ab* in kb_1 and in S_1. □

3. Diagnoses and explanations for inconsistency

Inconsistency in an MCS is the lack of an equilibrium. As the combination and interaction of heterogeneous, possibly autonomous, systems can easily have unforeseen and intricate effects, inconsistency is a major problem in MCS. To provide support for restoring consistency, we seek to understand and give reasons for inconsistency.

Example 4. The MCS in our running example (Fig. 1) is inconsistent since, unlike in Example 3, r_2 and r_4 are applicable, which in turn requires strong antibiotics. This is in conflict with the patient's allergy. Note that applicability of r_5 would resolve this inconsistency by activating *allow_strong_ab*. However, presence of the fact *allergy_strong_ab* in kb_1 together with body atom '**not** $(1 : allergy\_strong\_ab)$' in r_5 prevents the applicability of r_5 (due to negation as failure). □

We will use the following notation. Given an MCS M and a set R of bridge rules (that are compatible with M), we denote by $M[R]$ the MCS obtained from M by replacing its set of bridge rules $br(M)$ with R; e.g., $M[br(M)] = M$ and $M[\emptyset]$ is M with no bridge rules. By $M \models \bot$ we denote that M has no equilibrium, i.e., is inconsistent, and by $M \not\models \bot$ the opposite.

In the following, we consider two possibilities for explaining inconsistency in MCS: first, a consistency-based formulation, which identifies a subset of the bridge rules which need to be changed to restore consistency. Second, an entailment-based formulation, which identifies a subset of the bridge rules which is required to make the MCS inconsistent. Following common terminology, we call the first formulation *diagnosis* (cf. [84]) and the second *inconsistency explanation*.

3.1. Diagnoses

As well-known, adding knowledge in nonmonotonic reasoning can both cause and prevent inconsistency; the same is true for removing knowledge.

For our consistency-based explanation of inconsistency, we therefore consider pairs of sets of bridge rules, such that if we deactivate the rules in the first set, and add the rules in the second set in unconditional form, the MCS becomes consistent (i.e., admits an equilibrium). Adding rules unconditionally is the most severe form of modification of a rule's body, but as we later see, this notion also allows to capture more fine-grained forms of modification.

Definition 4. Given an MCS M, a *diagnosis* of M is a pair (D_1, D_2), $D_1, D_2 \subseteq br(M)$, such that $M[br(M) \setminus D_1 \cup cf(D_2)] \not\models \bot$. By notation, $D^{\pm}(M)$ is the set of all diagnoses.

To obtain a more relevant set of diagnoses, by Occam's razor we prefer subset-minimal diagnoses, where for pairs $A = (A_1, A_2)$ and $B = (B_1, B_2)$ of sets, the pointwise subset relation $A \subseteq B$ holds iff $A_1 \subseteq B_1$ and $A_2 \subseteq B_2$.

Definition 5. Given an MCS M, $D^{\pm}_m(M)$ is the set of all pointwise subset-minimal diagnoses of an MCS M, i.e.,

$$D^{\pm}_m(M) = \left\{D \in D^{\pm}(M) \,\middle|\, \forall D' \in D^{\pm}(M) : D' \subseteq D \Rightarrow D \subseteq D'\right\}.$$

Example 5. In our running example, we obtain

$$D_m^{\pm}(M) = \{(\{r_1\}, \emptyset), (\{r_2\}, \emptyset), (\{r_4\}, \emptyset), (\emptyset, \{r_5\})\}.$$

Accordingly, deactivating r_1, or r_2, or r_4, or adding r_5 unconditionally, will result in a consistent MCS.

In more detail, we find: diagnosis $(\{r_1\}, \emptyset)$ removes bridge rule r_1. This way we ignore the X-Ray finding and obtain the following equilibrium:

$$EQ_1 = (\{\mathit{allergy\_strong\_ab}\}, \{\mathit{blood\_marker}, \mathit{xray\_pneumonia}\},$$
$$\{(d, m1){:}\mathit{has\_marker}\}, \{\mathit{give\_nothing}\}).$$

It represents that we do not treat the patient since no illness is detected in it.

Diagnosis $(\{r_4\}, \emptyset)$ removes bridge rule r_4. This ignores the information that treating the illness requires the strong antibiotics. We obtain the following equilibrium:

$$EQ_2 = (\{\mathit{allergy\_strong\_ab}\}, \{\mathit{blood\_marker}, \mathit{xray\_pneumonia}\},$$
$$\{(d, m1){:}\mathit{has\_marker}, d{:}\mathit{Pneumonia}, d{:}\mathit{BacterialDisease}, d{:}\mathit{AtypPneumonia}\},$$
$$\{\mathit{need\_ab}, \mathit{give\_weak}\}).$$

Diagnosis $(\emptyset, \{r_5\})$ adds an unconditional copy of bridge rule r_5, which forces strong antibiotics to be allowed as a treatment. The modified system has the following equilibrium:

$$EQ_3 = (\{\mathit{allergy\_strong\_ab}\}, \{\mathit{blood\_marker}, \mathit{xray\_pneumonia}\},$$
$$\{(d, m1){:}\mathit{has\_marker}, d{:}\mathit{Pneumonia}, d{:}\mathit{BacterialDisease}, d{:}\mathit{AtypPneumonia}\},$$
$$\{\mathit{need\_ab}, \mathit{need\_strong}, \mathit{allow\_strong\_ab}, \mathit{give\_strong}\}).$$

Any or none of the above possibilities might be the right choice: such decisions ought to be taken by a domain specialist (e.g., a doctor) and cannot be done automatically. Therefore analysis of inconsistency is important to identify reasons for it. □

Preference on diagnoses can be defined in general, relying on some notion of plausibility (see e.g., for abduction [25]). This is, however, beyond the scope of this work, as we investigate basic notions of inconsistency here.

3.2. Explanations

In the spirit of abductive reasoning, we also propose an entailment-based notion of explaining inconsistency. An *inconsistency explanation* (in short, an *explanation*) is a pair of sets of bridge rules, whose presence or, expected absence entails a relevant inconsistency in the given MCS.

Definition 6. Given an MCS M, an *inconsistency explanation* of M is a pair (E_1, E_2) of sets $E_1, E_2 \subseteq br(M)$ of bridge rules, such that for all (R_1, R_2) where $E_1 \subseteq R_1 \subseteq br(M)$ and $R_2 \subseteq br(M) \setminus E_2$, it holds that $M[R_1 \cup cf(R_2)] \models \bot$. By $E^{\pm}(M)$ we denote the set of all inconsistency explanations of M, and by $E_m^{\pm}(M)$ the set of all pointwise subset-minimal ones.

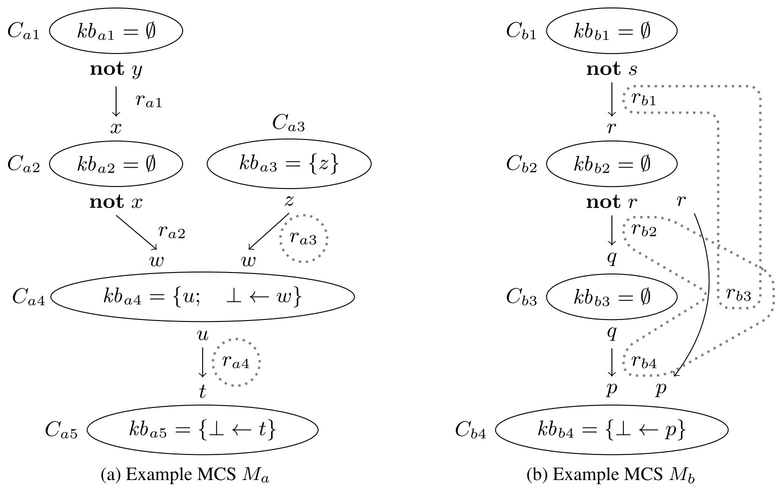
The intuition about E_1 is as follows: bridge rules in E_1 are crucial to create an inconsistency in M (i.e., $M[E_1] \models \bot$), moreover this inconsistency is *relevant* for M in the sense that adding some bridge rules from $br(M)$ to $M[E_1]$ never yields a consistent system, i.e., $M[E_1 \cup R] \models \bot$ for all $R \subseteq br(M)$.

This condition of relevancy is necessary for non-monotonic reasoning systems; for example the program $P = \{a \leftarrow not\, a.\}$ is inconsistent under the answer set semantics, but its superset $P' = \{a \leftarrow not\, a.\ a.\}$ is consistent. The inconsistency of P does not matter for P'. In terms of MCS, a set of bridge rules may create an inconsistency that is irrelevant if the system is consistent when more or all bridge rules are present.

The intuition about E_2 regards inconsistency w.r.t. the application of bridge rules: $M[E_1]$ cannot be made consistent unless at least one bridge rule from E_2 fires, i.e., $M[E_1 \cup cf\, R] \models \bot$ if $R \subseteq br(M) \setminus E_2$.

In summary, bridge rules E_1 create a relevant inconsistency, and at least one bridge rule in E_2 must be applied in unconditional form to repair that inconsistency.

Example 6. In our running example (Fig. 1), we have one minimal inconsistency explanation, namely $(\{r_1, r_2, r_4\}, \{r_5\})$. To trigger the only possible inconsistency, which is in C_4, we need to import *need_strong* (using r_4) and we must not import *allow_strong_ab* (using r_5). Furthermore, r_4 can only fire if C_3 accepts the belief *d:AtypPneumonia*, which is only possible if r_1 and r_2 fire. Therefore, r_1, r_2, and r_4 *must be present* to get inconsistency, and the head of r_5 *must not be present*. □



C_{a1} $kb_{a1} = \emptyset$
not y
r_{a1}
x
C_{a2} $kb_{a2} = \emptyset$
C_{a3} $kb_{a3} = \{z\}$
not x
z
r_{a2}
r_{a3}
w
w
C_{a4} $kb_{a4} = \{u;\quad \bot \leftarrow w\}$
u
r_{a4}
t
C_{a5} $kb_{a5} = \{\bot \leftarrow t\}$
(a) Example MCS M_a
C_{b1} $kb_{b1} = \emptyset$
not s
r_{b1}
r
C_{b2} $kb_{b2} = \emptyset$
not r
r
r_{b2}
q
C_{b3} $kb_{b3} = \emptyset$
r_{b3}
q
r_{b4}
p
p
C_{b4} $kb_{b4} = \{\bot \leftarrow p\}$
(b) Example MCS M_b

Fig. 2. Example MCS for illustrating properties and the usefulness of inconsistency explanations. Dotted areas indicate individual inconsistency explanations.

From Definition 6 the following property follows immediately.

Proposition 1. *Given an explanation E of an MCS M, every E' such that $E \subseteq E' \subseteq (br(M), br(M))$ is also an explanation.*

The following example shows how explanations separate reasons for inconsistency.

Example 7. Consider $M_a = (C_{a1}, C_{a2}, C_{a3}, C_{a4}, C_{a5})$ depicted in Fig. 2a: all contexts use logic L^{asp}_{Σ} from Example 1 with $\Sigma = \{a, b, \ldots, z\}$. This system is inconsistent, because u is a fact in C_{a4} and therefore r_{a4} adds fact t to C_{a5} which makes C_{a5} inconsistent. An alternative source of inconsistency is that z is a fact in C_{a3}, therefore r_{a3} adds fact w to C_{a4} which makes C_{a4} inconsistent. In both cases, consistency cannot be restored by adding rules or making rules unconditional. $E^{\pm}_{m}(M)$ contains only the explanations $(\{r_{a3}\}, \emptyset)$ and $(\{r_{a4}\}, \emptyset)$, which each capture one of these sources of inconsistency. Note that $M_a[\{r_{a2}\}]$ also is an inconsistent system, because r_{a2} adds the fact w to C_{a4}, which makes C_{a4} inconsistent. But, since $M_a[\{r_{a1}, r_{a2}\}]$ is consistent, this inconsistency is not relevant and therefore not reported by our notions.

We next consider the MCS $M_b = (C_{b1}, C_{b2}, C_{b3}, C_{b4})$ depicted in Fig. 2b to show that mutually exclusive bridge rules can be part of the same explanation, and some advantage of subset- over cardinality-minimality. Again, all contexts use logic L^{asp}_{Σ} from Example 1 with $\Sigma = \{a, b, \ldots, z\}$. M_b is inconsistent, as p causes inconsistency in C_{b4} and p is added to the knowledge base of C_{b4} by bridge rule r_{b3}. Due to bridge rule r_{b1}, r is always believed by C_{b2}, hence r_{b3} is always applicable. This inconsistency cannot be prevented by bridge rules of M_b, or unconditional versions thereof. Therefore $(\{r_{b1}, r_{b3}\}, \emptyset)$ is a minimal explanation of M_b. Another minimal explanation is $(\{r_{b2}, r_{b3}, r_{b4}\}, \emptyset)$, where the bodies of r_{b2} and r_{b3} are mutually exclusive. However, only together they ensure that C_{b4} is inconsistent, regardless of whether r_{b1} is present and whether fact r is believed at C_{b2}. □

The above example also shows that cardinality-minimal explanations cannot identify all sources of inconsistency, since there are two $\subseteq$-minimal explanations, but only one cardinality-minimal one. Additionally, the set of cardinality-minimal explanations does not point out all bridge rules that must be modified to obtain a consistent system.

3.3. Deletion-diagnoses/deletion-explanations

For domains where removal of bridge rules is preferred to making rules unconditional, we specialize $D^{\pm}$ to obtain diagnoses of the form $(D_1, \emptyset)$. By Occam's razor, subset-minimal diagnoses are preferred.

Definition 7. Given an MCS M, a deletion-diagnosis of M is a set $D \subseteq br(M)$ such that $M[br(M) \setminus D] \not\models \bot$. The set of all deletion-diagnoses (resp., $\subseteq$-minimal deletion-diagnoses) is $D^{-}(M)$ (resp., $D^{-}_{m}(M)$).

Specializing inconsistency explanations to the first component, i.e., disregarding that rules may be made unconditional, all explanations are of the form $(E_1, br(M))$.

Definition 8. Given an MCS M, a deletion-explanation of M is a set $E \subseteq br(M)$ such that each R, where $E \subseteq R \subseteq br(M)$, satisfies $M[R] \models \bot$. The set of all such ($\subseteq$-minimal) explanations is denoted by $E^{+}(M)$, and the set of $\subseteq$-minimal ones by $E^{+}_{m}(M)$.

In our running example $D^-_m(M) = \{\{r_1\}, \{r_2\}, \{r_4\}\}$ while the only (and thus minimal) deletion-explanation is given by $\{r_1, r_2, r_4\}$.

3.4. Refined notions of diagnosis and explanation

Refined diagnoses One can generalize Definition 4 to refined changes of bridge rules, such that bridge rules that need to be applicable for consistency, become applicable by only removing some body atoms instead of all. This achieves consistency with a less drastic change of the system. Intuitively this natural extension of diagnosis may let one expect an increase in expressivity of the refined notion. As we show in the following, however, the expressivity actually does not increase.

Let $br_{ref}(M)$ denote the set of bridge rules of M where some body literals have been removed, i.e., $br_{ref}(M) = \{head(r) \leftarrow B. \mid B \subseteq Body(r), r \in br(M)\}$ (where we identify the body of a bridge rule with the set of its literals). A function $fg : br(M) \to br_{ref}(M)$ is called a *body-reduction function*; it maps bridge rules to rules where some or no body atoms are removed. In the following, for any set $R \subseteq br(M)$ we let $fg(R) = \{fg(r) \mid r \in R\}$.

Definition 9. A *refined diagnosis* is a triple (D_1, D_2, fg) consisting of sets of bridge rules $D_1, D_2 \subseteq br(M)$ and a body-reduction function $fg : br(M) \to br_{ref}(M)$, such that the resulting MCS is consistent, i.e., $M[br(M) \setminus D_1 \cup fg(D_2)] \not\models \bot$. The set of all refined diagnoses is denoted by $D^{\pm,r}(M)$.

Again, by Occam's razor, we seek minimal refined diagnoses. To that end, we seek to change a minimal set of bridge rules and within this set, we seek a minimal change of bridge rule bodies. Therefore more conservation of body atoms is considered more minimal. Formally, let fg and fg' be two body-reduction functions on $br(M)$, then fg is more conservative than fg', written $fg \le fg'$, iff for every $r \in br(M)$ it holds that $Body(fg(r)) \supseteq Body(fg'(r))$. Furthermore, we write $fg < fg'$ iff $fg \le fg'$ and $fg \ne fg'$.

A refined diagnosis $(D_1, D_2, fg) \in D^{\pm,r}(M)$ is called *minimal*, iff for every $(D'_1, D'_2, fg') \in D^{\pm,r}(M)$ such that $D'_1 \subseteq D_1$ and $D'_2 \subseteq D_2$ it holds that $D_1 = D'_1$, $D_2 = D'_2$, and $fg' \not< fg$. The set of all minimal refined diagnoses is denoted by $D^{\pm,r}_m(M)$. Observe that the conservation of the body-reduction functions only comes into play if the sets of bridge rules are minimal.

Note that one could also think of refining rules in D_1, i.e., ensuring that a rule in D_1 is not applicable by adding additional atoms to its body. But as there are no hints to which atoms should be added, such a process would result in a large and arbitrary search space. For example, adding **not** *allergy_strong_ab* to r_2 would result in

$$r'_2: \quad \big(3 : (d, m1){:}has\_marker\big) \leftarrow (2 : blood\_marker), \textbf{not}\ (1 : allergy\_strong\_ab).$$

which would make the MCS of our running example consistent. Nevertheless, such a rule does not convey any meaning beyond making the MCS consistent, therefore we disregard such kind of manipulations.

But even in the case of minimal refined diagnoses, there is little information gain: every minimal diagnosis $(D_1, D_2) \in D^{\pm}_m(M)$, together with a (witnessing) equilibrium S_w of (D_1, D_2), can be refined to a minimal diagnosis (D_1, D_2, fg) using the following *refine* function. Let $\mathcal{S}$ be the set of belief states of the MCS M, then $refine(D_2, S_w) : 2^{br(M)} \times \mathcal{S} \to (br(M) \to br_{ref}(M))$ is given by $(D_2, S_w) \mapsto fg$ where fg is the body-reduction function defined as follows:

$$fg(r) = \begin{cases} head(r) \leftarrow B. & \text{if } r \in D_2, B \subseteq Body(r), S_w \mid\!\!\leadsto head(r) \leftarrow B, \\ & \text{and for no } B \subset B' \subseteq Body(r) \text{ holds } S \mid\!\!\leadsto head(r) \leftarrow B'; \\ r & \text{otherwise.} \end{cases}$$

Observe that a refined diagnosis (D_1, D_2, fg) obtained in such way also admits the equilibrium S_w, as all rules of $fg(D_2)$ are applicable in S_w and therefore all head beliefs of D_2 are added to the respective contexts, which results in the same knowledge bases as for $cf(D_2)$.

Proposition 2. *A triple (D_1, D_2, fg) is a minimal refined diagnosis of M iff there exists a diagnosis $(D_1, D_2) \in D^{\pm}_m(M)$ and a (witnessing) equilibrium S_w, such that $refine(D_2, S_w) = fg$ and no (witnessing) equilibrium S'_w exists where $refine(D_2, S'_w) = fg'$ and $fg' < fg$.*

Refined explanations Similar to diagnoses, it is possible to consider refined modifications of rules (rather than $cf(R_2)$) in Definition 6.

Definition 10. A *refined explanation* is a triple (E_1, E_2, fg) consisting of sets of bridge rules $E_1, E_2 \subseteq br(M)$ and a body-reduction function fg, such that $M[R_1 \cup fg'(R_2)] \models \bot$ holds, for every $E_1 \subseteq R_1 \subseteq br(M)$, $R_2 \subseteq br(M)$, and every body-reduction function fg' where $r \in E_2$ implies $Body(fg(r)) \subseteq Body(fg'(r))$.

Here, we shift the "prevention of inconsistency" expressed by E_2 in Definition 6 to the body-reduction fg: we do not add unconditional bridge rules, i.e., from $br(M) \setminus E_2$, but rather consider all body-reductions fg' for which it holds that bridge rules in E_2 retain all literals indicated by fg.

Example 8. Consider a slight modification of our running example where data from the patient history is only imported in the decision support system if the patient is currently under treatment in the hospital. So bridge rule r_5 is changed to

$$r'_5: \quad (4 : allow\_strong\_ab) \leftarrow (1 : under\_treatment), \mathbf{not}\ (1 : allergy\_strong\_ab)$$

and our patient is at the hospital, i.e., $kb_1 = \{allergy\_strong\_ab, under\_treatment\}$.

Let $fg(r'_5) = (4 : allow\_strong\_ab) \leftarrow (1 : under\_treatment)$ and $fg(r) = r$ for all $r \in br(M)$ with $r \neq r'_5$. Then, $(\emptyset, \{r'_5\}, fg) \in D^{\pm,r}_m(M)$, since $fg(r'_5)$ allows the strong antibiotic if the patient merely is under treatment. The set of minimal diagnoses is the same (exchanging r_5 with r'_5) as for the running example, in particular $(\emptyset, \{r'_5\})$ is a minimal diagnosis. The refinement of this diagnosis can be computed using its (only) witnessing equilibrium

$$\begin{aligned} S_v = \big(&\{allergy\_strong\_ab, under\_treatment\}, \\ &\{blood\_marker, xray\_pneumonia\}, \\ &\big\{d{:}Pneumonia, (d, m1){:}has\_marker, d{:}AtypPneumonia\big\}, \\ &\{need\_ab, need\_strong, allow\_strong\_ab, give\_strong\}\big), \end{aligned}$$

where only the negated literal of r'_5 is deleted, this is sufficient to make the rule applicable under S_w, i.e., $fg(r'_5) = (4 : allow\_strong\_ab) \leftarrow (1 : under\_treatment)$ and $(\emptyset, \{r'_5\}, fg) \in D^{\pm,r}_m(M)$. A refined explanation is (E_1, E_2, fg) where $E_1 = \{r_1, r_2, r_4\}$, $E_2 = \{r'_5\}$, and $fg(r'_5) = (4 : allow\_strong\_ab) \leftarrow \mathbf{not}\ (1 : allergy\_strong\_ab)$. □

The notion of a refined explanation is a generalization of the notion of explanation and there is a 1-to-1 correspondence between them.

Proposition 3. *For an inconsistent MCS M, it holds that $(E_1, E_2) \in E^{\pm}(M)$ iff there exists a body-reduction function fg such that (E_1, E_2, fg) is a refined explanation.*

In contrast to diagnoses, an explanation does not admit a witnessing equilibrium. Therefore, we cannot infer from an explanation whether the addition of a reduced version of a bridge rule would yield consistency. However, this can be achieved considering a transformed MCS: Consider $M = (C_1, \ldots, C_n)$, then $M^r = (C_1, \ldots, C_n, C_\alpha)$ is the transformed MCS where C_α is a context whose acceptable belief states contain exactly those formulas added to it via bridge rules, e.g., C_α uses the logic L^{asp} and an empty knowledge base $kb_\alpha = \emptyset$. Furthermore, the bridge rules of $br(M^r)$ are obtained from $br(M)$ in such a way that every bridge rule $r \in br(M)$ of form (1) is split via transformation $tr(\cdot)$ into a core rule $(r^{(0)})$ and a supplementary rule for each body atom $(r^{(1)}, \ldots, r^{(m)})$. The set $tr(r)$ of transformed rules corresponding to r is given by:

$$\begin{aligned} tr(r) = \big\{&r^{(0)} : \ (k : s) \leftarrow (C_\alpha : p_1), \ldots, (C_\alpha : p_j), (C_\alpha : p_{j+1}), \ldots, (C_\alpha : p_m). \\ &r^{(1)} : \ (C_\alpha : p_1) \leftarrow (c_1 : p_1). \\ &\ldots \\ &r^{(j)} : \ (C_\alpha : p_1) \leftarrow (c_j : p_j). \\ &r^{(j+1)} : \ (C_\alpha : p_1) \leftarrow \mathbf{not}\ (c_{j+1} : p_{j+1}). \\ &\ldots \\ &r^{(m)} : \ (C_\alpha : p_m) \leftarrow \mathbf{not}\ (c_m : p_m).\big\} \end{aligned}$$

Finally, M^r contains for each bridge rule of M the corresponding transformed rules, i.e., $br(M^r) = \bigcup_{r \in br(M)} tr(r)$. Note that, for readability, this transformation assumes beliefs of different contexts to be disjoint.

For example, a bridge rule r: $(c_1 : h) \leftarrow (c_2 : a), \mathbf{not}\ (c_3 : b)$ of M is transformed into bridge rules $r^{(0)}$: $(c_1 : h) \leftarrow (c_\alpha : a'), (c_\alpha : b')$; $r^{(1)}$: $(c_\alpha : a') \leftarrow (c_2 : a)$; and $r^{(2)}$: $(c_\alpha : b') \leftarrow \mathbf{not}\ (c_3 : b)$ of M^r.

An explanation $(E_1, E_2) \in E^{\pm}(M^r)$ then allows to construct a refined explanation (E_1, E^r_2, fg) for M as follows: For every $r \in br(M)$, it holds that $r \in E^r_2$ iff $tr(r) \cap E_2 \neq \emptyset$. Furthermore, let $sup(r) = \{Body(r') \mid r' \in tr(r) \wedge r' \neq r^{(0)}\}$, then fg is a body-reduction function on $br(M)$ such that $fg(r) = head(r) \leftarrow sup(r)$ if $r \in E_2$ and $fg(r) = (r)$ otherwise.

For example, if the supplementary rules $(c_\alpha : a') \leftarrow (c_2 : a)$., is in E_2, then the removal of the corresponding literal, here $(c_2 : a)$, from the original bridge rule in M contributes to avoiding the explained inconsistency in M. Removal of all corresponding literals indicated by E_2 yields a change of bridge rule bodies to avoid the explained inconsistency completely.

4. Properties

In this section we first show that, to some extent, diagnoses can be converted to explanations and vice versa; specifically, minimal diagnoses and minimal explanations point out the same bridge rules, a property we call duality. We then prove a useful non-intersection property of minimal diagnoses, and show how modularity of an MCS (defined in the spirit of splitting sets of logic programs) is reflected in the structure of its diagnoses and explanations.

4.1. Converting between diagnoses and explanations

While duality expresses that minimal diagnoses and minimal inconsistency explanations point out the same set of bridge rules, in the following we consider the relationships between these notions in more detail. We show that it is possible to characterize explanations in terms of diagnoses, and vice versa minimal diagnoses in terms of minimal explanations.

For the following theorem we generalize the notion of a hitting set [84] from sets to pairs of sets. Given a collection $\mathcal{C} = \{(A_1, B_1), \dots, (A_n, B_n)\}$ of pairs of sets (A_i, B_i), $A_i, B_i \subseteq U$ over a set U, a *hitting set of* $\mathcal{C}$ is a pair of sets (X, Y), $X, Y \subseteq U$ such that for every pair $(A_i, B_i) \in \mathcal{C}$, (i) $A_i \cap X \neq \emptyset$ or (ii) $B_i \cap Y \neq \emptyset$. A hitting set (X, Y) of $\mathcal{C}$ is *minimal*, if no $(X', Y') \subset (X, Y)$ is a hitting set of $\mathcal{C}$.

We consider hitting sets over pairs of sets of bridge rules, and denote by $HS_M(\mathcal{C})$ (respectively, $minHS_M(\mathcal{C})$) the set of all (respectively, all minimal) hitting sets of $\mathcal{C}$ over $U = br(M)$. Note that in particular $HS_M(\emptyset) = \{(\emptyset, \emptyset)\}$, and $HS_M(\{(\emptyset, \emptyset)\}) = \emptyset$.

Theorem 1. *For every MCS M,*

(a) *a pair (E_1, E_2) with $E_1, E_2 \subseteq br(M)$ is an inconsistency explanation of M iff $(E_1, E_2) \in HS_M(D^{\pm}(M))$, i.e., (E_1, E_2) is a hitting set of $D^{\pm}(M)$; and*
(b) *a pair (E_1, E_2) with $E_1, E_2 \subseteq br(M)$ is a minimal inconsistency explanation of M iff $(E_1, E_2) \in minHS_M(D^{\pm}(M))$, i.e., (E_1, E_2) is a minimal hitting set of $D^{\pm}(M)$.*

Clearly, a hitting set of a collection X is the same as a hitting set of the collection of the $\subseteq$-minimal elements in X, we therefore obtain the following from Theorem 1.

Corollary 1. *For every MCS M,*

(a) *a pair (E_1, E_2) with $E_1, E_2 \subseteq br(M)$ is an inconsistency explanation of M iff $(E_1, E_2) \in HS_M(D^{\pm}_m(M))$; and*
(b) *a pair (E_1, E_2) with $E_1, E_2 \subseteq br(M)$ is a minimal inconsistency explanation of M iff $(E_1, E_2) \in minHS_M(D^{\pm}_m(M))$.*

For our next result, we use the following generalization of a well-known result for minimal hitting sets [7].

Lemma 1. *For every collection $X = \{X^1, \dots, X^n\}$ of pairs $X^i = (X^i_1, X^i_2)$ of sets, $1 \leq i \leq n$, such that X is an anti-chain w.r.t. $\subseteq$, i.e., elements in X are pairwise incomparable ($X^i \subseteq X^j$ with $1 \leq i, j \leq n$ implies $X^i = X^j$) it holds that $minHS_M(minHS_M(X)) = X$.*

Combined with Corollary 1(b) we thus obtain.

Theorem 2. *A pair (D_1, D_2) with $D_1, D_2 \subseteq br(M)$ is a minimal diagnosis of M iff (D_1, D_2) is a minimal hitting set of $E^{\pm}_m(M)$, formally $D^{\pm}_m(M) = minHS_M(E^{\pm}_m(M))$.*

As for computation, Theorem 1 provides a way to compute the set of explanations $E^{\pm}(M)$ from the set $D^{\pm}(M)$ of diagnoses, while Theorem 2 allows us to compute the set $D^{\pm}_m(M)$ of minimal diagnoses from the set of minimal explanations $E^{\pm}_m(M)$. Corollary 1 shows that, for computing $E^{\pm}(M)$ and $E^{\pm}_m(M)$, it is sufficient to know the set $D^{\pm}_m(M)$ of minimal diagnoses.

Note that Theorem 2 generalizes a result of Reiter's approach to diagnosis [84], since the former describes relationships between minimal hitting sets in a sense similar to the relationship between diagnoses and conflict sets of the latter. In contrast, note that Theorem 1(a) uses hitting sets without the requirement of $\subseteq$-minimality.

4.1.1. Duality

As it appears, explanations and diagnoses point out bridge rules as causes of inconsistency on a dual basis. Intuitively, bridge rules in E_1 of an explanation (E_1, E_2) cause inconsistency, while bridge rules in D_1 of a diagnosis (D_1, D_2) remove inconsistency; furthermore, adding unconditional forms of bridge rules from E_2 spoils inconsistency, while not adding unconditional forms of bridge rules from D_2 spoils consistency.

Both notions point out rules that are erroneous in the way that those rules contribute to inconsistency. This naturally gives rise to the question whether diagnoses and explanations point out the same rules of an MCS as erroneous, or whether they characterize different aspects.

To formalize this question, we introduce relevancy for inconsistency. Given an MCS M, a bridge rule $r \in br(M)$ is *relevant for diagnosis* (*d-relevant*) iff there exists a minimal diagnosis (D_1, D_2) of M with $r \in D_1 \cup D_2$. Analogously, r is *relevant for explanation* (*e-relevant*) iff there exists a minimal explanation with $r \in E_1 \cup E_2$.

As the following proposition shows, the component-wise coincidence is not accidental. Not only are the d-relevant rules exactly the same that are e-relevant, but this even holds if the components of diagnoses and explanations are treated separately. Formalizing this, for any set X of pairs (A, B) we write $\bigcup X$ for $(\bigcup\{A \mid (A, B) \in X\}, \bigcup\{B \mid (A, B) \in X\})$.

Proposition 4. *For every inconsistent MCS M, $\bigcup D_m^{\pm}(M) = \bigcup E_m^{\pm}(M)$, i.e., the unions of all minimal diagnoses and all minimal inconsistency explanations coincide.*

Proposition 4 is an immediate consequence of the close structural relationships between diagnoses and explanations, which are shown by Theorems 1 and 2.

This provides evidence for our view that both notions capture exactly those parts of an MCS that are relevant for inconsistency, as duality shows that two very different perspectives on inconsistency state exactly the same parts of the MCS as erroneous.

In practice this allows one to compute the set of all bridge rules which are relevant for making an MCS consistent (i.e., appear in at least one diagnosis) in two ways: either to compute all minimal explanations, or to compute all minimal diagnoses. Furthermore, the duality result allows to exclude, under Occam's razor, all bridge rules that are not part of any diagnosis (or explanation) from further investigation as they can be skipped savely.

Our running example suggests that duality also holds for deletion-diagnoses and -explanations, which indeed is true:

Theorem 3. *For every inconsistent MCS M, $\bigcup D_m^{-}(M) = \bigcup E_m^{+}(M)$, i.e., the unions of all minimal deletion-diagnoses and all minimal deletion-inconsistency explanations coincide.*

Proof. This is a direct consequence of Theorem 4 (set in its proof the second components of diagnoses and explanations to the empty set). □

4.1.2. Asymmetry

We now investigate why it is possible to obtain the set of explanations from the set of diagnoses, while the other direction only works under ⊆-minimality. The following example illustrates this.

Example 9. Consider the MCS $M = (C_1)$ with the ASP context $C_1 = \{\leftarrow a.\}$, and the bridge rules $br(M) = \{r_1 = (1:a) \leftarrow (1:a).,r_2 = (1:a) \leftarrow \text{ not } (1:b)\}$. Then $D^{\pm}(M) = \{(\{r_2\},\emptyset),(\{r_1,r_2\},\emptyset)\}$, while $E_m^{\pm}(M) = \{(\{r_2\},\emptyset)\}$, because only r_2 is relevant (cf. Section 3.2) for inconsistency. Furthermore, $E^{\pm}(M)$ contains all pointwise supersets of $(\{r_2\},\emptyset)$, viz $(\{r_2\},\emptyset)$, $(\{r_1,r_2\},\emptyset)$, $(\{r_2\},\{r_1\})$, $(\{r_2\},\{r_2\})$, $(\{r_1,r_2\},\{r_1\})$, $(\{r_1,r_2\},\{r_2\})$, and $(\{r_1,r_2\},\{r_1,r_2\})$. Now the set of (non-minimal) hitting sets of the set $E^{\pm}(M)$ of explanations is the set $E^{\pm}(M)$ itself, while the set $D^{\pm}(M)$ of diagnoses only contains two elements. □

The reason behind this asymmetry is that the notion of explanation is a monotonic (order-increasing) concept, i.e., all supersets of an explanation are also explanations, while the notion of diagnosis is not, i.e., a superset of a diagnosis is not necessarily a diagnosis.

This difference is due to the fact that explanations characterize only relevant inconsistencies (as discussed in Section 3.2) and by its definition, all supersets of an explanation are explanations. Therefore the set of minimal explanations characterizes the set of explanations. For the notion of diagnosis this is not the case: a system might contain inconsistent bridge rule configurations which do not appear in explanations because they are irrelevant in the original system. Non-minimal diagnoses provide modifications of the system which might cause and at the same time suppress such an irrelevant inconsistency in order to achieve overall consistency.

In summary, a minimal hitting set of the set of diagnoses characterizes the set of minimal explanations (Corollary 1(b)) and a minimal hitting set of the set of explanations characterizes the set of minimal diagnoses (Theorem 2). With non-minimality it looks different: the non-minimal hitting sets of $D^{\pm}(M)$ characterize the set $E^{\pm}(M)$ of explanations (see Theorem 1(a)), however the non-minimal hitting sets of $E^{\pm}(M)$ do not characterize the set $D^{\pm}(M)$ of diagnoses (see Example 9 for a counterexample).

4.2. Non-overlap in minimal diagnoses

We note a simple but useful property of minimal diagnoses. Definition 4 reveals that, (D_1, D_2) such that $r \in D_2$ is a diagnosis regardless of whether $r \in D_1$. Therefore,

Proposition 5. *Every minimal diagnosis (D_1, D_2) of an MCS M, fulfills $D_1 \cap D_2 = \emptyset$, i.e., no rule occurs in both components.*

An analogous property does not hold for inconsistency explanations: e.g., the bridge rule r: $(1:a) \leftarrow \textbf{not } (1:a)$ may cause an inconsistency which can be repaired only by adding $cf(r)$ to the system, hence the system has a minimal explanation $(\{r\},\{r\})$.

4.3. Modularity of explanations and diagnoses

We next give a syntactic criterion which enables the computation of explanations for an MCS M in a divide-and-conquer fashion. In particular, minimal explanations of M are then just combinations of the minimal explanations of the smaller parts. Based on the results about conversion between explanations and diagnoses, these results then carry over to diagnoses as well. This can be exploited to compute minimal explanations and minimal diagnoses for certain classes of MCS more efficiently.

An approach to modularization (in particular for hierarchical and partitionable MCS) is that some part does not impact the rest of the system. To this end, we adapt the notion of a *splitting set* as introduced by [71] in the context of logic programming; a splitting set characterizes a subset of a logic program which is independent of other rules in the program by a syntactic property.

Since an MCS may include contexts with arbitrary logics, a purely syntactical criterion can only be obtained by resorting to beliefs occurring in bridge rules, under the implicit assumption that every output belief of a context depends on every input belief of the context. Hence, we split at the level of contexts, i.e., a splitting set is a set of contexts rather than a set of literals.

Definition 11. A set of contexts $U \subseteq C(M)$ is a *splitting set* of an MCS M, if every rule $r \in br(M)$ such that $C_h(r) \in U$ satisfies $C_b(r) \subseteq U$. More formally, U is a splitting set iff $U \supseteq \bigcup\{C_b(r) \mid r \in br(M), C_h(r) \in U\}$. Furthermore, for such U, the set $b_U = \{r \in br(M) \mid C_h(r) \in U\}$ is called the *bottom* relative to U.

Example 10. In our running example, we have $C(M) = \{C_1, \dots, C_4\}$, with e.g., $C_h(r_1) = C_h(r_2) = C_3$, and $C_b(r_1) = C_b(r_2) = \{C_2\}$. So the set $U_1 = \{C_2, C_3\}$ is a splitting set of M; its bottom is $b_{U_1} = \{r_1, r_2\}$. The other splitting sets of M are $U_2 = \{C_1\}$ with $b_{U_2} = \emptyset$, $U_3 = \{C_2\}$ with $b_{U_3} = \emptyset$, and $U_4 = \{C_4, C_3, C_2, C_1\}$ with bottom $b_{U_4} = br_M$. □

Intuitively, if U is a splitting set of M, then the consistency (respectively inconsistency) of contexts in U does not depend on the contexts in $C(M) \setminus U$. Thus, if $M[b_U]$ is inconsistent, M stays inconsistent (under the assumption that $M[\emptyset] \not\models \bot$).

For a pair $R = (R_1, R_2)$ of sets of bridge rules compatible with M and a set U of contexts we say that R is *U-headed* iff $r \in (R_1 \cup R_2)$ implies $C_h(r) \in U$.

Proposition 6. *Suppose U is a splitting set of an MCS M. Then,*

(i) *$E \in E^{\pm}(M[b_U])$ iff $E \in E^{\pm}(M)$ and E is U-headed, and*
(ii) *$D \in D^{\pm}(M[b_U])$ iff there exists some $D' \in D^{\pm}(M)$ such that $D \subseteq D'$.*

Corollary 2. *Every minimal explanation of $M[b_U]$ is a minimal explanation of M.*

Note that $M[b_U]$ does not yield all explanations that contain rules from b_U, but it yields all explanations that contain only rules from $M[b_U]$.

In the particular case that two splitting sets form a partitioning of the MCS, then both partitions can be treated without considering the other one. This means that explanations only contain rules from one partition and diagnoses of the whole MCS are obtained by simply combining diagnoses of each of the partitions.

Proposition 7. *Suppose that both, U and $U' = C(M) \setminus U$, are splitting sets of an MCS M. Then, every $E \in E^{\pm}_m(M)$ is either U-headed or U'-headed.*

Corollary 3. *Suppose U and $U' = C(M) \setminus U$ are splitting sets of an MCS M. Then, $E^{\pm}_m(M) = E^{\pm}_m(M[b_U]) \cup E^{\pm}_m(M[b_{U'}])$.*

Thus, using U, U' the MCS M can be partitioned into two parts where minimal explanations can be computed independently. From this and Theorem 2 we can conclude that for a partitionable MCS, the set of all minimal diagnoses can be obtained by combining the minimal diagnoses of each partition.

Proposition 8. *Suppose that U and $U' = C(M) \setminus U$ are splitting sets of an MCS M. Then, $D^{\pm}_m(M) = \{(A_1 \cup B_1, A_2 \cup B_2) \mid (A_1, A_2) \in D^{\pm}_m(M[b_U])$ and $(B_1, B_2) \in D^{\pm}_m(M[b_{U'}])\}$.*

5. Computational complexity

We next consider the complexity of consistency checking, and of diagnosis and explanation recognition in MCS in a parametric fashion. To this end, we recall the complexity classes that we will use, and show that we can abstract an MCS to beliefs used in bridge rules. We use *context complexity* as a parameter to characterize the overall complexity and we establish for hardness generic results for all complexity classes that are closed under conjunction and projection. Table 1 summarizes our results for complexity classes that are typically encountered in knowledge representation.

Table 1
Complexity of consistency checking and recognizing (minimal) diagnoses and explanations, given (A,B) and an MCS M for complexity classes of typical knowledge-representation formalisms. Membership holds for all cases, completeness holds if at least one context is complete for the respective context complexity.

Context complexity $\mathcal{CC}(M)$	Consistency checking MCSEQ	$(A,B) \overset{?}{\in}$			
		$D^{\pm}(M)$ MCSD	$D^{\pm}_m(M)$ MCSD$_m$	$E^{\pm}(M)$ MCSE	$E^{\pm}_m(M)$ MCSE$_m$
P	**NP**	**NP**	$\mathbf{D^P_1}$	**coNP**	$\mathbf{D^P_1}$
NP	**NP**	**NP**	$\mathbf{D^P_1}$	**coNP**	$\mathbf{D^P_1}$
$\mathbf{\Sigma^P_i}$, $i \geq 1$	$\mathbf{\Sigma^P_i}$	$\mathbf{\Sigma^P_i}$	$\mathbf{D^P_i}$	$\mathbf{\Pi^P_i}$	$\mathbf{D^P_i}$
PSPACE	**PSPACE**	**PSPACE**	**PSPACE**	**PSPACE**	**PSPACE**
EXPTIME	**EXPTIME**	**EXPTIME**	**EXPTIME**	**EXPTIME**	**EXPTIME**
Proposition	9	10	11	12	13

5.1. Complexity classes

Recall that **P**, **EXPTIME**, and **PSPACE** are the classes of problems that can be decided using a deterministic Turing machine in polynomial time, exponential time, and polynomial space, respectively. Furthermore **NP** (resp., **coNP**) is the class of problems that can be decided on a nondeterministic Turing machine in polynomial time, where one (resp., all) execution paths accept. Recall the polynomial hierarchy, where $\mathbf{\Sigma^P_0} = \mathbf{\Pi^P_0} = \mathbf{P}\mathbf{\Sigma^P_i}$ is **NP** with a $\mathbf{\Sigma^P_{i-1}}$ oracle, and $\mathbf{\Pi^P_i}$ is **coNP** with a $\mathbf{\Sigma^P_{i-1}}$ oracle.

Given complexity class C, we denote by $\mathbf{D}(C)$ the "difference class" of C, i.e., $\mathbf{D}(C) = \{L_1 \times L_2 \mid L_1 \in C,\ L_2 \in \mathbf{co}\text{-}C\}$ denotes the complexity class of decision problems that are the "conjunction" of a problem L_1 in C and a problem L_2 in **co**-C. Two particular classes that we use are $\mathbf{D^P_1} = \mathbf{D}(\mathbf{NP})$ and $\mathbf{D^P_i} = \mathbf{D}(\mathbf{\Sigma^P_i})$. A prototypical problem complete for $\mathbf{D^P_1}$ is deciding, given a pair (F_1, F_2) of propositional Boolean formulas, whether F_1 is satisfiable and F_2 is unsatisfiable. Note in particular that $\mathbf{D}(\mathbf{PSPACE}) = \mathbf{PSPACE}$ and that $\mathbf{D}(\mathbf{EXPTIME}) = \mathbf{EXPTIME}$.

Closure under conjunction and projection A complexity class C is *closed under conjunction*, if the following holds: given a problem L in C, it holds that the problem $L' = \{(I_1, \dots, I_n) \mid n \geq 0, I_j \in L\}$ is also in C, where the 'yes' instances of L' are arbitrarily long but finite list of 'yes' instances of L. All classes **P**, **NP**, $\mathbf{\Sigma^P_i}$, $\mathbf{\Pi^P_i}$, $D(\mathbf{\Sigma^P_i})$, **PSPACE**, etc. here are closed under conjunction.

A decision problem $L \subseteq \Sigma^\star \times \Sigma^\star$, where $\Sigma^\star$ is as usual the set of all finite strings over Σ, is *polynomially balanced*, if some polynomial p exists such that $|I'| \leq p(|I|)$ for all $(I, I') \in L$. Moreover, L is a *polynomial projection* of $L' \subseteq \Sigma^\star \times \Sigma^\star$ if $L = \{I \mid \exists I' : (I, I') \in L'\}$ and L' is polynomially balanced (intuitively, I' is a witness of polynomial size for I). Given a complexity class C, let $\pi(C)$ contain all problems which are a polynomial projection of a problem L' in C. Then a complexity class C is *closed under projection* if $\pi(C) \subseteq C$. The classes $\mathbf{\Sigma^P_i}$, **NP**, **EXPTIME**, **PSPACE** are closed under projection, while **coNP** and $\mathbf{\Pi^P_i}$ are presumably not. For further background see [76].

5.2. Output-projected equilibria

Computing equilibria by guessing and verifying so-called "kernels of context belief sets" has been outlined in [32]. For the purpose of recognizing diagnoses and explanations, it suffices to check for consistency, i.e., for existence of an arbitrary equilibrium in an MCS.

Here we first define *output beliefs*, which are the beliefs used in bodies of bridge rules. Then we show that for checking consistency of an MCS, it is sufficient to consider equilibria *projected to output beliefs*.

Definition 12. Given an MCS $M = (C_1, \dots, C_n)$, the *set of output beliefs of* C_i, $OUT_i = \{p \mid (i:p) \in body(r), r \in br(M)\}$, is the set of beliefs p of C_i that occur in the bodies of bridge rules.

Using the notion of output beliefs, we let $S^o_i = S_i \cap OUT_i$ be the projection of S_i to OUT_i, and for any belief state $S = (S_1, \dots, S_n)$ we let $S^o = (S^o_1, \dots, S^o_n)$ be the *output-projected belief state* S^o of S.

An output-projected belief state provides sufficient information for evaluating the applicability of bridge rules. We next show how to obtain witnesses for equilibria using this projection.

Definition 13. An output-projected equilibrium of an MCS M is an output-projected belief state $T = (T_1, \dots, T_n)$ such that for all $1 \leq i \leq n$,

$$T_i \in \left\{S^o_i \;\middle|\; S_i \in \mathbf{ACC}_i\left(kb_i \cup \left\{\varphi(r) \;\middle|\; r \in app(br_i, T_i)\right\}\right)\right\}.$$

T contains information about all (and only about) output beliefs. As these are the beliefs that determine bridge rule applicability, in every equilibrium S, $app(R, S) = app(R, S^o)$; thus we obtain:

Lemma 2. *For each equilibrium S of an MCS M, S^o is an output-projected equilibrium. Conversely, for each output-projected equilibrium T of M, there exists some equilibrium S of M such that $S^o = T$.*

Given an MCS M, we denote by $\mathrm{EQ}^o(M)$ the set of output-projected equilibria of M. Every equilibrium is witnessed by a single output-projected equilibrium, and every output-projected equilibrium witnesses at least one equilibrium. For consistency checking (i.e., equilibrium existence) it is therefore sufficient to consider output-projected equilibria.

5.3. Context complexity

The complexity of consistency checking for an MCS clearly depends on the complexity of its contexts. We next define a notion of *context complexity* by considering the roles which contexts play in the problem of consistency checking.

For all complexity considerations, we represent logics L_i of contexts C_i *implicitly*; they are fixed and we do not consider these (possibly infinite) objects to be part of the input of the decision problems we investigate. Accordingly, the instance size of a given MCS M will be denoted by $|M| = |kb_M| + |br(M)|$ where $|kb_M|$ denotes the size of knowledge bases in M and $|br(M)|$ denotes the size of its set of bridge rules.

Consistency of an MCS M can be decided by a Turing machine with input M which (a) guesses an output-projected belief state $T \in OUT_1 \times \cdots \times OUT_n$, (b) evaluates the bridge rules on T, yielding for each context C_i a set of active bridge rule heads H_i w.r.t. T, and (c) checks for each context whether it accepts the guessed T w.r.t. H_i. We call the complexity of step (c) *context complexity*, formalized as follows.

Definition 14. Given a context $C_i = (kb_i, br_i, L_i)$ and a pair (H, T_i), with $H \subseteq IN_i$ and $T_i \subseteq OUT_i$, the *context complexity* $\mathcal{CC}(C_i)$ of C_i is the computational complexity of deciding whether there exists an $S_i \in \mathbf{ACC}_i(kb_i \cup H)$ such that $S_i \cap OUT_i = T_i$.

We now give examples for context complexities of logics in Example 1, then we give context complexities for other well-known knowledge-based formalisms. Contexts with *propositional logic* L^c_Σ have $\mathbf{D^P_1}$-complete context complexity. On the other hand, the restricted logic L^{pl}_Σ which is used in our running example for contexts C_1 and C_2, is tractable: the context complexity is $\mathcal{O}(n)$. Default Logic programs and *disjunctive logic programs* (L^{asp}_Σ) have $\mathbf{\Sigma^P_2}$-complete acceptability checking and thus also complexity [29,56]. For contexts hosting ontological reasoning in the *Description Logic* $\mathcal{ALC}$ ($L_\mathcal{A}$) acceptability checking corresponds to a set of instance checks. As individual instance checking is **EXPTIME**-complete [2] and **EXPTIME** is closed under conjunction, such a context is in **EXPTIME**. For $|OUT_i| = 1$ we see that such a context is also **EXPTIME**-hard. Therefore a context using logic $L_\mathcal{A}$ has context complexity **EXPTIME**. A *relational database* can be captured by knowledge bases and belief sets which are sets of tuples in relations. Acceptability of a belief set computes whether a belief set is the closure of a knowledge base w.r.t. a fixed set of (possibly recursive) Datalog view definitions. Such a context is complete for **P** [29]. A *propositional answer set program* can be captured by a context where knowledge bases are sets of rules and belief sets are sets of propositions. Acceptability of such a context then checks whether a set of propositions is an answer set of a knowledge base. Such a context is complete for **NP** [29]. Similarly, *satisfiability checking of Boolean formulas* can be captured by **NP** contexts. An agent using one of the widely-known *modal logics* K_n, T_n, or $S4_n$ with $n \geq 1$ knowledge operators can be represented as a context. Assuming that such a context has knowledge bases and belief sets consisting of formulas, and that the context accepts the closure $\mathcal{C}_X$ of a set of formulas X in the knowledge base, this context is **PSPACE**-complete [58].

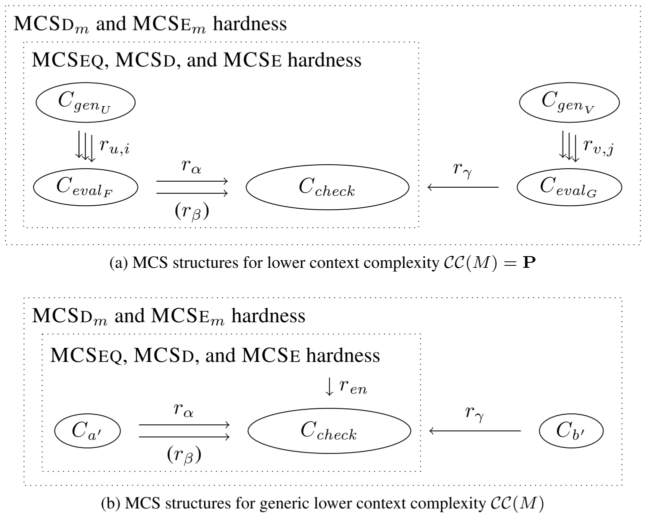
Given an MCS M, we say M has *upper context complexity* C, denoted $\mathcal{CC}(M) \leq C$, if $\mathcal{CC}(C_i) \subseteq C$ for every context C_i of M; We say M has *lower context complexity* C, denoted $\mathcal{CC}(M) \geq C$, if $C \subseteq \mathcal{CC}(C_i)$ for some context C_i of M. We say that M has *context complexity* C, denoted $\mathcal{CC}(M) = C$, iff $\mathcal{CC}(M) \leq C$ and $\mathcal{CC}(M) \geq C$. That is, if $\mathcal{CC}(M) = C$ all contexts in M have complexity at most $\mathcal{CC}(M)$, and some context in M has C-complete complexity, provided the class C has complete problems.

Example 11. In our running example, for $M = (C_1, C_2, C_3, C_4)$ we have $\mathcal{CC}(C_1) = \mathcal{CC}(C_2) = \mathcal{O}(n)$, $\mathcal{CC}(C_3) =$ **EXPTIME**, and $\mathcal{CC}(C_4) = \mathbf{\Sigma^P_2}$. As $\mathcal{O}(n) \subseteq \mathbf{\Sigma^P_2} \subseteq$ **EXPTIME**, we obtain $\mathcal{CC}(M) \leq$ **EXPTIME**, and as C_2 is **EXPTIME**-complete, we obtain $\mathcal{CC}(M) \geq$ **EXPTIME**; hence $\mathcal{CC}(M) =$ **EXPTIME**. □

5.4. Overview of complexity results

We now give an overview of complexity results, and brief intuition about the proofs that are available in Appendices A–C.

We study the decision problem for consistency (MCSEQ) and recognition problems for diagnoses (MCSD), minimal diagnoses (MCSD$_m$), explanations (MCSE), and minimal explanations (MCSE$_m$). Note that *existence* of diagnoses and explanations is trivial by our assumptions that M is inconsistent and that $M[\emptyset]$ is consistent.



(a) MCS structures for lower context complexity $\mathcal{CC}(M) = \mathbf{P}$

(b) MCS structures for generic lower context complexity $\mathcal{CC}(M)$

Fig. 3. MCS structures for hardness reductions, where dotted areas indicate parts of the MCS used for respective reductions.

Table 1 summarizes our results for context complexities that are present in typical monotonic and nonmonotonic KR formalisms. Corresponding theorems are given in Section 5.6, which are more general than the results shown in Table 1.

For a given context complexity $\mathcal{CC}(M)$ of an MCS M, MCSEQ has the same computational complexity as MCSD. If context complexity is **NP** or above, this complexity is equal to context complexity; for context complexity **P**, it is **NP**. Intuitively, this is explained as follows: for context complexity **NP** and above, guessing a belief state and checking whether it is an equilibrium can be incorporated into context complexity without exceeding checking cost; if context complexity is **P**, this complexity is **NP**.

Recognizing minimal diagnoses MCSD$_m$ is complete for the complexity of MCSD, which captures diagnosis recognition, and an additional complementary problem of refuting MCSD, which captures diagnosis minimality recognition. For context complexity **P** we have that MCSD$_m$ is $\mathbf{D^P}$-complete.

The complexity of MCSE is in the complementary class of the corresponding problem MCSD. Intuitively this is because diagnosis involves existential quantification and explanation involves universal quantification. Accordingly, the complexity of MCSE$_m$ is complementary to MCSD$_m$. As the complexity classes of MCSD$_m$ are closed under complement, MCSE$_m$ and MCSD$_m$ have the same complexity.

These results show that minimal diagnosis and minimal explanation recognition are harder than checking consistency (under usual complexity assumptions), while they are polynomially reducible to each other.

5.5. *Proof outline*

We treat context complexity of **NP** and above uniformly and the case of **P** separately. For hardness results we use MCS structures depicted in Fig. 3.

For context complexity **P** we use reductions from SAT, UNSAT or SAT-UNSAT instances F and/or G to MCS with context complexity **P**. These reductions use the structure shown in Fig. 3a, where contexts C_{gen_U} and C_{gen_V} generate a set of possible truth assignments to sets of variables, C_{eval_F} and C_{eval_G} evaluate formulas F and G under these assignments, and C_{check} checks whether the formulas are satisfiable and/or unsatisfiable. We obtain the hardness via the nondeterministic guess that arises from the different belief sets accepted by contexts C_{gen_U} and C_{gen_V}. (See also the description of logic L_{GUESS} in the following.) Our reductions use an acyclic system topology without negation as failure in bridge rules. Note that hardness can also be obtained using a nonmonotonic guess in cyclic bridge rules which contain negation as failure; in that case all contexts of the reduction can be deterministic, i.e., every context accepts at most one belief set for any input. We give such an alternative hardness reduction in the proof of Proposition 9, where we prove **NP** hardness of MCSEQ in an MCS of context complexity **P**.

Hardness results for context complexity **NP** and above are established by a generic reduction: we reduce the problem of acceptability checking of contexts C_a (resp., C_b) with context complexity X to decision problems in an MCS M with complexity X. These reductions use the scheme shown in Fig. 3b, where $C_{a'}$ (resp., $C_{b'}$) evaluates the acceptability checking problem of C_a (resp., C_b), and C_{check} tests whether the original problems are "yes" or "no" instances.

For hardness reductions we use the following context logics.

L_{ASP} is a logic for contexts that contain stratified propositional ASPs with constraints. More in detail, if $L_{ASP} = (\mathbf{BS}, \mathbf{KB}, \mathbf{ACC})$, then **BS** is the collection of sets of atoms over a propositional alphabet Σ, **KB** is a set of logic programming rules over Σ, and given a knowledge base $kb \in \mathbf{KB}$, we define $\mathbf{ACC}(kb) = \mathcal{AS}(kb)$, i.e., the context accepts the set of answer sets of the logic program kb. If clear, Σ is omitted. In case of stratified propositional ASPs with constraints, a program has at most one answer set. From [29, Theorem 4.2] it follows that whether an atom A is part of this model is **P**-complete. Thus, deciding given OUT_i whether $T_i \subseteq OUT_i$ is a projected accepted belief set, is **P**-complete; therefore context complexity is **P**.

$L_{GUESS(B)}$ is a trivial logic over the set B that accepts all subsets of its knowledge base. In detail, if logic $L_{GUESS(B)} = (\mathbf{BS}, \mathbf{KB}, \mathbf{ACC})$ then $\mathbf{BS} = \mathbf{KB} = 2^B$ is the powerset of B, and $\mathbf{ACC}(kb) = 2^{kb}$ for $kb \in \mathbf{KB}$. If clear, B is omitted. The check whether belief set T_i is accepted by knowledge base kb_i can be done in time $\mathcal{O}(|kb_i| + |T_i|)$.

5.6. Detailed results

We first formally define the decision problems we consider and then report the complexity results.

Definition 15. Given an MCS M, MCSEQ is the problem of deciding whether M has an equilibrium.

Definition 16. Given an MCS M and a pair (A, B) with $A, B \subseteq br(M)$,

- MCSD decides whether $(A, B) \in D^{\pm}(M)$, i.e., whether (A, B) is a diagnosis of M;
- MCSD$_m$ decides whether $(A, B) \in D^{\pm}_m(M)$, i.e., whether (A, B) is a minimal diagnosis of M;
- MCSE decides whether $(A, B) \in E^{\pm}(M)$, i.e., whether (A, B) is an inconsistency explanation of M; and
- MCSE$_m$ decides whether $(A, B) \in E^{\pm}_m(M)$, i.e., whether (A, B) is a minimal inconsistency explanation of M.

We next formulate the complexity results.

Proposition 9. *The problem* MCSEQ *is*

- **NP**-*complete if* $\mathcal{CC}(M) = \mathbf{P}$, *and*
- C-*complete if* $\mathcal{CC}(M) = C$ *and* C *is a class with complete problems that is closed under conjunction and projection.*

Diagnosis recognition can be done by transforming the MCS using the given diagnosis candidate and deciding MCSEQ. On the other hand, MCSEQ can be reduced to diagnosis recognition of the empty diagnosis candidate $(\emptyset, \emptyset)$. Therefore, diagnosis recognition has the same complexity as consistency checking.

Proposition 10. *The problem* MCSD *is*

- **NP**-*complete if* $\mathcal{CC}(M) = \mathbf{P}$, *and*
- C-*complete if* $\mathcal{CC}(M) = C$ *and* C *is a class with complete problems that is closed under conjunction and projection.*

Deciding whether a pair (A, B) is a $\subseteq$-minimal diagnosis of an MCS M requires two checks: (a) whether (A, B) is a diagnosis, and (b) whether no pair $(A', B') \subset (A, B)$ is a diagnosis. The pair (A, B) is a minimal diagnosis iff both checks succeed. This intuitively leads to the following complexity result.

Proposition 11. *The problem* MCSD$_m$ *is*

- $\mathbf{D}^{\mathbf{P}}_{\mathbf{1}}$-*complete if* $\mathcal{CC}(M) = \mathbf{P}$,
- $\mathbf{D}(C)$-*complete if* $\mathcal{CC}(M) = C$ *and* C *is a class with complete problems that is closed under conjunction and projection.*

Note that, as shown in Table 1, the second item implies that MCSD$_m$ is $\mathbf{D}^{\mathbf{P}}_{\mathbf{i}}$-complete if $\mathcal{CC}(M)$ is complete for $\mathbf{\Sigma}^{\mathbf{P}}_{\mathbf{i}}$ with $i \geq 1$.

Refuting a candidate (A, B) as an explanation of M can be done by guessing a pair of sets (R_1, R_2) from Definition 6 and checking that $M[R_1 \cup cf(R_2)]$ is inconsistent. Then (A, B) is a yes instance iff all guesses succeed, which leads to complementary complexity of consistency checking for that problem. Hardness for context complexity classes C that are closed under conjunction and projection is established via reducing two contexts of complexity C to an MCS which (a) is consistent if both instances are 'yes' instances, (b) has a minimal diagnosis D if both instances are 'no' instances, and (c) has a nonempty minimal diagnosis which is a subset of D if one is a 'yes' and the other a 'no' instance. For context complexity **P** we use a similar approach with two SAT instances.

Proposition 12. *The problem* MCSE *is*

- **coNP**-*complete if* $\mathcal{CC}(M) = \mathbf{P}$, *and*
- **co**-C-*complete if* $\mathcal{CC}(M) = C$ *and* C *is a class with complete problems that is closed under conjunction and projection.*

Note that, as shown in Table 1, the second item implies that MCSE is $\mathbf{\Pi_i^P}$-complete if $\mathcal{CC}(M)$ is complete for $\mathbf{\Sigma_i^P}$ with $i \geq 1$.

For complexity results of recognizing minimal explanations we need the following lemma which limits the number of explanations that need to be checked to verify subset-minimality.

Lemma 3. *An explanation $Q = (Q_1, Q_2)$ is $\subseteq$-minimal iff no pair $(Q_1, Q_2 \setminus \{r\})$ with $r \in Q_2$ is an explanation and no pair $(Q_1 \setminus \{r\}, Q_2)$ with $r \in Q_1$ is an explanation.*

Hence, we can check subset-minimality of explanations by deciding whether for linearly many subsets of the candidate (A, B), none is an explanation, i.e., whether for each subset, some (R_1, R_2) exists s.t. $M[R_1 \cup cf(R_2)]$ is consistent. As **NP** (resp., $\mathbf{\Sigma_i^P}$) is closed under conjunction and projection, this check is in **NP** (resp., $\mathbf{\Sigma_i^P}$). In combination with checking whether the candidate is an explanation, this leads to a complexity of $\mathbf{D_1^P}$ (resp., $\mathbf{D_i^P}$). For context complexity $C \in$ **PSPACE** (resp., $C \in$ **EXPTIME**), $\mathbf{D}(C) = C$. The hardness reduction for MCSE$_m$ is very similar to the one for MCSD$_m$.

Proposition 13. *The problem* MCSE$_m$ *is*

- $\mathbf{D_1^P}$*-complete if* $\mathcal{CC}(M) = \mathbf{P}$,
- *complete for* $\mathbf{D}(C)$ *if* $\mathcal{CC}(M) = C$ *and* C *is a class with complete problems that is closed under conjunction and projection.*

6. Computation

In this section, we first recall HEX-programs, which extend answer set programs, then show how to use HEX to compute diagnoses and explanations of MCS, and finally give an overview of the MCS-IE tool,[1] which is an open source experimental prototype.

6.1. Preliminaries: HEX-programs

HEX-programs [41,42] extend disjunctive logic programs by allowing for access to external information with *external atoms*, and by *predicate variables*. In this paper, we only use ground (variable-free) HEX-programs and thus recall simplified definitions.

Let $\mathcal{C}$ and $\mathcal{G}$ be mutually disjoint sets of *constants* and *external predicate names*, respectively. Elements from $\mathcal{G}$ are prefixed with "&". An *ordinary atom* is a formula $p(c_1, \ldots, c_n)$ where $p, c_1, \ldots, c_n$ are constants. An *external atom* is a formula $\&g[\vec{v}](\vec{w})$, where $\vec{v} = Y_1, \ldots, Y_n$ and $\vec{w} = X_1, \ldots, X_m$ are two lists of constants (called *input* and *output* lists, respectively), and $\&g \in \mathcal{G}$ is an external predicate name. Intuitively, an external atom provides a way for deciding the truth value of tuple $\vec{w}$ depending on the extension of input predicates $\vec{v}$. For example the external predicate named &*temp* could measure temperature with a particular sensor: atom &*temp*[*front*](5, 7) then is true if the 'front' sensor detects a value between 5 and 7.

A HEX *rule* r is of the form

$$\alpha_1 \vee \cdots \vee \alpha_k \leftarrow \beta_1, \ldots, \beta_m, \text{not } \beta_{m+1}, \ldots, \text{not } \beta_n \quad m, k \geq 0, \tag{2}$$

where all α_i are ordinary atoms and all β_j are ordinary or external atoms. Rule r is a *constraint*, if $k = 0$; it is a *fact* if $n = 0$ (in this case we omit $\leftarrow$). A HEX-*program* (or *program*) is a finite set of HEX rules, it is *ordinary* if it contains only ordinary atoms.

The semantics of HEX-programs is defined as a conservative extension of answer sets as in [52] (see also Appendix C. HEX-programs can be evaluated using the dlvhex solver.[2] A detailed comparison of HEX programs and MCS, showing similarities and differences, is given in [32].

6.2. Computing diagnoses

We can compute diagnoses for some MCS M by guessing a candidate diagnosis and output beliefs of contexts, defining bridge rule applicability, and checking with external atoms whether each context accepts the guessed output beliefs given applicable bridge rule inputs.

We only consider diagnoses (D_1, D_2) where $D_1 \cap D_2 = \emptyset$; diagnoses with $D_1 \cap D_2 \neq \emptyset$ are trivially obtained from these, and they are never minimal (cf. Proposition 5); while we are often interested only in the latter.

Given an MCS M, we assemble a HEX-program $P_p^D(M)$ as follows. For each bridge rule $r \in br(M)$, we add the following guessing rule. Here and in the following, we simply write r for a constant symbol denoting r.

[1] http://www.kr.tuwien.ac.at/research/systems/mcsie/.
[2] http://www.kr.tuwien.ac.at/research/systems/dlvhex/.

$$um(r) \vee d_1(r) \vee d_2(r). \tag{3}$$

Intuitively, the predicates d_1 and d_2 hold bridge rules that are removed from M; respectively are added in unconditional form to M; um denotes unmodified bridge rules.

We guess presence or absence of each output belief p of each context in M.

$$pres_i(p) \vee abs_i(p). \quad \text{for every } p \in OUT_i,\ 1 \le i \le n. \tag{4}$$

Given an interpretation I of $P_p(M)$, we use $A_i(I) = \{p \mid pres_i(p) \in I\}$, $1 \le i \le n$, to denote the set of output beliefs at context C_i, corresponding to the guess in (4).

We evaluate each bridge rule (1) by two corresponding HEX rules, depending on output beliefs guessed in (4) and on the diagnosis guessed in (3).

$$\begin{aligned} in_i(s) \leftarrow\ & \text{not } d_1(r), pres_{c_1}(p_1), \ldots, pres_{c_j}(p_j), \\ & \text{not } pres_{c_{j+1}}(p_{j+1}), \ldots, \text{not } pres_{c_m}(p_m). \qquad (5) \\ in_i(s) \leftarrow\ & d_2(r). \qquad (6) \end{aligned}$$

If $d_1(r)$ is true, then (5) becomes inactive corresponding to removing r from the MCS; conversely if $d_2(r)$ is true, then (6) will be applicable just as if we would add $cf(r)$ to the MCS. Given an interpretation I of $P_p(M)$, we use $B_i(I) = \{s \mid in_i(s) \in I\}$ to denote the set of bridge rule heads at context C_i, activated by the output-projected belief state $\mathcal{A}(I) = (A_1(I), \ldots, A_n(I))$.

Finally, we ensure that answer sets of $P_p(M)$ correspond to output-projected equilibria by checking whether each context C_i accepts the guessed $A_i(I)$ w.r.t. the set $B_i(I)$ of bridge rule heads activated by bridge rules. For that, we create an external atom $\&con\_out_i[pres_i, b_i]()$ which computes $\mathbf{ACC}_i$ in an external computation. This external atom returns true iff context C_i, when given $B_i(I)$, accepts a belief set S_i such that its projection to output-beliefs OUT_i is equal to $A_i(I)$. Formally,

$$f_{con\_out_i}(I, pres_i, in_i) = 1 \quad \text{iff} \quad A_i(I) \in \{S_i^o \mid S_i \in \mathbf{ACC}_i(kb_i \cup B_i(I))\}.$$

We complete $P_p^D(M)$ by adding the following constraints.

$$\leftarrow \text{not } \&con\_out_i[pres_i, b_i](). \quad \text{for every } i \text{ with } 1 \le i \le n. \tag{7}$$

The answer sets of $P_p^D(M)$ then correspond to the diagnoses and the output-projected equilibria of the modified/repaired MCS as follows.

Theorem 4. *Let M be an MCS, and let $P_p^D(M)$ be as above. Then*

(i) *for each answer set I of $P_p^D(M)$, the pair $(D_{I,1}, D_{I,2}) = (\{r \in br(M) \mid d_1(r) \in I\}, \{r \in br(M) \mid d_2(r) \in I\})$ is a diagnosis of M and $\mathcal{A}(I) = (A_1(I), \ldots, A_n(I))$ is an output-projected equilibrium of $M[br(M) \setminus D_{I,1} \cup cf(D_{I,2})]$; and*
(ii) *for each diagnosis $(D_1, D_2) \in D^{\pm}(M)$ where $D_1 \cap D_2 = \emptyset$, and for each output-projected equilibrium T of $M[br(M) \setminus D_1 \cup cf(D_2)]$, there exists an answer set I of $P_p^D(M)$ such that $(D_1, D_2) = (D_{I,1}, D_{I,2})$ and $T = \mathcal{A}(I)$.*

Note that the above encoding computes all diagnoses. Computing only $\subseteq$-minimal diagnoses is possible with HEX, however this requires a more involved encoding using saturation (see the next section) or other external atoms. Our tool MCS-IE obtains the $\subseteq$-minimal diagnoses by filtering the diagnoses generated via the encoding $P_p^D(M)$.

6.3. Computing explanations

We next address computing explanations and present an encoding in HEX. This encoding is more involved since explanations show relevant inconsistencies only and this relevancy requires to check that all pairs of sets of bridge rules in the explanation range yield inconsistent systems. Given an explanation candidate $E = (E_1, E_2) \in 2^{br(M)} \times 2^{br(M)}$, the *explanation range* of E is

$$Rg(E) = \{(R_1, R_2) \mid E_1 \subseteq R_1 \subseteq br(M) \text{ and } R_2 \subseteq br(M) \setminus E_2\}.$$

Intuitively, $Rg(E)$ are "relevant pairs" for E. It follows directly from Definition 6 that, $E = (E_1, E_2) \in E^{\pm}(M)$ iff $M[R_1 \cup cf(R_2)] \models \bot$ for all $(R_1, R_2) \in Rg(E)$.

So, the computational complexity of diagnosis recognition is not the same as the one for explanation recognition (for $\mathcal{CC}(M)$ being **P** it is **NP** versus **coNP**). In the following we present a direct encoding, $P_p^E(M)$, in HEX using a technique from answer-set programming called saturation (cf. [40,69]). We first guess an explanation candidate $E = (E_1, E_2)$ and then ensure via saturation, that for all pairs of sets $(R_1, R_2) \in Rg(E)$ the modified system is inconsistent, i.e., we check for every $(R_1, R_2) \in Rg(E)$ and for every belief state S, that some context does not accept S under the bridge rules $R_1 \cup cf(R_2)$.

For each $r \in br(M)$, $P^E_P(M)$ contains the following rules to guess an explanation candidate.

$$e1(r) \vee ne1(r). \tag{8}$$

$$e2(r) \vee ne2(r). \tag{9}$$

To give some intuition of the saturation technique, assume that I is the (partial) interpretation corresponding to an explanation candidate guessed by the above rules. To check that every $(R_1, R_2) \in Rg(E)$ yields an inconsistent system, saturation is used as follows: via disjunctive rules, $(R_1, R_2) \in Rg(E)$ is guessed as well as a belief state S. If S is not an equilibrium for $M[R_1 \cup cf(R_2)]$, then the atom *spoil* is concluded to be true. This in turn leads to the truth of all other atoms that occur in rules to guess R_1, R_2, S, and all other atoms that are necessary to check that S is not an equilibrium. The resulting interpretation, $I^\star$, is said to be saturated (or spoiled); formally, it contains I_{spoil}, which is given by:

$$\begin{aligned} I_{spoil} = {} & \bigl\{r1(r), nr1(r), r2(r), nr2(r), body(r) \bigm| r \in br(M)\bigr\} \cup \\ & \bigl\{in_i(b) \bigm| r \in br(M) \wedge C_h(r) = i \wedge \varphi(r) = b\bigr\} \cup \{spoil\} \cup \\ & \bigcup_{a \in OUT_i} \bigl\{pres_i(a), abs_i(a)\bigr\} \cup \bigcup_{b \in IN_i} \bigl\{in_i(b)\bigr\}. \end{aligned}$$

Most importantly, $I^\star$ is a maximal model of $fP^E_P(M)^I$ and every other guess for (R_1, R_2) and S will result in the same interpretation $I^\star$, if S is not an equilibrium of $M[R_1 \cup cf(R_2)]$.

On the other hand, if there is a guess for (R_1, R_2) and S such that S is an equilibrium of $M[R_1 \cup cf(R_2)]$, then the corresponding interpretation I' will not be saturated. Since $I^\star$ is a maximal model, it then holds that $I' \subset I^\star$, hence $I^\star$ is not a minimal model of $fP^E_P(M)^I$. Thus, if $I^\star$ is indeed the minimal model of $fP^E_P(M)^I$, then there cannot exist such an I', i.e., for all (R_1, R_2) and S it then holds that S is not an equilibrium of $M[R_1 \cup cf(R_2)]$.

Since we are only interested in explanation candidates E where no equilibrium exists for any $(R_1, R_2) \in Rg(E)$, a constraint is added to ensure that only saturated models comprise an answer set, i.e, we ensure that only $I^\star$ may yield an answer set.

To generate $(R_1, R_2) \in Rg(E)$, for every $r \in br(M)$ we have the following rules:

$$r1(r) \leftarrow e1(r). \tag{10}$$

$$r1(r) \vee nr1(r) \leftarrow ne1(r). \tag{11}$$

$$nr2(r) \leftarrow e2(r). \tag{12}$$

$$r2(r) \vee nr2(r) \leftarrow ne2(r). \tag{13}$$

We guess a belief state of M, so $P^E_P(M)$ contains for each $a \in OUT_i$ with $1 \le i \le n$ the following rule:

$$pres_i(a) \vee abs_i(a). \tag{14}$$

Recall that I is an answer set of $P^E_P(M)$ iff I is a $\subseteq$-minimal model of $fP^E_P(M)^I$. As we use saturation and external atoms, this can lead to the undesired effect that some $r \in fP^E_P(M)^I$ is unsupported, i.e., for a being the head of r it can happen that $a \in I$ but the body of r is false under I and no other rule's body with head a is true. To avoid this, each bridge rule of M is encoded such that $a \in I$ implies that a corresponding body also evaluates to true. This is achieved by the addition of a unique atom $body(r)$ for each $r \in br(M)$ and by further rules ensuring that each literal in the body of r holds if $body(r) \in I$. $P^E_P(M)$ contains for each $r \in br(M)$ of form $(i : b) \leftarrow (i_1 : b_1), \ldots, (i_{k-1} : b_{k-1}), not(i_k : b_k), \ldots, not(i_m : b_m)$ the following rules:

$$body(r) \leftarrow r1(r), pres_{i_1}(b_1), \ldots, pres_{i_{k-1}}(b_{k-1}), abs_{i_k}(b_k), \ldots, abs_{i_m}(b_m). \tag{15}$$

$$r1(r) \leftarrow body(r). \tag{16}$$

$$pres_{i_1}(b_1) \leftarrow body(r). \tag{17}$$

$$\ldots$$

$$pres_{i_{k-1}}(b_{k-1}) \leftarrow body(r). \tag{18}$$

$$abs_{i_k}(b_k) \leftarrow body(r). \tag{19}$$

$$\ldots$$

$$abs_{i_m}(b_m) \leftarrow body(r). \tag{20}$$

$$in_i(b) \leftarrow body(r). \tag{21}$$

$$in_i(b) \leftarrow r2(r). \tag{22}$$

Rules (21) and (22) ensure that the head of r is derived if either the body holds, or if r is unconditional, i.e., $r \in R_2$. For the head $(i:b)$ of r, let $[(i:b)]$ be the set of bridge rules whose head is the same, i.e., $[(i:b)] = \{r \in br(M) \mid C_h(r) = i \wedge \varphi(r) = b\}$. For each head $(i:b)$ of a bridge rule with $[(i:b)] = \{r_1, \dots, r_k\}$ the following rule of $P^E_P(M)$ ensures that $(i:b)$ is supported:

$$body(r_1) \vee \dots \vee body(r_k) \vee r2(r_1) \vee \dots \vee r2(r_k) \leftarrow in_i(b). \tag{23}$$

So far $P^E_P(M)$ guesses an explanation candidate E, a pair $(R_1, R_2) \in Rg(E)$, a belief state encoded by *pres* and *abs*, and the beliefs of applicable bridge rule heads are computed. To ensure that E is an explanation it must be the case that for every pair (R_1, R_2) and belief state S some context C_i does not accept S_i given the input encoded by in_i. If some context does not accept S_i then a special atom *spoil* is derived, i.e., if the external atom $\&con\_out'_i[spoil, pres_i, in_i, out_i]()$ is false then *spoil* is derived. This atom is also derived if the guess of S and (R_1, R_2) is contradictory by itself. So for every $r \in br(M), a \in OUT_i, i \in \{1, \dots, n\}$ the following rules are in $P^E_P(M)$:

$$spoil \leftarrow not\ \&con\_out'_i[spoil, pres_i, in_i](). \tag{24}$$

$$spoil \leftarrow r1(r), nr1(r). \tag{25}$$

$$spoil \leftarrow r2(r), nr2(r). \tag{26}$$

$$spoil \leftarrow pres_i(a), abs_i(a). \tag{27}$$

We slightly extend the external atom $\&con\_out_i[pres_i, in_i]()$ for checking consistency of a context: if *spoil* is present, then the external atom must be false. This is needed, since a spoiled interpretation $I^\star$ must be a model of the HEX program, which is only guaranteed if the external atom is false in $I^\star$. So, $\&con\_out'_i[spoil, pres_i, in_i]()$ is based on $\&con\_out_i[pres_i, in_i]()$ as follows:

$$f_{\&con\_out'_i}(I, spoil, pres_i, in_i) = 0 \quad \text{iff} \quad f_{\&con\_out_i}(I, pres_i, b_i) = 0 \vee spoil \in I.$$

To saturate all guesses, we add the following rules, for all $r \in br(M), i \in ci(M), a \in OUT_i, b \in IN_i$, to $P^E_P(M)$:

$$r1(r) \leftarrow spoil. \qquad r2(r) \leftarrow spoil. \tag{28}$$

$$nr1(r) \leftarrow spoil. \qquad nr2(r) \leftarrow spoil. \tag{29}$$

$$abs_i(a) \leftarrow spoil. \qquad pres_i(a) \leftarrow spoil. \tag{30}$$

$$in_i(b) \leftarrow spoil. \qquad body(r) \leftarrow spoil. \tag{31}$$

As an interpretation I of a program P is only an answer set if it is a minimal model of fP^I, it follows that I is not an answer set if there is a model I' of fP^I with $I' \subset I$. If the guess for (R_1, R_2) and the belief state S is not acceptable at context C_i, then *spoil* is derived and saturation takes place, i.e., I' becomes $\subset$-maximal. If, however, some guess for (R_1, R_2) and S yields an equilibrium of M, then the corresponding interpretation I' is a subset of the saturated guesses, thus making the explanation candidate no minimal model of its reduct.

To obtain only valid explanations, $P^E_P(M)$ contains the following constraint:

$$\leftarrow not\ spoil. \tag{32}$$

It ensures that only saturated interpretations $I^\star$ can be answer sets. $I^\star$ is a $\subseteq$-minimal model of $fP^E_P(M)^{I^\star}$ only if no $I' \subset I^\star$ exists, i.e., if all $(R_1, R_2) \in Rg(E)$ yield an inconsistent system. For details on the saturation technique we refer to [43,70].

The answer sets of $P^E_P(M)$ now exactly encode all explanations of the inconsistent MCS M.

Theorem 5. *Let M be an inconsistent MCS. Then $(E_1, E_2) \in E^\pm(M)$ iff there exists an answer set I of $P^E_P(M)$ where $E_1 = \{r \mid e1(r) \in I\}$ and $E_2 = \{r \mid e2(r) \in I\}$.*

6.4. Implementation

We have implemented the rewritings to HEX in the MCS-IE[3] tool, the MCS Inconsistency Explainer [18], which is an experimental prototype based on the dlvhexsolver. MCS-IE solves the reasoning tasks of enumerating output-projected equilibria, diagnoses, minimal diagnoses, explanations, and minimal explanations of a given MCS.

Contexts can be realized as ASP programs, or by writing a context reasoning module using a C++ interface which allows for implementing arbitrary formalisms that can be captured by MCS contexts.

An online version of MCS-IE is available,[4] which is a useful research tool for quick analysis of inconsistency in small-scale MCS. It requires no installation of additional software on the user side and allows direct editing of bridge rules and context

[3] http://www.kr.tuwien.ac.at/research/systems/mcsie/.
[4] http://www.kr.tuwien.ac.at/research/systems/mcsie/tut/.

master.hex:
```
#context(1,"dlv_asp_context_acc", "kb1.dlv").
#context(2,"dlv_asp_context_acc", "kb2.dlv").
#context(3,"ontology_context3_acc", "").
#context(4,"dlv_asp_context_acc", "kb4.dlv").
r1: (3:pneum) :- (2:xraypneum).
r2: (3:marker) :- (2:marker).
r3: (4:need_ab) :- (3:pneum).
r4: (4:need_strong) :- (3:atyppneum).
r5: (4:allow_strong_ab) :- not (1:allergystrong).
```

kb1.dlv:
```
allergystrong.
```

kb2.dlv:
```
marker. xraypneum.
```

kb4.dlv:
```
give_strong v give_weak :- need_ab.
give_strong :- need_strong.
give_nothing :- not need_ab, not need_strong.
:- give_strong, not allow_strong_ab.
```

Fig. 4. Examples for MCS specification and knowledge base input files of the MCS-IE tool. These files encode most parts of our running example.

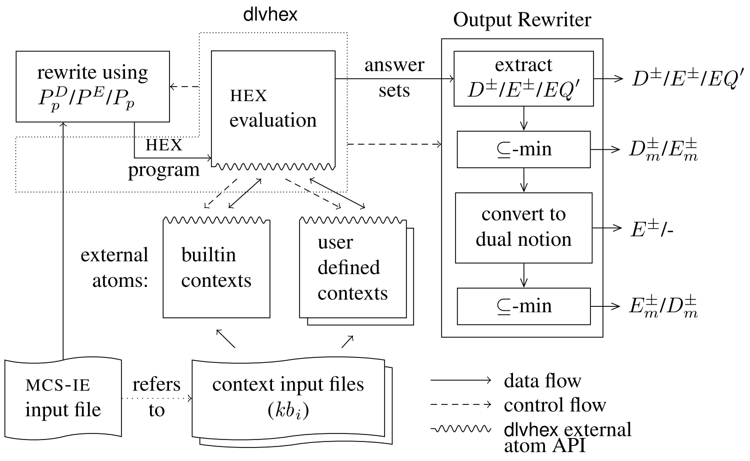


Fig. 5. Architecture of the MCS-IE system.

knowledge-bases. A list of showcase MCS allows to directly compute (minimal) diagnoses and (minimal) explanations also for MCS given in this paper.

Example 12. Fig. 4 shows files which encode our running example MCS in the MCS-IE input format. Contexts C_1, C_2, and C_4 are formalized in ASP, with knowledge bases `kb1.dlv`, `kb2.dlv`, and `kb4.dlv`, these contexts are evaluated through a HEX-plugin for external atoms, which in turn uses the dlv solver. On the other hand, ontology reasoning C_3 is implemented in C++. For more details about the format and the interface we refer to [18]. □

Fig. 5 shows the architecture of the MCS-IE system, which is implemented as a plugin to the dlvhexsolver. The MCS M at hand is described by the user in a master input file, which specifies all bridge rules and contexts (it may refer to context knowledge base files). Depending on the configuration of MCS-IE, the desired reasoning tasks are solved using one of the three rewritings $P_p(M)$, $P_p^D(M)$, resp. $P^E(M)$, on the input MCS M. MCS-IE enumerates answer sets of the rewritten program, and potentially uses a $\subseteq$-minimization module, and a module which realizes the conversions between diagnosis and explanation notions as described in Theorem 2 and Corollary 1. Explanations can be computed by MCS-IE using the direct encoding given in Section 6.3 or through the conversion from diagnoses.

As expected, MCS-IE shows the following behavior w.r.t. efficiency: the rewriting $P_p^D(M)$, which uses guess-and-check, shows better performance than the rewriting $P^E(M)$, which expresses the **coNP** task of recognizing explanations in the $\mathbf{\Sigma_2^P}$ formalism of full-fledged disjunctive HEX programs.

Nevertheless, it appeared that also $P_p^D(M)$ does not scale well. This has led to the development of a better HEX evaluation framework, which divides and conquers the guessing space more efficiently [33]. While the old evaluation of $P_p^D(M)$ scales exponentially in the total number of output beliefs and bridge rules, the improved one scales exponentially only in the number of output beliefs and bridge rules of the largest context of M. For a thorough experimental evaluation of the above encodings using different version of the dlvhex solver, we refer to [33, Section 5] and [34, Section 6.1].

Other approaches to compute diagnoses and explanations of MCS may be faster than the HEX rewriting approach, e.g., distributed evaluation with an extended version of the DMCS algorithm [4]. However, the primary focus of this work are the notions of diagnosis and explanation, investigation of their properties, and an experimental framework for evaluation; therefore the efficient (and more intricate) evaluation methods are left for future work.

7. Related work

Non-monotonicity in MCS was introduced in [85] and then further developed in [21,24] to eventually allow heterogeneous as well as nonmonotonic systems, and in particular nonmonotonic MCS [21] as considered in this article (cf. [22] for a more comprehensive account of work related to MCS). However, issues arising from inconsistency of such systems have been largely disregarded.

7.1. Inconsistency in MCS

A remarkable exception, and thus most closely related to ours (see also [37]), is [15], where inconsistency in a homogeneous MCS setting is addressed. The approach is to consider defeasible bridge rules for inconsistency removal, i.e., a rule is applicable only if its conclusion does not cause inconsistency. This concept is described in terms of an argumentation semantics in [14]. The decision which bridge rules to ignore is based, for every context, on a *strict total order* of all contexts. The set of rules that are ignored thus corresponds to a unique deletion-only diagnosis whose declarative description is more involved compared to our notion, but which is polynomially computable. Note however, that the second component of diagnoses, i.e., rules that are forced to be applicable, have no counterpart in the defeasible MCS inconsistency management approach. Furthermore, the strict total order over contexts forces the user to make (perhaps unwanted) decisions at design time; alternative orders would require a redesign and separate evaluation. Our approach avoids this and can be refined to respect various kinds of orderings and preferences.

Another formalism for homogenous contextualized reasoning that incorporates a form of inconsistency tolerance is the Contextualized Knowledge Repository (CKR) approach [87]. It is similar to the MCS approach of formalizing context-dependent knowledge, i.e., a CKR is a set of contexts. Contexts are based on description logic and a hierarchical coverage relation is used to specify that the knowledge of one context, regarding specified topics, is broader than the knowledge of another context. A CKR model then is a collection containing a local DL-model for each context such that constants, concepts and roles that are covered are interpreted exactly the same way in both contexts. The coverage relation itself is specified using a DL-like meta-language.

Consider a DL assertion $P(a)$: if context C_1 covers context C_2, then the concept P is interpreted in C_1 and in C_2 in the same way, as well as the individual a. MCS are different since the interpretation of $P(a)$ in C_1 is not related to that in C_2. Similar as in bridge rules of MCS, a CKR context can refer to knowledge from other contexts using a so-called qualifier, e.g., $P(a)_{\{location=\mathit{Italy},time=2010\}}$ refers to $P(a)$ of a context that covers knowledge about *Italy* in the year 2010.

A CKR is inconsistency tolerant in the sense that if some context is inconsistent (i.e., its local model is the one with empty domain), then this inconsistency does not propagate to other unrelated contexts. The same property also holds for MCS, but in contrast to CKR, our approach allows to restore consistency by modifying the interlinking of contexts.

Similar in vein to CKR systems are *Modular Ontologies*, i.e., a framework where consistent description logic modules utilize and realize a set of interfaces [44]. These interfaces are connected by bridge rules for Distributed Description Logic (DDL) [19]. Consistent query answering in a module is achieved by using the maximum consistent set of interfaces utilized by this module only, therefore whole interfaces will be ignored if they would cause any inconsistency in the module. Again, in addition to addressing a more general setting in terms of heterogeneity, our work considers potential modifications of bridge rules that allow to go beyond simple masking of inconsistent parts of the system in order to analyze inconsistency and potentially restore consistency.

Conceptually close to the above homogeneous forms of MCS are *Federated Databases*, a distributed formalism for linked databases [59]: objects can be *exported* and *imported* using a decentralized negotiation between two databases. Notably, [89] is a survey that, in addition to autonomy (access granting and revoking), is taking up on issues of heterogeneity, however mostly referring to the integration of different query languages. Existing approaches handle incoherence in a database-typical manner of *cascading* or *rejecting* local or distributed constraints. For instance, several protocols for global integrity constraint enforcement are presented in [57]. These protocols define the *quiescent* state of the system—when it is *at rest*—and ensure that no constraints are violated in such states. Hence, inconsistency in federated databases is addressed at the level of the (individual) databases rather than their interlinking. Even though resorting to SQL and stratified Datalog allows for non-monotonicity, the possibility of instability in a distributed database system—due to a cyclic dependencies—has not been addressed in the literature. Our work would be suitable to deal with such situations, given that federated databases can be described as MCS with stratified (mostly monotonic) contexts including constraints, and with positive bridge rules.

Concerning the complexity results we established for diagnoses of MCS, we remark that they are related to respective results in abduction: by associating abducible hypotheses with bridge rules, due to the non-monotonicity of the system, recognition of diagnoses corresponds to *cancellation abduction problems*. The latter have been shown to be **NP**-complete in [25] under the assumption of a tractable underlying theory (i.e., for **P** contexts in our terminology).

7.2. Broader context

In a broader context, we have explored the relationship of our work to approaches and methods for inconsistency management in knowledge bases, grouped into debugging techniques (e.g., for Prolog [80,81] and ASP [51,74]), repairing methods (for instance based on abductive reasoning [62], discrimination among fusion rules [61], or policies for subquery propagation in peer-to-peer systems [8]), consistent query answering (e.g., over ontologies [65], propositional knowledge bases in peer-to-peer systems [16], etc.), and paraconsistent reasoning (applying, e.g., syntactic [12], logic-based [86], or domain-specific [46] methods). We refer to Appendix B resp. [38] for more details. Different from most approaches our primary aim is to provide a solid theoretical framework for analyzing inconsistency; we do not aim at automatically restoring consistency (but our notions can be used to achieve that).

8. Conclusion

We have considered the problem of inconsistency analysis in nonmonotonic Multi-Context Systems (MCS), which are a flexible, abstract formalism to interlink heterogeneous knowledge sources for information exchange. We have presented a consistency-based and an entailment-based notion of inconsistency explanation, called diagnosis and explanation, which are in a duality relation that can be exploited for computational purposes, and which enjoy modularity properties. We have characterized the computational complexity of the two notions, establishing generic results for a range of context complexities. They show that in many cases, explaining inconsistency does not lead to a jump in complexity compared to inconsistency testing, although (unsurprisingly) depending on the interlinking intractability might arise. We have furthermore shown how the notions can be computed by a transformation to HEX programs, which has been implemented in the experimental software tool MCS-IE.

Our results provide a basis for building advanced systems of interlinked knowledge sources, in which the natural need for inconsistency management is supported, by taking specifically the information linkage as a source of inconsistency into account, in contrast to traditional works on inconsistency management that focus on the contents of the knowledge sources; however, in loosely connected systems, control over autonomous knowledge sources is elusive and modifying the information exchange may be the only resort to remove inconsistency.

Further work The work presented in this article has been continued in several directions. One of them is to impose different kinds of preferences on the notions of diagnosis and explanation that were introduced here, as in [39,91]. They allow for filtering and comparing diagnoses; using meta-programming techniques, the most-preferred ones can be selected from all diagnoses.

Another direction concerns incomplete information about contexts. The setting considered in this article assumes complete information about the behavior of the contexts in information exchange, i.e., for each ‘input’ of relevant beliefs from other contexts accessed via bridge rules, the ‘output’ in terms of firing bridge rules is fully known. In real-world applications, however, this information may be only available for specific (classes of) inputs, and querying a context arbitrarily often to gain this knowledge might be infeasible. In such scenarios the notions introduced in [36] allow to obtain reasonable approximations for diagnoses and explanations of inconsistency.

Finally, another implementation is currently underway in which diagnoses and explanations can be computed by distributed algorithms, exploiting the distributed MCS evaluation framework of [3,4,30].

Open issues Several issues remain for future work. Building on the notions of preferred diagnosis and explanation, a further topic is to establish concrete inconsistency management procedures for analysis. To this end, a system administrator might ask repeatedly for diagnoses and explanations, considering subsystems and/or a modified interlinkage, and select among the ones presented a most appealing one; the information about past selections may in turn be used to adjust the preferences for calculations.

On the computational side, scalability to scenarios with larger data volume and number of bridge rules is desirable, where the intrinsic complexity of our diagnoses and explanations is prohibitive in general. It remains to single out settings where scalability is still possible, and to get a clearer picture of the scalability frontier. This is linked to the complexity of consistency checking for an MCS; restrictions on the interlinking, in numbers and structure (for the latter, see [4]) will be helpful, as well as properties of the context logics (e.g., monotonicity and unique accepted belief sets). Related to this is developing pragmatic variants of our notions, like focusing by protecting bridge rules (which does not increase worst case complexity), giving up properties (e.g., minimality), or by tolerating inconsistency in parts of the system.

Another issue is to combine inconsistency management of contents and of context interlinking. Recall that B.2 points out how maximal consistent subsets of a knowledge base (which are ubiquitous in content-based inconsistency management) might be simulated using bridge rules. However, an emerging combination—although in a uniform formalism—would be inflexible and less amenable to refinement. More promising is to combine the notions in this article and in [23], which generalized MCS with a management component for each context and operations to be performed on the knowledge base when a bridge rule fires; this allows for a more sophisticated content-change than simple addition of formulas. Nevertheless, consistency cannot be guaranteed in general with such content-based approaches, as inconsistency caused by cyclic information flow cannot be resolved. Since the latter can be dealt with by modifying the interlinking, as for instance by our notion of diagnosis, a combination of techniques can be successful.

Acknowledgements

We thank the reviewers of this paper and of the preliminary KR 2010 version for their constructive comments to improve our work.

This work was supported by the Vienna Science and Technology Fund (WWTF) under grant ICT08-020.

Appendix A. Examples

In this section, we give the abstract logics $L_{\mathcal{A}}$ and L_{Σ}^{asp} in detail.

Example 13. We formally introduce the abstract logic $L_{\mathcal{A}}$ to capture ontologies and description logic. Over a signature of atomic concepts $\mathbb{C}$, roles $\mathbb{R}$, and individuals I, T-Box axioms and A-Box axioms are defined based on the notion of concepts. Concepts are inductively defined as follows: every atomic concept is a concept, and if C, D are concepts and $R \in \mathbb{R}$ is a role, then $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, and $\exists R.C$ are concepts. Given concepts C, D, a role $R \in \mathbb{R}$, and individuals $a, b \in I$, a T-Box axiom (terminological axiom) is a formula of the form $C \sqsubseteq D$, and an A-Box axiom (assertional axiom) is either of the form $a{:}C$, or of the form $(a, b){:}R$. Finally, $\mathcal{ALC}$ axioms are either T-Box axioms or A-Box axioms.

Then, $L_{\mathcal{A}}$ is composed of

- **KB**, being the collection of sets of $\mathcal{ALC}$ axioms,
- **BS**, being the set of possibly believed assertions, i.e., **BS** is the powerset of the set of atomic A-Box axioms, and
- **ACC**, being a mapping from knowledge bases to the set of assertions entailed by the knowledge base. For our purpose, $\mathbf{ACC}(kb) = \{S\}$ where S is the set of atomic A-Box axioms entailed by kb (see [2] for details). $\square$

Example 14. We give the formal definition of L_{Σ}^{asp}, the abstract logic for disjunctive logic programs under the answer-set semantics over a non-ground signature Σ. For $L_{\Sigma}^{asp} = (\mathbf{KB}, \mathbf{BS}, \mathbf{ACC})$,

- **KB** is the set of normal disjunctive logic programs over Σ, i.e., each $kb \in \mathbf{KB}$ is a set of rules of the form

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_i, \text{ not } b_{i+1}, \ldots, \text{ not } b_m,$$

  where all a_i, b_j, are atoms over a first-order language Σ, and $n + m > 0$. Let r be a rule of the aforementioned form, then $H(r) = \{a_1, \ldots, a_n\}$, $B^+(r) = \{b_1, \ldots, b_i\}$, $B^- = \{b_{i+1}, \ldots, b_m\}$, and $B(r) = B^+(r) \cup B^-(r)$. Each rule $r \in kb$ must be safe, i.e., $vars(H(r)) \cup vars(B^-(r)) \subseteq vars(B^+(r))$, where for a set of atoms A, $vars(A) = \{vars(a) \mid a \in A\}$ and $vars(a)$ is the set of first-order variables occurring in the atom a,
- **BS** is the set of Herbrand interpretations over Σ, i.e, each $bs \in \mathbf{BS}$ is a set of ground atoms from Σ, and
- $\mathbf{ACC}(kb)$ returns the set of kb's answer sets: for $P \in \mathbf{KB}$ and $T \in \mathbf{BS}$ let $P^T = \{r \in grnd(P) \mid T \models B(r)\}$ be the FLP-reduct of P w.r.t. T, where $grnd(P)$ returns the ground version of all rules in P. Then $bs \in \mathbf{BS}$ is an answer set, i.e., $bs \in \mathbf{ACC}(kb)$, iff bs is a minimal model of kb^{bs}. $\square$

Appendix B. Related work in broader context

For putting our work in a broader context, we subsequently relate it more generally to work on inconsistency management in knowledge bases. We classify and discuss some of the most relevant literature according to the following basic approaches:

- *debugging* techniques serve the purpose of diagnosing information systems, aiming at identifying sources of unexpected and in most cases unintended computation outcomes, and at explaining the latter;
- *repairing* techniques modify the content of knowledge bases in order restore consistency, in particular when new information is incorporated into a knowledge base, or when several knowledge bases are integrated into a single one;
- *consistent query answering* virtually repairs a knowledge base or system, often by ignoring a minimal subset of beliefs or subsystems, and operates on the resulting (virtual) consistent system (i.e., no knowledge is permanently removed);
- *paraconsistent reasoning* accepts contradictory knowledge and, rather than repairing or ignoring (parts of) the information, a more tolerant mode of reasoning is applied that handles also inconsistent pieces of knowledge in a non-trivial way.

As already mentioned above, different from most approaches to inconsistency management our main goal is a solid analytic framework for inconsistency rather then automatic consistency restoring.

B.1. Debugging in logic programming

Debugging in logic programming, i.e., finding out why some logic program has no or an unexpected answer, is remotely related to the problem considered in this paper given that bridge rules look similar to rules of logic programming. A major difference is that in MCS we take contexts with an opaque content into account. In logic programming, presence of an atom in a model of a program directly depends on the firing of rules, which in turn directly depends on the presence or absence of other atoms in the bodies; in the MCS framework, which allows to capture arbitrary logics by abstract belief set functions, there is in general no visible link between the firing of bridge rules and beliefs accepted by a context.

B.1.1. Prolog debugging

A framework for debugging Prolog programs was developed in [88]. It relies strongly on the operational specifics of Prolog and consists of a diagnosis and a bug-correction component, where three basic types of errors are considered: (i) termination with incorrect output, (ii) termination with missing output, and (iii) nontermination. For the latter, the approach identifies rules that behave unexpectedly by tracing procedure calls and querying the user whether the procedure call at hand of the form $\langle \mathit{procedure}, \mathit{input}, \mathit{output}\rangle$ is correct. A similar goal is achieved in [78], where the user should not tell whether such a triple is wrong, but point to a wrong subterm of a procedure call; for that, the implementation builds on a modified unification algorithm that keeps track of the origins of subterms. This is further refined in [79], where the different types of bugs are treated uniformly and by the use of a heuristics the number of questions to the user is reduced.

In comparison, our notion of inconsistency diagnosis roughly corresponds to type (i) and (ii) errors: in a diagnosis (D_1, D_2), D_1 contains bridge rules whose head belief is “incorrect”, while D_2 contains bridge rules whose head belief is “missing”. As for (iii), nontermination is not an issue for MCS since no infinite recursion can emerge (modulo computations inside contexts). Furthermore, our approach is fully declarative, without operational attachment adherent to Prolog, and it does not require user input; on the other hand, it only covers consistency and no further aspects. Nonetheless, it is possible to mimic behavior under user input to some extent by using a technique similar to the meta-reasoning transformation in [39].

A purely declarative perspective on Prolog debugging is taken in [72], based on the formal semantics of extended programs under SLDNF resolution. Again two types of errors are considered, so called “wrong clause instances” (wrong solutions) and “uncovered atoms” (missing solutions). To pinpoint the origin of such errors, the user must specify the intended interpretation of the program, by repeatedly answering queries about the behavior of the rules.

In [81] a connection between logic program debugging and abductive diagnosis is investigated. It considers extended logic programs (with strong and default negation) under closed-world assumption (CWA). Based on revisables, i.e., a subset R of the set of literals $\mathit{not}\,L$ assumed true by CWA, and the notion of supported sets $SS(L)$ of a literal L, the removal sets of L are defined as the hitting sets of $SS(L)$ restricted to R; the ones of the literal $\perp$ indicate how to obtain a non-contradictory program. Using a transformed program P_1 of P and information about wrong and missing solutions in P, so called minimal revising assumptions (MRAs) of P_1 are computed in an iterative manner which identify the reasons for wrong and missing solutions. For programs P that model diagnostic problems, minimal solutions can be obtained from the MRAs.

The ideas and notions in [80,81] are merged in [72,80] for normal logic programs with constraint rules under well-founded semantics. Referring to them, a diagnosis for a set U of literals is a pair $D = \langle \mathit{Unc}, \mathit{InR}\rangle$ where Unc are uncovered atoms and InR are incorrect rules of P, such that U is contained in the well-founded model (WFM) of the program P' that results from P by removing all incorrect rules and adding all uncovered atoms. In case of a single minimal diagnosis, the bug in the program is pinpointed precisely; otherwise, the user is asked which diagnosis corresponds to the intended interpretation. This leads to an iterative debugging algorithm that only asks disambiguating queries, i.e., it asks about a subset of the intended interpretation and adds the answer to U. Our notion of inconsistency diagnosis, where $D = (D_1, D_2)$ is a diagnosis iff $M[br(M) \setminus D_1 \cup cf(D_2)] \not\models \perp$ resembles this notion for $U = \emptyset$; the underlying semantics of MCS is however very different from WMF. Furthermore, there is no counterpart of our inconsistency explanations, nor have refined diagnoses been considered.

B.1.2. ASP debugging

Answer-set Programming (ASP) is as a rule-based paradigm related to MCS, yet more under grounded equilibrium semantics, which imposes a minimality condition on equilibria [21]; in fact, answer-set programs can be modeled as particular MCS with monotonic rules and non-monotonic bridge rules.

The declarative debugging of answer-set programs was approached by [90] for programs that have no cycles of odd length; in subsequent works, tagging [20], meta-programming for ground [51] and non-ground programs [74], and establishing procedural techniques (breakpoints, step-wise execution) [75] have been considered. The idea is that an expected answer-set E and an (erroneous) ASP program P are transformed into a program T whose answer-sets explain why E is not an answer-set of P. Explanations cover that an instantiation of some rule in P is not satisfied by E, as well as the presence of unfounded loops (i.e., lack of foundedness). The latter could be of interest for developing diagnosis of MCS under grounded equilibria semantics; this remains for future work. On the other hand, the procedural techniques seem to be less promising, as MCS lack rule chaining at the abstract level.

A different approach to debug answer-set programs is given in [5], where A-Prolog (an ASP-based language) is extended by *consistency-restoring* (CR) rules of the form

$$r: \quad h_1 \text{ or } \ldots \text{ or } h_k \overset{+}{\leftarrow} l_1, \ldots, l_m,\ \text{not}\, l_{m+1}, \ldots,\ \text{not}\, l_n.$$

which intuitively reads as: if $l_1, \ldots, l_m$ are accepted beliefs while $l_{m+1}, \ldots, l_n$ are not, then one of $h_1, \ldots, h_k$ “may possibly” be believed. In addition, a preference relation on the rules may be provided. The semantics of CR rules is defined via a translation to abductive logic programs, i.e., logic programs where certain atoms are abducibles (cf. [63]). In answer sets of such programs, a minimal set of abducibles may be assumed to be true without further justification.

Disregarding possible rule preferences, a logic program P with CR rules CR can be embedded to an MCS $M=(C_1)$, where the single context C_1 is over disjunctive logic programs, such that the answer sets of P with CR correspond to the witnessing equilibria of the minimal diagnoses (D_1, D_2) of M. In more detail, C_1 has the knowledge base $kb_1 = P \cup \{cr(r) \mid r \in CR\}$ and bridge rules $br_1 = \{(c_1 : ab(r)) \leftarrow a_\perp. \mid r \in CR\}$, where $a_\perp$ and $ab(r)$ are fresh atoms, for each r as above, and

$$cr(r) = h_1 \vee \cdots \vee h_k \leftarrow ab(r), l_1, \ldots, l_m,\ \text{not}\, l_{m+1}, \ldots,\ \text{not}\, l_n;$$

informally, unconditional firing of a bridge rule simulates the corresponding CR rule; note that $D_1 = \emptyset$.

B.2. Content-based methods

The methods and approaches underlying research issues and works presented in this subsection exhibit more foundational differences to our approach. Therefore, we will mostly discuss them on a more general level, pointing to some seminal works and survey articles for more extensive coverage of the relevant literature.

B.2.1. Repair approaches in integrating information

A lot of work on inconsistency management has been concentrating on the repair of data when merging, incorporating, or integrating data from different sources. In contrast to our work, in such approaches usually the mappings that relate data of different knowledge bases are fixed, while the contents of the knowledge bases are subject to change in order to restore consistency. This subsumes approaches that do not actually modify original data but modify it virtually (i.e., a view), or operate on a copy.

Belief revision and *belief merging* are well understood problems, in particular for classical propositional theories [64,77]. They address how to incorporate a new belief into an existing knowledge base, respectively how to combine knowledge bases, such that the resulting knowledge base is consistent. In this regard, our approach is more related to belief merging than to belief revision. A major difference to belief merging is, however, that MCS connect heterogeneous knowledge bases in a decentralized fashion (compared to a centralized merge of uniform knowledge bases), and that selective information exchange among knowledge bases is possible via bridge rules in complex topologies. Furthermore, our work concentrates on changing the mappings between these components in case of conflict, while belief merging strives for modified contents (i.e., knowledge base).

Abductive reasoning is often applied to identify pieces of information that need to be changed in order to repair a logical theory or knowledge base, cf. [62,73,92]. In particular, in [62] abduction is applied to repair theories in (nonmonotonic) logic based on notions of ‘explanation’ and ‘anti-explanation’. Given a theory K and a set Γ of abducible formulas, they remove the formulas of a set $O \subseteq \Gamma$, and add the formulas of a set $I \subseteq \Gamma$, to entail (resp. not entail) an observation F; i.e., $(K \cup I) \setminus O \models F$ (explanation), resp. $(K \cup I) \setminus O \not\models F$ (anti-explanation). A *repair* of an inconsistent theory K is given by an anti-explanation of $F = \perp$; in particular, if $\Gamma = K$ and $I = \emptyset$, then such a repair is a *maximal consistent subset* of K; the use of such sets to restore consistency is central to many approaches of belief revision. Our notion of diagnosis may be regarded as a generalized 2-sorted variant of such anti-explanations, where $O \subseteq \Gamma_O$ and $I \subseteq \Gamma_I$; moreover, under suitable conditions, it is reducible to ordinary anti-explanations. In particular, for $\Gamma_I = \emptyset$ and $\Gamma_O = K$, the maximal consistent subsets of K correspond to the minimal diagnoses of M_K. Indeed, for the MCS $M = (C_1)$ with single context C_1 having knowledge base $kb_1 = K$ and bridge rules $br_1 = br_\Gamma^\top \cup br_\Gamma^\perp$, where $br_\Gamma^\top = \{(c_1 : \phi) \leftarrow (c_1 : \top) \mid \phi \in \Gamma\}$ and $br_\Gamma^\perp = \{(c_1 : \phi) \leftarrow (c_1 : \perp) \mid \phi \in \Gamma\}$, then the minimal diagnoses of M correspond to the repairs of K. One may replace Γ with Γ_O in both, $br_\Gamma^\top$ and $br_\Gamma^\perp$; furthermore, modified bodies in $br_\Gamma^\top$ allow for *conditional removal* of formulas, with conditions that might be beyond the expressiveness of the language of K. As regards explanations, our notion of explanation has no counterpart in the approach of [62].

Information integration approaches (see, e.g., [31,61,67,68]) wrap several information sources and materialize the information into one schema. Differences exist in whether the global schema is expressed as a view in terms of the local schemata (global-as-view approaches), or vice versa (local-as-view). The relevant relationships are represented as mappings, which often are specified by database queries; inconsistencies are resolved by modifying the materialized information, thus again by changing contents. However, since the original information sources are not altered, one might consider it closer in spirit to our approach than belief merging. The information *fusion* approach described in [61] uses fusion rules that look similar to bridge rules and handle inconsistency using preferences, voting, and generalizations of conflicting values (e.g., ‘truck’ and ‘van’ could generalize to ‘car’). Different from our approach in MCS, fusion rules anticipate potential inconsistencies instead of analyzing it, moreover they assemble a *single target knowledge base* without cycles in the system such that achieving consistency does not require a global view of the system. Inconsistency management in information integration systems, and in particular the global-as-view approach, may be regarded as implicit change of mappings, by discarding tuples and/or generating missing tuples. Naturally, this corresponds to deactivating bridge rules and forcing bridge rules to fire, respectively. Different from MCS however, information integration approaches rely on hierarchical, acyclic system topologies and monotone semantics (that can be evaluated using fixpoint algorithms). On the other hand, they apply a more expressive mapping formalism compared to bridge rules in MCS.

Peer-to-peer data integration systems [27] allow for a dynamically changing architecture of a data integration scenario in which peers can enter or leave the system anytime. Inconsistency handling in such systems resorts mainly to approaches that are motivated or akin to techniques of consistent query answering; we thus postpone this to the respective subsection below. The approach in [8] allows cyclic queries and topologies, however "[the peer-to-peer querying layer] also needs a policy on how far to propagate subqueries transitively through chains of P2P connections, which can be arbitrarily long and cyclic". Opposed to that approach, MCS resolve cyclic dependencies using equilibrium semantics, which is a more clean and predictable approach, as MCS do not require cycle breaking at some predefined nesting depth.

Ontology mapping [28] and the related tasks of ontology alignment, merging, and integration aim at reusing ontologies in a suitable combination. To this end, mappings between concepts, roles, and individuals are identified to denote the same entity in different ontologies, usually by automatic, statistical methods to 'discover' mappings. They may introduce inconsistency in the (global or local) view on the resulting ontology, even if each individual ontology is consistent. Consistency is achieved by disregarding a mapping if it would add an inconsistency. Heterogeneity in ontology mapping usually refers to different nomenclatures prevailing in different ontologies, or to ontologies in different yet closely related formalisms (e.g., different description logics). In contrast, in MCS heterogeneity refers to combining systems based on different logical formalisms, which need not share any relationship. More notably, however, our work aims at explaining inconsistency, and to provide via diagnoses a more fine-grained possibility to achieve consistency than by simply discarding mappings.

To summarize, the main difference between our work and the contributions to these rather diverse settings of integrating information—and in particular the issue of achieving integrity in doing so—is that we consider modifying the 'mapping', i.e., the interlinking, rather than the data. While the importance of maintaining and repairing mappings has been recognized [31], major breakthroughs are still missing.

B.2.2. Consistent query answering

The approaches considered in this section do not actually modify data to repair an inconsistent system, but virtually consider possible repairs in order to return consistent answers to queries. As this includes (partial) ignorance of information (and thus inconsistency) for the sake of reasoning on a consistent system, the approaches may be regarded as in between repairing and paraconsistent reasoning.

The term *consistent query answering (CQA)* has been coined in the database area where various settings (w.r.t. integrity constraints and operations for repair) have been considered [1,9,10]. For instance, in the case of denial constraints (including key constraints, functional dependencies, etc.), it is sufficient to restrict the attention to tuple deletions for obtaining repairs and answering queries consistently. Thus, CQA might be regarded as an approach that automatically applies deletion-diagnoses to suppress inconsistent information for answering queries over inconsistent relational databases. Despite this superficial similarity to our work, the differences are apart from heterogeneity much more fundamental: diagnoses and explanations address the interlinking of knowledge bases rather than their content and they aim at making inconsistencies amenable to analysis, explicitly hinting at problems that should be investigated, rather than treating them implicitly for the sake of providing consistent answers. CQA techniques have also been extended to description logic ontologies, e.g., in [65,66], where the taxonomy part is considered to be correct but the data part as possibly inconsistent. Consistent answers to queries are then obtained on maximal consistent subsets of the data w.r.t. the taxonomy part (and potential further constraints).

Other approaches (but similar in nature) have been applied to answering queries in peer-to-peer data integration settings. An automatic approach for repair was presented in [26] that ignores inconsistent components, resp. the beliefs held by a minority of peers in the system. Another work on peer-to-peer systems over propositional knowledge bases [16] answers queries over a maximal consistent subset of the knowledge bases. Besides the conceptual difference to MCS regarding the system architecture (dynamic vs. static), our approach explains inconsistency by pointing out mappings that must be changed to achieve consistency. Furthermore, its does not aim at automatic fixes to the system, and in particular not by ignoring entire contexts or beliefs held by a minority among them.

B.2.3. Paraconsistent approaches

Paraconsistent reasoning approaches (see, e.g., [11,60]) aim upfront at ignoring or tolerating inconsistency in knowledge bases, providing means to reason on them without knowledge explosion, i.e., without justifying arbitrary beliefs (*ex falso quodlibet*); thus, they do not focus on eliminating inconsistency. Nevertheless, in addition to keeping information systems operable in case of inconsistency, paraconsistent reasoning may, similar to our aim, also serve the purpose of analyzing inconsistency.

Taking again a very general perspective, in particular disregarding heterogeneity and even the fact that our techniques apply to the interlinking of information, syntactic approaches such as [12] would be closest to our approach. They essentially restrict theories to the intersection of maximal consistent subsets of formulae as a basis for drawing paraconsistent conclusions. However, while minimal deletion-diagnoses might be viewed as corresponding to maximal consistent subsets, our approach does not prescribe a particular reasoning mode upon them (like considering the system obtained by their intersection). Moreover, our notions of diagnosis and explanation provide more fine-grained structures for analysis than just considering deletion diagnosis, and they deal with nonmonotonic behavior.

The methods that are applied in logic-based approaches to paraconsistent reasoning are completely orthogonal to our techniques. The most prominent representatives resort to *many-valued logics* in order to deal with inconsistency (cf. [6,82]).

The same applies to *paraconsistent logic programming* [17] (see e.g. [35] for more references and recent works), which therefore also elude themselves from a detailed comparison. Nevertheless, developing model-based techniques for paraconsistent reasoning from inconsistent MCS is an interesting topic for future research. In this regard, [86] can be inspiring, where trust on information sources on the web has been modeled using an extension of Belnap's four-valued logic [6] and bridge-rule like constructions based on external predicates govern the information flow.

In the area of *process modeling* and *requirements engineering*, [46] describes an approach of inconsistency management where inconsistencies are considered an important part of the system and not necessarily removed. In this approach, so-called ViewPoints [47] describe stakeholders in an engineering process which use different formalisms and representations, where coordination is performed using inter-ViewPoint coordination rules. The inconsistency handling approach in [46] uses temporal action logic to specify reactions to detected inconsistencies. Similar to inconsistency explanations in MCS, this inconsistency handling is based on the subset of knowledge that evoked the inconsistency. MCS describe distributed knowledge based systems and inconsistency is absence of a global equilibrium; different from that, ViewPoints describe distributed processes with synchronization points and inconsistency is defined as inconsistency during an inter-ViewPoint coordination effort.

We conclude this section with a pointer to Gabbay and Hunter [48] who argued strongly for *managing inconsistency*, in contrast to avoiding, removing, or ignoring it. Their point is that an inconsistent system requires actions to be taken, and in order to do so, different issues need to be taken into account that require a variety of methods. Notably they also developed a corresponding framework in a relational database setting [49,50]. In this spirit, we consider the notions of diagnosis and inconsistency explanation for MCS as providing a foundational basis for developing methods for more specific tasks on top in order to manage inconsistency of the system.

Appendix C. Proofs

Proof of Proposition 2. ($\Rightarrow$) Let $D_r = (D_1, D_2, fg) \in D^{\pm,r}_m(M)$, we have to prove that $(D_1, D_2) \in D^{\pm}_m(M)$.

We first show that $(D_1, D_2) \in D^{\pm}(M)$. Since D_r is a refined diagnosis, it holds that $M[br(M) \setminus D_1 \cup fg(D_2)] \not\models \bot$. Let S_w be a witnessing equilibrium of $M[br(M) \setminus D_1 \cup fg(D_2)]$, then it holds for every $r \in D_2$ that $S_w |\!\!\leadsto fg(r)$ since D_r is minimal. Therefore, S_w is an equilibrium of $M[br(M) \setminus D_1 \cup cf(D_2)]$, hence $(D_1, D_2) \in D^{\pm}(M)$. Since D_r is minimal, it follows that $S_w |\!\!\leadsto head(r) \leftarrow Body(fg(r)) \cup B.$, where $Body(fg(r)) \subset B \subseteq Body(r)$, does not hold for any $r \in D_2$, hence $refine(D_2, S_w) = fg$.

It remains to show that $(D_1, D_2) \in D^{\pm}_m(M)$. Assume for contradiction that there exists $(D'_1, D'_2) \subset (D_1, D_2)$ such that $(D'_1, D'_2) \in D^{\pm}_m(M)$. Let S'_w be a witnessing equilibrium of (D'_1, D'_2) and $fg' = refine(D'_2, S'_w)$, then it holds that $(D'_1, D'_2, fg') \in D^{\pm,r}(M)$ since S'_w is a witnessing equilibrium of $M[br(M) \setminus D'_1 \cup fg'(D'_2)]$. Since $(D'_1, D'_2, fg') \in D^{\pm,r}(M)$ and $(D'_1, D'_2) \subset (D_1, D_2)$ it holds that D_r is not a minimal refined diagnosis, which is a contradiction. Therefore, no such (D'_1, D'_2) exists and $(D_1, D_2) \in D^{\pm}_m(M)$.

($\Leftarrow$) Let $D = (D_1, D_2) \in D^{\pm}_m(M)$, let S_w be a witnessing equilibrium of D, and let $refine(D_2, S_w) = fg$. Furthermore, it holds that no witnessing equilibrium S'_w exists with $refine(D_2, S'_w) = fg'$ and $fg' < fg$. We show that (D_1, D_2, fg) is a minimal refined diagnosis of M. By definition of *refine* it holds for every $r \in D_2$ that $S_w |\!\!\leadsto fg(r)$. Furthermore, there is no body-reduction function fg' such that it holds for every $r \in D_2$ that $S_w |\!\!\leadsto fg'(r)$. Therefore, S_w is an equilibrium of $M[br(M) \setminus D_1 \cup fg(D_2)]$ and $(D_1, D_2, fg) \in D^{\pm,r}(M)$.

Towards contradiction assume that (D_1, D_2, fg) is not minimal, then there exists $(D'_1, D'_2, fg') \in D^{\pm,r}_m(M)$ such that $(D'_1, D'_2) \subset (D_1, D_2)$ or $D_1 = D'_1$, $D_2 = D'_2$, and $fg' < fg$. In the former case, there exists a witnessing equilibrium S'_w of $M[br(M) \setminus D'_1 \cup fg'(D'_2)]$. Therefore S'_w is a witnessing equilibrium of $M[br(M) \setminus D'_1 \cup cf(\{r \in D'_2 \mid S'_w |\!\!\leadsto fg'(r)\})]$, i.e., $D'' = (D'_1, \{r \in D'_2 \mid S'_w |\!\!\leadsto fg'(r)\}) \in D^{\pm}(M)$. Since $D'' \subset D$ this is a contradiction to $D \in D^{\pm}_m(M)$. In the latter case holds $fg' < fg$ and there exists a witnessing equilibrium S'_w of $M[br(M) \setminus D_1 \cup fg'(D_2)]$. Since $(D'_1, D'_2, fg') \in D^{\pm,r}_m(M)$ and $D'_1 = D_1, D'_2 = D_2$, it holds that S'_w also is an equilibrium of $M[br(M) \setminus D_1 \cup cf(D_2)]$ and $refine(D_2, S'_w) = fg'$. Then, $fg' < fg$ directly contradicts that our assumption that no such S'_w and fg' exist. Since all cases are contradicting, it must hold that (D_1, D_2, fg) is a minimal refined diagnosis. □

Proof of Proposition 3. ($\Rightarrow$) Let $(E_1, E_2) \in E^{\pm}(M)$, pick fg such that for every $r \in E_2$ holds $fg(r) = head(r) \leftarrow .$, i.e. $\{fg(r) \mid r \in E_2\} = cf(E_2)$. Observe that for every $r \in E_2$ holds that $fg(r) = r$, therefore for all sets R_1, R_2 of bridge rules with $r \in R_2$ holds that $R_1 \cup fg(R_2) = R_1 \cup \{r\} \cup fg(R_2 \setminus \{r\})$. Then, for all $E_1 \subseteq R_1 \subseteq br(M)$, $R_2 \subseteq br(M)$, and body-reduction functions fg' such that $Body(fg(r)) \subseteq Body(fg'(r))$ holds if $r \in E_2$, it holds that $M[R_1 \cup fg'(R_2)] \models \bot$, i.e., (E_1, E_2, fg) is a refined explanation.

($\Leftarrow$) Let (E_1, E_2, fg) be a refined explanation, i.e., $M[R_1 \cup fg'(R_2)] \models \bot$ for every $E_1 \subseteq R_1 \subseteq br(M)$, $R_2 \subseteq br(M)$, and body-reduction function fg' with $Body(fg(r)) \subseteq Body(fg'(r))$ for every $r \in E_2$. Consider the body-reduction function fg' such that for all $r \in br(M) \setminus E_2$ it holds that $fg'(r) = head(r) \leftarrow .$, i.e., $fg'(R'_2) = cf(R'_2)$ for every $R'_2 \subseteq br(M) \setminus E_2$. Observe that $Body(fg(r)) \subseteq Body(fg'(r))$ holds for every $r \in E_2$, therefore $M[R_1 \cup fg'(R_2)] \models \bot$ for every $E_1 \subseteq R_1 \subseteq br(M)$ and $R_2 \subseteq br(M)$. Hence, $M[R_1 \cup cf(R'_2)] \models \bot$ for every $R'_2 \subseteq br(M) \setminus E_2$ and thus $(E_1, E_2) \in E^{\pm}(M)$. □

Proof of Theorem 1. In this proof, we denote by $\overline{X}$ the complement of a set X w.r.t. $br(M)$, i.e., $\overline{X} = br(M) \setminus X$.

(a) Consider a pair $(E_1, E_2) \subseteq (br(M), br(M))$. Then for all diagnoses $(D_1, D_2) \in D^{\pm}(M)$, $D_1 \cap E_1$ or $D_2 \cap E_2$ or both are nonempty iff for all $(D_1, D_2) \subseteq (br(M), br(M))$ we have that

$$M\big[\overline{D_1} \cup cf(D_2)\big] \not\models \bot \quad \text{implies } D_1 \cap E_1 \neq \emptyset \text{ or } D_2 \cap E_2 \neq \emptyset$$

which (by reversing the implication and simplifying) is equivalent to

for all $(D_1, D_2) \subseteq (br(M), br(M))$ we have that

$$(D_1 \cap E_1 = \emptyset \text{ and } D_2 \cap E_2 = \emptyset) \quad \text{implies } M\big[\overline{D_1} \cup cf(D_2)\big] \models \bot.$$

As $A \cap B = \emptyset$ with $A, B \subseteq br(M)$ is equivalent to $A \subseteq \overline{B}$ we next obtain

for all $(D_1, D_2) \subseteq (br(M), br(M))$ we have that

$$(E_1 \subseteq \overline{D_1} \text{ and } D_2 \subseteq \overline{E_2}) \quad \text{implies } M\big[\overline{D_1} \cup cf(D_2)\big] \models \bot.$$

If we let $D_1 = \overline{R_1}$ and $D_2 = R_2$ this amounts to

for all $(R_1, R_2) \subseteq (br(M), br(M))$ we have that

$$(E_1 \subseteq R_1 \text{ and } R_2 \subseteq \overline{E_2}) \quad \text{implies } M\big[R_1 \cup cf(R_2)\big] \models \bot. \tag{C.1}$$

This proves the result (a) as this last condition is the one of an explanation (E_1, E_2) in Definition 6. Note that, if $(\emptyset, \emptyset) \in D^{\pm}(M)$, then no explanation exists; this is intentional and corresponds to the definitions of diagnosis and explanation for consistent systems.

(b) As $minHS_M(X)$ contains the $\subseteq$-minimal elements in $HS_M(X)$, and $E^{\pm}_m(M)$ contains the $\subseteq$-minimal elements in $E^{\pm}(M)$, (b) follows from (a). □

Proof of Corollary 1. Let $min(X)$ be the set of $\subseteq$-minimal elements in a collection X of sets. Then for every $(A, B) \in X \setminus min(X)$ there is a pair $(A', B') \in min(X)$ with $(A', B') \subseteq (A, B)$. Given $HS_M(min(X))$, every pair $(A, B) \in X \setminus min(X)$ is hit by every pair $(C, D) \in HS_M(min(X))$. Therefore $HS_M(min(X)) = HS_M(X)$. Then (a) immediately follows from Theorem 1(a), and (b) immediately follows from Theorem 1(b). □

Proof of Lemma 1. A collection of sets $C = \{C_1, \ldots, C_n\}$ over a universe, i.e., $C_i \subseteq U$, $1 \leq i \leq n$, can be seen as a *hypergraph* $\mathcal{H} = (U, C)$ with vertices U and hyperedges $C_i \in C$. If no hyperedge C_i is contained in any hyperedge C_j, $i \neq j$, it is called *simple*. A hitting set on C is called *transversal*, and the hypergraph (U, C') containing as hyperedges C' all minimal hitting sets of the hypergraph $\mathcal{H}$ is called *transversal hypergraph* $Tr(\mathcal{H})$.

We can map a collection $X = \{X^1, \ldots, X^n\}$ of pairs $X^i = (X^i_1, X^i_2)$ of sets, $X^i_1, X^i_2 \subseteq U$ bijectively to a collection $\mu(X) = \{\mu(X^1), \ldots, \mu(X^n)\}$ over $U \cup \{u' \mid u \in U\}$ where $\mu(X^i_1, X^i_2) = X^i_1 \cup \{u' \mid u \in X^i_2\}$. Then, (A, B) is a hitting set of X iff $\mu(A, B)$ is a hitting set of $\mu(X)$, and well-known results for transversal hypergraphs [7] carry over to minimal hitting sets over pairs.

In particular, given a simple hypergraph $\mathcal{H} = \mu(X)$, it holds that $Tr(Tr(\mu(X))) = \mu(X)$. This directly translates into the lemma, because $\mu(X)$ is a simple hypergraph due to incomparability (also called the anti-chain property) of X, and μ is bijective, therefore transversal hypergraphs can be mapped back to minimal hitting sets. □

Proof of Theorem 2. From Corollary 1(b) we have that $E^{\pm}_m(M) = minHS_M(D^{\pm}_m(M))$. Applying $minHS_M$ on both sides of this formula and then using Lemma 1 yields $minHS_M(E^{\pm}_m(M)) = minHS_M(minHS_M(D^{\pm}_m(M))) = D^{\pm}_m(M)$. □

Proof of Proposition 5. Let $(D_1, D_2) \in D^{\pm}_m(M)$ and let S be a witnessing belief state for it, i.e., S is an equilibrium of $M[br(M) \setminus D_1 \cup cf(D_2)]$. Towards contradiction, assume that $D_1 \cap D_2 \neq \emptyset$. Consider any bridge rule $r \in D_1 \cap D_2$ and let $C_h(r) = i$ and $\varphi(r) = p$. Furthermore, consider $r' = cf(r) = (i : p) \leftarrow .$, then $body(r') = \emptyset$ and thus r' is applicable in any belief state. Therefore, $r' \in app(br_i(M[br(M) \setminus D_1 \cup cf(D_2)]), S)$ and consequently $p \in \{\varphi(r) \mid r \in app(br_i(M[br(M) \setminus D_1 \cup cf(D_2)]), S)\}$. For $(D'_1, D'_2) = (D_1 \setminus \{r\}, D_2)$, we thus obtain that $p \in \{\varphi(r) \mid r \in app(br_i(M[br(M) \setminus D'_1 \cup cf(D'_2)]), S)\}$ and since all other bridge rules are as before, we conclude that $app(br_i(M[br(M) \setminus D'_1 \cup cf(D'_2)]), S) = app(br_i(M[br(M) \setminus D_1 \cup cf(D_2)]), S)$ for all $i \in C(M)$. Consequently S is an equilibrium of $M[br(M) \setminus D'_1 \cup cf(D'_2)]$ and $(D'_1, D'_2) \in D^{\pm}(M)$. But $(D'_1, D'_2) \subset (D_1, D_2)$ contradicts $(D_1, D_2) \in D^{\pm}_m(M)$, which proves the result. □

Lemma 4. *Let U be a splitting set of an MCS M and let $R_1, R_2 \subseteq br(M)$. Then, U is also a splitting set of $M[R_1 \cup cf(R_2)]$.*

Proof. Towards contradiction assume that U is not a splitting set for $M[R_1 \cup cf(R_2)]$, i.e., there exists a rule $r \in br(M[R_1 \cup cf(R_2)])$ such that $C_h(r) \in U$ and $C_b(r) \nsubseteq U$. Thus, there exists $(i : p) \in body(r)$ such that $i \notin U$. Since $body(r') = \emptyset$ for all

$r' \in cf(R_2)$, it follows that $r \in R_1$ and since $R_1 \subseteq br(M)$, it follows that $r \in br(M)$. By the assumption that $C_h(r) \in U$ and because U is a splitting set of M, it follows that $i \in U$ for all $(i : p) \in body(r)$, which contradicts that $C_b(r) \nsubseteq U$. Therefore, no such r can exist and U is also a splitting set of $M[R_1 \cup cf(R_2)]$. □

Lemma 5. *Let M be an MCS, let B be a set of bridge rules compatible with M, and let $U \subseteq C(M)$ be a splitting set for $M[B]$. Then, for every $i \in U$ and belief state $S = (S_1, \dots, S_n)$ of M it holds that:*

$$S_i \in \mathbf{ACC}_i\big(kb_i \cup app\big(br_i\big(M[b_U]\big), S\big)\big) \quad \textit{iff} \quad S_i \in \mathbf{ACC}_i\big(kb_i \cup app\big(br_i\big(M[B]\big), S\big)\big).$$

Proof. We first show that $br_i(M[b_U]) = br_i(M[B])$ holds for all $i \in U$:

($\subseteq$) From the definition of the bottom, b_U, it follows that $b_U \subseteq B$, thus $br_i(M[b_U]) \subseteq br_i(M[B])$.

($\supseteq$) Consider $r \in br_i(M[B])$, it holds that $C_h(r) = i$. Since U is a splitting set and $i \in U$ it follows that $r \in b_U$ by definition of the bottom b_U. Hence, $br_i(M[b_U]) \supseteq br_i(M[B])$.

As a consequence of the above, i.e., of $br_i(M[b_U]) = br_i(M[B])$, it follows that $app(br_i(M[b_U]), S) = app(br_i(M[B]), S)$ holds for all $i \in U$, and therefore it is also the case that $\mathbf{ACC}_i(kb_i \cup app(br_i(M[b_U]), S)) = \mathbf{ACC}_i(kb_i \cup app(br_i(M[B]), S))$, which proves the lemma. □

Observe that splitting sets preserve acceptability not only when bridge rules in the remainder of the MCS are modified (as in Lemma 5), but also when belief sets in the remainder are exchanged. For two belief states $S = (S_1, \dots, S_n)$ and $S' = (S'_1, \dots, S'_n)$ of an MCS, we say that S coincides with S' on U, written $S =_U S'$, if for all $i \in U$ holds $S_i = S'_i$.

Lemma 6. *Let M be an MCS, let B be a set of bridge rules compatible with M, and let U be a splitting set for $M[B]$. Furthermore, let $S = (S_1, \dots, S_n)$ and $S' = (S'_1, \dots, S'_n)$ be belief states of M, and let $b_U \subseteq R \subseteq B$. Then, $S =_U S'$ implies $\mathbf{ACC}_i(kb_i \cup app(br_i(M[B]), S)) = \mathbf{ACC}_i(kb_i \cup app(br_i(M[R]), S'))$.*

Proof. Since $b_U \subseteq R$ it holds for all $i \in U$ that $br_i(M[B]) = br_i(M[R])$. Furthermore, because U is a splitting set, it follows that $c \in U$ for all $(c : p) \in body(r)$ such that $r \in br_i(M[B])$ and $i \in U$. As a consequence $p \in S_c$ iff $p \in S'_c$ since S and S' coincide on U and $r \in br_i(M[B])$ iff $r \in br_i(M[R])$. □

Proof of Proposition 6. For reasoning about explanations, the concept of explanation range proves to be useful. For a given pair $E = (E_1, E_2) \in 2^{br(M)} \times 2^{br(M)}$ of sets of bridge rules and $B \subseteq br(M)$, the *explanation range* of E with respect to B is $Rg(E, B) = \{(R_1, R_2) \mid E_1 \subseteq R_1 \subseteq B$ and $R_2 \subseteq B \setminus E_2\}$. Intuitively, $Rg(E, B)$ are "relevant pairs" for E wit respect to the upper bound B. It follows directly from Definition 6 that, $E = (E_1, E_2) \in E^{\pm}(M)$ iff $M[R_1 \cup cf(R_2)] \models \bot$ for all $(R_1, R_2) \in Rg(E, br(M))$.

In the following we prove Item (i): $E \in E^{\pm}(M[b_U])$ holds iff $E \in E^{\pm}(M)$ holds and E is U-headed.

($\Rightarrow$) Let $(R'_1, R'_2) \in Rg(E, br(M))$ be arbitrary, then both $R'_1 \subseteq br(M)$ and $R'_2 \subseteq br(M)$. By Lemma 4, U is also a splitting set for the MCS $N' = M[R'_1 \cup cf(R'_2)]$.

Let $R_1 = R'_1 \cap b_U$ and let $R_2 = R'_2 \cap b_U$. As $E_1, E_2 \subseteq b_U$, it follows that $(R_1, R_2) \in Rg(E, b_U)$. Because E is an explanation of $M[b_U]$, it holds for $N = M[R_1 \cup cf(R_2)]$ that $N \models \bot$, i.e., for every belief state S exists a context $i \in U$ with $S_i \notin \mathbf{ACC}_i(kb_i \cup app(br_i(N), S))$.

Since $B = R'_1 \cup cf(R'_2)$ is compatible with M and U is a splitting set for $N' = M[B]$, we conclude from Lemma 5 that for every belief state S it holds that $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(N'), S))$ iff $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(N), S))$. Since $N \models \bot$ this implies that for every S there exists some $i \in U$ such that $S_i \notin \mathbf{ACC}_i(kb_i \cup app(br_i(N'), S))$ and thus $N' \models \bot$.

Since $(R'_1, R'_2) \in Rg(E, br(M))$ is arbitrary, it follows that $E \in E^{\pm}(M)$. Furthermore, E is U-headed by definition.

($\Leftarrow$) Let $E = (E_1, E_2) \in E^{\pm}(M)$ such that E is U-headed, and consider some arbitrary $(R_1, R_2) \in Rg(E, b_U)$. Since $b_U \subseteq br(M)$, we conclude that $(R_1, R_2) \in Rg(E, br(M))$. Since E is an explanation of M, it follows that $N = M[R_1 \cup cf(R_2)]$ is such that $N \models \bot$. As this holds for every $(R_1, R_2) \in Rg(E, b_U)$, it follows that $(E_1, E_2) \in E^{\pm}(M[b_U])$.

This establishes item (i).

Next we prove Item (ii): $D \in D^{\pm}(M[b_U])$ holds iff there exists $D' \in D^{\pm}(M)$ such that $D \subseteq D'$.

($\Rightarrow$) Let $D = (D_1, D_2) \in D^{\pm}(M[b_U])$. Then, there exists an equilibrium S of $M[R]$ where $R = (b_U \setminus D_1) \cup cf(D_2)$. Consider $(D'_1, D'_2) = (D_1 \cup (br(M) \setminus b_U), D_2)$ and observe that $(br(M) \setminus D'_1) \cup cf(D'_2) = R$, because $br(M) \setminus D'_1 = b_U \setminus D_1$. Since S is an equilibrium of $M[R]$, it follows that $D' \in D^{\pm}(M)$.

($\Leftarrow$) Assume $D' \in D^{\pm}(M)$ where $D' = (D'_1, D'_2)$. First assume that $E^{\pm}(M[b_U]) = \emptyset$, i.e., $M[b_U]$ is consistent. Then, $D = (\emptyset, \emptyset) \in D^{\pm}(M[b_U])$, hence $D \subseteq D'$ and $D \in D^{\pm}(M[b_U])$.

Otherwise, $E^{\pm}(M[b_U]) \neq \emptyset$. Consider $(D_1, D_2) = (D'_1 \cap b_U, D'_2 \cap b_U)$ and let $R' = br(M) \setminus D'_1 \cup cf(D'_2)$ and $R = b_U \setminus D_1 \cup cf(D_2)$. Observe that $br_j(M[R]) = \emptyset$ for all $j \in C(M) \setminus U$, because $R \subseteq b_U \cup cf(b_U)$ and for no rule $r \in b_U \cup cf(b_U)$ it holds that $C_h(r) = j$.

As $M[\emptyset]$ is consistent, there exists some $S^0_j \in \mathbf{ACC}_j(kb_j)$ for every $j \in C(M)$. Let $S' = (S'_1, \dots, S'_n)$ be an equilibrium for $M[R']$ (which exists because $D' \in D^{\pm}(M)$). Let $S = (S_1, \dots, S_n)$ such that $S_i = S'_i$ if $i \in U$, and $S_i = S^0_i$ otherwise. Then, S is an equilibrium for $M[R]$. Indeed, first consider $i \in C(M) \setminus U$. Since $br_i(M[R]) = \emptyset$, it follows that

$app(br_i(M[R]), S) = \emptyset$, hence $S_i^0 \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R]), S))$. Second, consider $i \in U$. Note that U is a splitting set of $M[R]$, because $br_j(M[R]) = \emptyset$ for all $j \in C(M) \setminus U$. Since $b_U \subseteq R \subseteq R'$ and $S =_U S'$, it follows from Lemma 6 that $\mathbf{ACC}_i(kb_i \cup app(br_i(M[R]), S)) = \mathbf{ACC}_i(kb_i \cup app(br_i(M[R']), S'))$. From $S'_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R']), S'))$ and $S_i = S'_i$, it thus follows that $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R]), S))$.

Consequently, $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R]), S))$ for all $i \in C(M)$; hence S is an equilibrium of $M[R]$. Since $R_1 \cup R_2 \subseteq b_U$, it follows that $D \in D^{\pm}(M[b_U])$. □

Proof of Corollary 2. Let $E \in E_m^{\pm}(M[b_U])$, then it follows from Proposition 6 that $E \in E^{\pm}(M)$ and E is U-headed. Assume for a contradiction that $E \notin E_m^{\pm}(M)$. Hence, there exists some $E' \in E^{\pm}(M)$ such that $E' \subset E$. Since E is U-headed, it follows that E' also is U-headed. Thus by Proposition 6 it follows that $E' \in E^{\pm}(M[b_U])$, which contradicts that $E \in E_m^{\pm}(M[b_U])$. □

Proof of Proposition 7. As in the proof of Proposition 6, let $(R_1, R_2) \in Rg(E, B)$ iff $E_1 \subseteq R_1 \subseteq B$ and $R_2 \subseteq B \setminus E_2$.

Wlog. assume that $M = (C_1, \dots, C_n)$, $U = \{1, \dots, k\}$, and $U' = \{k+1, \dots, n\}$, where $1 \le k \le n$. Towards a contradiction assume that some $E = (E_1, E_2) \in E_m^{\pm}(M)$ exists which contains rules from both, b_U and $b_{U'}$. Consider an arbitrary $(R_1, R_2) \in Rg(E, br(M))$. Since E is an explanation, it holds that $M[R_1 \cup cf(R_2)] \models \bot$.

Given $R = (R_1, R_2)$ s.t. $R_1, R_2 \subseteq b_U \subseteq br(M)$ and V, we say that *the V-projection of R is inconsistent* iff $M[(R_1 \cap V) \cup cf(R_2 \cap V)] \models \bot$. We prove that for every $R = (R_1, R_2) \in Rg(E, br(M))$ either its U-projection or its U'-projection is inconsistent, or both.

Towards contradiction assume that neither projection is inconsistent. Then, there exists an equilibrium $S = (S_1, \dots, S_n)$ of $M[(R_1 \cap U) \cup cf(R_2 \cap U)]$ and an equilibrium $S' = (S'_1, \dots, S'_n)$ of $M[(R_1 \cap U') \cup cf(R_2 \cap U')]$. Consider the belief state $S'' = (S_1, \dots, S_k, S'_{k+1}, \dots, S'_n)$. By Lemma 6, it holds that $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R_1 \cup cf(R_2)]), S''))$ for all $i \in U$, because U is a splitting set of M, $b_U \subseteq (R_1 \cap U) \cup cf(R_2 \cap U) \subseteq R_1 \cup cf(R_2)$, and $S =_U S''$. Analogously, it holds that $S'_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R_1 \cup cf(R_2)]), S''))$ for all $i \in U'$. Consequently, S'' is an equilibrium of $M[R_1 \cup cf(R_2)]$, which contradicts that E is an explanation. Therefore, for every $R \in Rg(E, br(M))$ it holds that either the U-projection of R, the U'-projection of R, or both are inconsistent.

Next, we distinguish for all $R \in Rg(E, br(M))$ which projections are inconsistent.

Case (1): For every $R \in Rg(E, br(M))$ its U-projection is inconsistent. Then, $E' = (E_1 \cap b_U, E_2 \cap b_U)$ is an explanation, since for every $R' \in Rg(E', br(M))$ it holds that R' is a U-projection of some $R \in Rg(E, br(M))$, which is inconsistent. Since $E_1 \cup E_2 \nsubseteq b_U$, we have $E' \subset E$. Since $E' \in E^{\pm}(M)$, it follows that $E \notin E_m^{\pm}(M)$, which contradicts the assumption that $E \in E_m^{\pm}(M)$.

Case (2): For all $R \in Rg(E, br(M))$ it holds that the U'-projection is inconsistent. Analogously to the previous case, we conclude that $E' = (E_1 \cap b_{U'}, E_2 \cap b_{U'})$ is an explanation of M such that $E' \subset E$, which contradicts the assumption that $E \in E_m^{\pm}(M)$.

Case (3): Neither case (1) nor case (2) applies. That is, for some $R = (R_1, R_2) \in Rg(E, br(M))$ the U-projection is consistent, and also for some $R' = (R'_1, R'_2) \in Rg(E, br(M))$ the U'-projection is consistent. This means that there exists some belief state $S = (S_1, \dots, S_n)$ such that $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[(R_1 \cap b_U) \cup cf(R_2 \cap b_U)]), S))$ for all $i \in C(M)$ and there exists some belief state $S' = (S'_1, \dots, S'_n)$ such that $S'_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[(R'_1 \cap b_{U'}) \cup cf(R'_2 \cap b_{U'})]), S))$ for all $i \in C(M)$.

Now consider $R'' = (R''_1, R''_2) = ((R_1 \cap b_U) \cup (R'_1 \cap b_{U'}), (R_2 \cap b_U) \cup (R'_2 \cap b_{U'}))$. First, we show that $R'' \in Rg(E, br(M))$. Since U and U' partition $C(M)$ it holds that $E_1 = (E_1 \cap b_U) \cup (E_1 \cap b_{U'})$; since $E_1 \subseteq R_1$, clearly $E_1 \cap b_U \subseteq R_1 \cap b_U$. Analogously, it holds that $E_1 \cap b_{U'} \subseteq R_1 \cap b_{U'}$. Consequently, $E_1 = (E_1 \cap b_U) \cup (E_1 \cap b_{U'}) \subseteq (R_1 \cap b_U) \cup (R'_1 \cap b_{U'})$; hence $E_1 \subseteq R''_1 \subseteq br(M)$. For R''_2 observe that $(R_2 \cup R'_2) \cap E_2 = \emptyset$ since both, R_2 and R'_2, are disjoint with E_2 by definition. Therefore $((R_2 \cap b_U) \cup (R'_2 \cap b_{U'})) \cap E_2 = \emptyset$; hence $R''_2 \subseteq br(M) \setminus E_2$. In conclusion, it holds that $R'' \in Rg(E, br(M))$.

Second, we show that $S'' = (S_1, \dots, S_k, S'_{k+1}, \dots, S'_n)$ is an equilibrium of the MCS $M[R''_1 \cup cf(R''_2)]$. Since $S'' =_U S$ and as already shown, $R_1 \cap b_U \subseteq R''_1$ and $cf(R_2 \cap b_U) \subseteq cf(R''_2)$, it follows by Lemma 6 that $S_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R''_1 \cup cf(R''_2)]), S''))$ for all $i \in U$. Analogously, the same is shown for U', i.e., $S'_i \in \mathbf{ACC}_i(kb_i \cup app(br_i(M[R''_1 \cup cf(R''_2)]), S''))$ for all $i \in U'$. Therefore, S'' is an equilibrium of $M[R''_1 \cup cf(R''_2)]$. Since $R'' \in Rg(E, br(M))$, it follows that $E \notin E^{\pm}(M)$. This is a contradiction to the assumption that $E \in E_m^{\pm}(M)$.

Since all cases yield a contradiction, it follows that every $E \in E_m^{\pm}(M)$ is either U-headed or U'-headed. □

Proof of Corollary 3. ($\subseteq$) Let $E_m^{\pm}(M)$. Then by Proposition 7, E is either U-headed or U'-headed. If E is U-headed, then by Proposition 6 $E \in E^{\pm}(M[b_U])$. Assume that $E \notin E_m^{\pm}(M[b_U])$, hence some $E' \subset E$ exists such that $E' \in E_m^{\pm}(M[b_U])$. By Proposition 7, $E' \in E_m^{\pm}(M)$. This contradicts that $E \in E_m^{\pm}(M)$, which gives $E \in E_m^{\pm}(M[b_U)$. Analogously, if E is U'-headed, then $E \in E_m^{\pm}(M[b_{U'}])$. It follows that $E \in E_m^{\pm}(M[b_U]) \cup E_m^{\pm}(M[b_{U'}])$.

($\supseteq$) Let $E \in E_m^{\pm}(M[b_U])$ (respectively $E \in E_m^{\pm}(M[b_{U'}])$). Since U (respectively U') is a splitting set of M, from Corollary 2 it follows that $E \in E_m^{\pm}(M)$. In conclusion it holds that $E_m^{\pm}(M) \supseteq E_m^{\pm}(M[b_U]) \cup E_m^{\pm}(M[b_{U'}])$. □

Proof of Proposition 8. By Corollary 3, $E_m^{\pm}(M) = E_m^{\pm}(M[b_U]) \cup E_m^{\pm}(M[b_{U'}])$, while by Theorem 2 each diagnosis is a minimal hitting set on $E_m^{\pm}(M)$. Because U and U' partition M, $E_m^{\pm}(M[b_U])$ and $E_m^{\pm}(M[b_{U'}])$ are on disjoint sets. Therefore the minimal hitting set of their unions is the pairwise combination of their minimal hitting sets. That is, $(D_1, D_2) \in minHS_M(E_m^{\pm}(M))$ iff $(D_1, D_2) = (A_1 \cup B_1, A_2 \cup B_2)$ with $(A_1, A_2) \in minHS_M(E_m^{\pm}(M[b_U])$ and $(B_1, B_2) \in minHS_M(E_m^{\pm}(M[b_{U'}])$. From Theorem 2 it follows that $D_m^{\pm}(M) = minHS_M(M)$. This proves the proposition. □

Proof of Lemma 2. ($\Rightarrow$) Let $S = (S_1, \dots, S_n)$. Then $S_i \in \mathbf{ACC}(kb_i \cup H_i)$, where the set H_i of active bridge rule heads at context C_i is $app(br_i, S)$. Bridge rule applicability depends on output beliefs only, hence $app(br_i, S) = app(br_i, S^o)$. Thus $S^o = (S^o_1, \dots, S^o_n)$ with $S^o_i = S_i \cap OUT_i$ is an output-projected equilibrium of M.

($\Leftarrow$) Let $T = (T_1, \dots, T_n)$, then, as T is an output-projected equilibrium, for each i, $1 \le i \le n$, $T \in \{S^o_i \mid S_i \in \mathbf{ACC}_i(kb_i \cup \{\varphi(r) \mid r \in app(br_i, T)\})\}$, and therefore for each Q there exists some belief set S_i such that $S_i \in \mathbf{ACC}_i(kb_i \cup \{\varphi(r) \mid r \in app(br_i, T)\})$ and $T = S_i \cap OUT_i$. If we take for each i some arbitrary S_i satisfying the above condition, we obtain $S = (S_1, \dots, S_n)$. As T and S agree on all output beliefs of all contexts, $app(br_i, T) = app(br_i, S)$ and hence S is an equilibrium of M. By construction of S, it holds that also $S^o = T$. $\square$

Proof of Proposition 9. (*Membership*) Given an MCS $M = (C_1, \dots, C_n)$ we compute OUT_i for all C_i in $\mathcal{O}(|br(M)|)$, then we guess output projected belief sets $T_i \subseteq OUT_i$, $1 \le i \le n$, yielding an output-projected belief state T. We evaluate bridge rule applicability of all rules in T in time $\mathcal{O}(|br(M)|)$ and thereby obtain a set of active bridge rule heads H_i for each context C_i, $1 \le i \le n$. Finally we check acceptability of T for all contexts C_i, i.e., whether $T \in \mathbf{ACC}_i(kb_i \cup H_i)|_{OUT_i}$. We accept if all contexts accept, otherwise we reject. This check is a conjunction of n independent acceptability checks of maximum complexity equal to the smallest upper bound C on context complexities which is $C = \mathcal{CC}(M)$. If C is closed under conjunction, we can unite these checks into one check of complexity C over an instance of size $\mathcal{O}(|M|)$. Then the overall acceptability check is in C as well. This way we check the output-projected equilibrium property for all possible output-projected equilibria. Therefore if no computation path accepts, then the MCS M is inconsistent. If there is one path that accepts, then the output-projected belief state T corresponding to the guesses on this path is an output-projected equilibrium which fulfills all conditions of Definition 13. Therefore M is consistent iff at least one path accepts. Hence if C is closed under conjunction and projection, then the guess of size $\mathcal{O}(|br(M)|)$ can be projected away (i.e., incorporated into I', see Section 5) and the complexity of MCSEQ is in C. For $\mathcal{CC}(M) = \mathbf{P}$ (which is presumably not closed under projection) the complexity of MCSEQ is in **NP**.

(**NP**-*hardness* for $\mathcal{CC}(M) = \mathbf{P}$) We show that consistency checking in an MCS M with lower context complexity $\mathcal{CC}(M) \ge \mathbf{P}$ is **NP**-hard. We use the part of the MCS structure in Fig. 3a labeled with MCSEQ. We reduce a 3-SAT instance $F = c_1 \wedge \dots \wedge c_n$ on variables $\mathcal{X} = \{x_1, \dots, x_k\}$ and clauses $c_i = c_{i,1} \vee c_{i,2} \vee c_{i,3}$ with $c_{i,j} \in \mathcal{X} \cup \{\neg x \mid x \in \mathcal{X}\}$ to consistency checking in an MCS $M = (C_{gen_U}, C_{eval_F}, C_{check})$. Context $C_{gen_U} = (L_{GUESS}, kb_{gen_U}, br_{gen_U})$ with $kb_{gen_U} = \mathcal{X}$ and $br_{gen_U} = \emptyset$ has linear complexity, while $C_{eval_F} = (L_{ASP}, kb_{eval_F}, br_{eval_F})$ and $C_{check} = (L_{ASP}, kb_{check}, br_{check})$ have context complexity **P**. M contains the following bridge rules:

$$r_{u,i}: \quad (eval_F : x_i) \leftarrow (gen_U : x_i). \quad \forall i : 1 \le i \le k. \tag{C.2}$$

$$r_\alpha: \quad (check : nsat) \leftarrow \mathbf{not}\ (eval_F : sat). \tag{C.3}$$

Hence $br_{eval_F} = \{r_{u,i} \mid \forall i : 1 \le i \le k\}$ and $br_{check} = \{r_\alpha\}$. The knowledge base kb_{eval_F} is as follows:

$$sat_i \leftarrow l_{i,1}. \qquad sat_i \leftarrow l_{i,2}. \qquad sat_i \leftarrow l_{i,3}. \quad \forall i : 1 \le i \le n. \tag{C.4}$$

$$sat \leftarrow sat_1, \dots, sat_n. \tag{C.5}$$

$$\text{where } l_{i,j} \text{ is } \begin{cases} x_v & \text{if } c_{i,j} = x_v \\ not\ x_v & \text{if } c_{i,j} = \neg x_v. \end{cases}$$

The knowledge base kb_{check} is as follows:

$$\perp \leftarrow nsat. \tag{C.6}$$

Context C_{gen_U} accepts all possible subsets of $\mathcal{X}$, representing all possible truth assignments for the variables $\mathcal{X}$. (C.2) imports the truth assignment into C_{eval_F}, which evaluates F under that truth assignment using rules (C.4) and (C.5). Then C_{eval_F} puts the belief sat in its belief set iff F is satisfied given the truth assignment accepted by C_{gen_U}. Finally C_{check} imports the belief $nsat$ iff sat is not accepted at C_{eval_F}. Therefore constraint (C.6) makes C_{check} inconsistent, i.e., accepts no belief set, iff sat is not true in C_{eval_F} iff there is no satisfying truth assignment for F. Therefore, if F has a satisfying assignment with variables $\mathcal{T} \subseteq \mathcal{X}$ set to **t** and variables $\mathcal{X} \setminus \mathcal{T}$ set to **f**, then M has an equilibrium $S = (S_{gen_U}, S_{eval_F}, S_{check})$ where $S_{gen_U} = \mathcal{T}$, $S_{eval_F} = \mathcal{T} \cup \{sat_i \mid 1 \le i \le n\} \cup \{sat\}$, and $S_{check} = \emptyset$. Conversely, if M has an equilibrium $S = (S_{gen_U}, S_{eval_F}, S_{check})$, then S_{check} does not contain $nsat$ due to constraint (C.6). Hence S_{eval_F} must contain sat, thus S_{eval_F} contains $\{sat\} \cup \{sat_i \mid 1 \le i \le n\}$ due to (C.5). It follows that the set of bridge rule heads active at C_{eval_F} corresponds to a satisfying assignment of F. This shows that MCS M is consistent iff F is a satisfiable 3-SAT instance. As the size of M is linear in the size of the formula F and 3-SAT is an **NP**-hard problem, hardness for equilibrium existence follows.

($\mathcal{CC}(M)$-*hardness*) We show that consistency checking in an MCS M with lower context complexity $\mathcal{CC}(M) \ge C$ is C-hard if C is a class with complete problems that is closed under conjunction and projection. For that we use part of the MCS structure labeled with MCSEQ in Fig. 3b. We reduce context acceptability checking, i.e., an instance (H_a, S_a), $C_a = (kb_a, br_a, L_a)$ with IN_a, OUT_a and context complexity $\mathcal{CC}(C_a)$ to consistency checking in an MCS $M = (C_{a'}, C_{check})$ such that the context complexity $\mathcal{CC}(C_{a'}) = \mathcal{CC}(C_a)$ and $\mathcal{CC}(C_{check}) = \mathbf{P}$. Intuitively, $C_{a'}$ gets input H_a, bridge rule r_α is applicable only if S_a is accepted by H_a, and C_{check} verifies whether r_α is applicable. Then M is consistent iff (H_a, S_a), C_a is a 'yes' instance. Formally, $C_{a'} = (kb_a \cup H_a, \emptyset, L_a)$ uses knowledge base and logic from C_a, while $C_{check} = (kb_{check}, br_{check}, L_{ASP})$ use the specific logic L_{ASP} that can be decided in **P**. Bridge rules of M are as follows:

$$r_\alpha:\quad (check : equal_{S'_a}) \leftarrow l_1, \ldots, l_j, \ldots, l_{|OUT_a|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_a \wedge s_j \in S_a \\ \textbf{not}\, s_j & \text{if } s_j \in OUT_a \wedge s_j \notin S_a. \end{cases} \tag{C.7}$$
$$r_{en}:\quad (check : en) \leftarrow. \tag{C.8}$$

The knowledge base kb_{check} is as follows:

$$\perp \leftarrow \textit{not } equal_{S'_a}, en. \tag{C.9}$$

Bridge rule r_{en} ensures that C_{check} fulfills our assumption that a context without input is consistent. Wlog. we assume that C_a accepts some belief set given input H_a. $C_{a'}$ contains the logic of C_a and its knowledge base already contains bridge rule heads H_a. Therefore $C_{a'}$ accepts a belief set S_a^{full}, such that $S_a^{full} \cup OUT_a = S_a$, iff (H_a, S_a), C_a is a 'yes' instance. Therefore, belief state $S = (S_a^{full}, \{equal_{S'_a}, en\})$ is an equilibrium iff (H_a, S_a), C_a is a 'yes' instance. All belief states where $C_{a'}$ accepts a belief set T with $T \cap OUT_a \neq S_a$ trigger constraint (C.9) and therefore lead to an inconsistency. Therefore M has an equilibrium, and this equilibrium is S iff context (H_a, S_a), C_a is a 'yes' instance for context acceptability checking. We thus have reduced context acceptability checking to consistency checking in M and hardness follows. For an alternative reduction showing **NP**-hardness for **P**-contexts see [38, p. 57]. □

Proof of Proposition 10. (*Membership*) Given MCS M and $D_1, D_2 \subseteq br(M)$, we compute $M' = M[br(M) \setminus D_1 \cup cf(D_2)]$ and return the result of deciding MCSEQ on M'. By Definition 4, this returns 'yes' iff $(D_1, D_2) \in D^{\pm}(M)$. The transformation can be done in time $\mathcal{O}(|M|)$ therefore MCSD is in the same complexity class as MCSEQ.

(*Hardness*) Deciding whether $(\emptyset, \emptyset)$ is a diagnosis of M can be decided by checking consistency of M, because $(\emptyset, \emptyset) \in D^{\pm}(M)$ iff M is consistent. Therefore MCSD is as hard as MCSEQ for respective context complexity. □

Proof of Proposition 11. (*Membership*) Given an MCS M and $D_1, D_2 \subseteq br(M)$, we solve two independent decision problems: (a) we decide whether (D_1, D_2) is a diagnosis of M, and (b) we check whether a smaller diagnosis $(D', D'') \subset (D_1, D_2)$ exists in M. We return 'yes' if (a) returns 'yes' and (b) returns 'no'. Thus, this procedure returns 'yes' iff (a) (D_1, D_2) is a diagnosis and (b) no $\subseteq$-smaller diagnosis exists. Therefore the computation yields the correct result. For (a) we decide MCSD on M and (D_1, D_2). For (b) we guess for each bridge rule in D_1 whether it is contained in D', and for each bridge rule in D_2 whether it is contained in D''. Then we continue with the decision procedure MCSD on M and (D', D''), i.e., we guess presence of output belief sets, evaluate bridge rule applicability, and check acceptability for each context. Consequently for deciding (b) we decide the complement of a polynomial projection of MCSD. Therefore MCSD$_m$ is in the complexity class of solving the MCSD problem and independently solving the complement of a polynomially projected MCSD problem. Hence if $\mathcal{CC}(M)$ is closed under conjunction and projection, then the complexity of MCSD$_m$ is in $\mathcal{D}(\mathcal{CC}(M))$. For $\mathcal{CC}(M) = \mathbf{P}$ (which is presumably not closed under projection) the complexity of MCSD$_m$ is in $\mathbf{D_1^P}$.

($\mathbf{D^P}$-*hardness* for $\mathcal{CC}(M) = \mathbf{P}$) We reuse ideas from the MCSEQ hardness proof for 3-SAT, but we now use the complete topology shown in Fig. 3a. We reduce two 3-SAT instances F and G on variables $\mathcal{X}$ and $\mathcal{Y}$, respectively, to minimal diagnosis recognition on MCS $M = (C_{gen_V}, C_{eval_F}, C_{gen_U}, C_{eval_G}, C_{check})$. Intuitively, C_{gen_U} and C_{eval_F} provide **NP**-hardness for satisfiability of F, while C_{gen_V} and C_{eval_G} provide **coNP**-hardness for unsatisfiability of G. C_{gen_U} and C_{eval_F} are constructed from F exactly as for the proof of MCSEQ hardness. Similarly, C_{gen_V} and C_{eval_G} are constructed from G with bridge rules $r_{v,j}$ transferring a guessed set $V \subseteq \mathcal{Y}$ from C_{gen_V} to C_{eval_G}. The bridge rules in M are as follows:

$$r_{u,i}:\quad (eval_F : x_i) \leftarrow (gen_U : x_i). \quad \forall i : 1 \leq i \leq |\mathcal{X}|. \tag{C.10}$$
$$r_{v,j}:\quad (eval_G : y_j) \leftarrow (gen_V : y_j). \quad \forall j : 1 \leq j \leq |\mathcal{Y}|. \tag{C.11}$$
$$r_\alpha:\quad (check : nsat_F) \leftarrow \textbf{not}\ (eval_F : sat). \tag{C.12}$$
$$r_\gamma:\quad (check : nsat_G) \leftarrow \textbf{not}\ (eval_G : sat). \tag{C.13}$$

Context C_{check} has the following knowledge base kb_{check}:

$$\perp \leftarrow nsat_F. \tag{C.14}$$
$$\perp \leftarrow nsat_G. \tag{C.15}$$

If F and G are both satisfiable, M is consistent so $(\emptyset, \emptyset) \in D_m^{\pm}(M)$. If F is satisfiable and G is unsatisfiable, M is inconsistent and a minimal diagnosis for M is $(\{r_\gamma\}, \emptyset)$. If F and G are both unsatisfiable, M is inconsistent and a minimal diagnosis is $(\{r_\alpha, r_\gamma\}, \emptyset) \in D_m^{\pm}(M)$. If F is unsatisfiable and G is satisfiable, M is inconsistent; $(\{r_\gamma\}, \emptyset)$ is no minimal diagnosis, because every diagnosis containing r_γ in D_1 must also contain r_α in D_1 to restore consistency in M. Therefore $(\{r_\gamma\}, \emptyset)$ is a minimal diagnosis of M iff F is satisfiable and G is unsatisfiable. Therefore recognizing a minimal diagnosis in an MCS with $\mathcal{CC}(M) = \mathbf{P}$ is hard for $\mathbf{D^P}$. Note that it is possible to do this reduction with one context that evaluates F and G and checks the result, using bridge rules that guess U and V and bridge rules that individually activate satisfiability checking for F and G. However this would make the reduction less readable.

($\mathbf{D}(\mathcal{CC}(M))$-*hardness*) We show that recognizing minimal diagnoses in an MCS M with lower context complexity $\mathcal{CC}(M) \geq C$ is hard for $\mathbf{D}(C)$ if C is a class with complete problems that is closed under conjunction and projection. We reduce two context complexity check instances (H_a, S_a), C_a with IN_a, OUT_a and (H_b, S_b), C_b with IN_b, OUT_b to an MCS $M = (C_{a'}, C_{b'}, C_{check})$ with the topology shown in Fig. 3b. Similar to the generic hardness reduction for MCSEQ, we reduce H_a and $C_a = (kb_a, br_a, L_a)$ to the context $C_{a'} = (kb_a \cup H_a, \emptyset, L_a)$ and we reduce H_b and $C_b = (kb_b, br_b, L_b)$ to the context $C_{b'} = (kb_b \cup H_b, \emptyset, L_b)$. Then $\mathcal{CC}(C_{a'}) = \mathcal{CC}(C_a)$ and $\mathcal{CC}(C_{b'}) = \mathcal{CC}(C_b)$. Furthermore $C_{a'}$ accepts a belief set S_a^{full} with $S_a^{full} \cap OUT_a = S_a$ iff (H_a, S_a), C_a is a 'yes' instance. Similarly $C_{b'}$ accepts a belief set S_b^{full} with $S_b^{full} \cap OUT_b = S_b$ iff (H_b, S_b), C_b is a 'yes' instance. The bridge rules br_{check} are as follows.

$$r_\alpha\colon \quad (check : equal_{S'_a}) \leftarrow l_1, \ldots, l_j, \ldots, l_{|OUT_a|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_a \wedge s_j \in S_a \\ \textbf{not } s_j & \text{if } s_j \in OUT_a \wedge s_j \notin S_a. \end{cases} \tag{C.16}$$
$$r_\gamma\colon \quad (check : equal_{S'_b}) \leftarrow l_1, \ldots, l_j, \ldots, l_{|OUT_b|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_b \wedge s_j \in S_b \\ \textbf{not } s_j & \text{if } s_j \in OUT_b \wedge s_j \notin S_b. \end{cases} \tag{C.17}$$
$$r_{en}\colon \quad (check : en) \leftarrow . \tag{C.18}$$

The knowledge base kb_{check} is as follows:

$$n\_equal \leftarrow not\ equal_{S'_a}. \tag{C.19}$$
$$n\_equal \leftarrow not\ equal_{S'_b}. \tag{C.20}$$
$$\bot \leftarrow not\ n\_equal, en. \tag{C.21}$$

Bridge rule r_{en} ensures that C_{check} fulfills our assumption that a context without input is consistent. Wlog. we assume that C_a and C_b accept some belief set given input H_a and H_b, respectively. Bridge rule r_α adds $equal_{S'_a}$ to C_{check} iff the first instance (H_a, S_a), C_a we reduce from is a 'yes' instance. The same is true for r_γ, $equal_{S'_b}$ and the second instance. Therefore there exists an equilibrium $S = (S_a^{full}, S_b^{full}, \{equal_{S'_a}, equal_{S'_b}, en\})$ in M, i.e., $(\emptyset, \emptyset) \in D_m^{\pm}(M)$, iff both instances are 'yes' instances. Moreover, if the first instance is a 'yes' instance and the second instance is a 'no' instance, then the system is inconsistent and there is a minimal diagnosis $(\emptyset, \{r_\gamma\}) \in D_m^{\pm}(M)$. If both instances are 'no' instances, activating $equal_{S'_b}$ is not sufficient for restoring consistency, and a minimal diagnosis for M is then $(\emptyset, \{r_\alpha, r_\gamma\})$. Therefore $(\emptyset, \{r_\gamma\})$ is a minimal diagnosis for M iff (H_a, S_a), C_a is a 'yes' instance and (H_b, S_b), C_b is a 'no' instance of context acceptability checking. Therefore we have established that MCSD$_m$ is hard for $\mathbf{D}(C)$. Note that by nesting contexts $C_{a'}$ and $C_{b'}$ into a new context it is possible, although more complicated, to obtain a reduction with just one context that is hard for $\mathbf{D}(C)$. □

Proof of Proposition 12. (*Membership*) For deciding $(E_1, E_2) \in E^{\pm}(M)$, we guess $R_1, R_2 \subseteq br(M)$ and check whether $E_1 \subseteq R_2$ and $R_2 \subseteq br(M) \setminus E_2$. If not, we immediately reject, otherwise we decide MCSEQ of $M[R_1 \cup cf(R_2)]$. Then all execution paths reject iff (E_1, E_2) is an explanation. Therefore, if $\mathcal{CC}(M)$ is a class with complete problems that is closed under conjunction and projection, the complexity is in $\mathbf{co}$-$\mathcal{CC}(M)$. For $\mathcal{CC}(M) = \mathbf{P}$ (which is presumably not closed under projection) we obtain that MCSE is in $\mathbf{coNP}$.

($\mathbf{coNP}$-*hardness* for $\mathcal{CC}(M) = \mathbf{P}$) We reuse the MCSEQ hardness proof where a 3-SAT instance F was reduced to MCS $M = (C_{gen_U}, C_{eval_F}, C_{check})$. Then satisfiability of F implies consistency, therefore $E^{\pm}(M) = \emptyset$, i.e., no inconsistency explanations exist. Unsatisfiability of F implies inconsistency, and in that case, $(\{r_\alpha\}, \emptyset)$ is an inconsistency explanation of M. Therefore $(\{r_\alpha\}, \emptyset)$ is recognized as inconsistency explanation of M iff F is unsatisfiable. Therefore the problem MCSE in an MCS with $\mathcal{CC}(M) \geq \mathbf{P}$ is hard for $\mathbf{coNP}$.

($\mathbf{co}$-$\mathcal{CC}(M)$-*hardness*) We reuse the MCSEQ hardness proof where we reduced an instance $I = (H_a, S_a)$, C_a to an MCS $M_I = (C_{a'}, C_{check})$. If I is a 'yes' instance, then M_I is consistent so no inconsistency explanation exists. If I is a 'no' instance, an inconsistency explanation of M_I is $(\{r_{en}\}, \{r_\gamma\}) \in E^{\pm}(M_I)$. Therefore the problem MCSE in an MCS M with lower context complexity $\mathcal{CC}(M) \geq C$ is hard for $\mathbf{co}$-C if C is a class with complete problems that is closed under conjunction and projection. □

Proof of Lemma 3. We write $(A_1, A_2) \subset (B_1, B_2)$ iff both, $(A_1, A_2) \subseteq (B_1, B_2)$ and $(A_1, A_2) \neq (B_1, B_2)$.

($\Rightarrow$) Assume $Q = (Q_1, Q_2)$ is a minimal explanation. Contrary to the lemma, assume there exists another explanation Q', such that $Q' = (Q_1, Q_2 \setminus \{r\})$ with $r \in Q_2$ or $Q' = (Q_1 \setminus \{r\}, Q_2)$ with $r \in Q_1$. Then $Q' \subset Q$, therefore Q is not minimal, contradicting the assumption.

($\Leftarrow$) Assume an explanation $Q = (Q_1, Q_2)$, and no pair $(Q_1, Q_2 \setminus \{r\})$ with $r \in Q_2$ or $(Q_1 \setminus \{r\}, Q_2)$ with $r \in Q_1$ is an explanation. Contrary to the Lemma, assume another explanation $P = (P_1, P_2)$ exists with $P \subset Q$. By $P \subset Q$, either a) $P_1 \subset Q_1$ and $P_2 \subseteq Q_2$ or b) $P_1 \subseteq Q_1$ and $P_2 \subset Q_2$. For a) we create $T' = (Q_1 \setminus \{r\}, Q_2)$ for some $r \in Q_1 \setminus P_1$. Then $P \subseteq T' \subset Q$. Due to Corollary 1, T' is an explanation, contradicting the initial assumption. The case b) is similar. □

Proof of Proposition 13. (*Membership*) We can decide $(E_1, E_2) \in E^{\pm}_m(M)$ by using Lemma 3, i.e., we decide (1) whether $(E_1, E_2) \in E^{\pm}(M)$, and (2) whether all of $(E_1, E_2 \setminus \{r \mid r \in E_2\}) \notin E^{\pm}(M)$ and $(E_1 \setminus \{r \mid r \in E_1, E_2\}) \notin E^{\pm}(M)$ are true. Note that the number of $E^{\pm}$-checks in (2) is linear in the size of the instance, hence we obtain the following membership results: if the upper context complexity $\mathcal{CC}(M)$ is a class with complete problems that is closed under conjunction and projection, deciding (1) is in **co**-$\mathcal{CC}(M)$ and deciding (2) is in $\mathcal{CC}(M)$, therefore MCSE$_m$ is in $\mathcal{D}(\mathcal{CC}(M))$. For upper context complexity $\mathcal{CC}(M) = \mathbf{P}$ (which is not closed under projection) deciding (1) is in **coNP** and deciding (2) is in **NP** and therefore MCSE$_m$ is in $\mathbf{D^P_1}$.

($\mathbf{D^P}$-*hardness* for $\mathcal{CC}(M) = \mathbf{P}$) We use the same topology as for the MCSD$_m$ hardness proof, i.e., the complete topology shown in Fig. 3a. We reduce two 3-SAT instances F and G on variables $\mathcal{X}$ and $\mathcal{Y}$, respectively, to minimal explanation recognition on MCS $M = (C_{gen_V}, C_{eval_F}, C_{gen_U}, C_{eval_G}, C_{check})$. Again, C_{gen_U} and C_{eval_F} provide **NP**-hardness for satisfiability of F, while C_{gen_V} and C_{eval_G} provide **coNP**-hardness for unsatisfiability of G. All contexts except for C_{check} are constructed from F and G exactly as in the MCSD$_m$ hardness proof. Wlog. we assume that F is not valid. The bridge rules in M are as follows:

$$r_{u,i}: \quad (eval_F : x_i) \leftarrow (gen_U : x_i). \quad \forall i : 1 \le i \le |\mathcal{X}|. \tag{C.22}$$

$$r_{v,j}: \quad (eval_G : y_j) \leftarrow (gen_V : y_j). \quad \forall j : 1 \le j \le |\mathcal{Y}|. \tag{C.23}$$

$$r_\alpha: \quad (check : sat\_or\_nsat_F) \leftarrow (eval_F : sat). \tag{C.24}$$

$$r_\beta: \quad (check : sat\_or\_nsat_F) \leftarrow \mathbf{not}\ (eval_F : sat). \tag{C.25}$$

$$r_\gamma: \quad (check : nsat_G) \leftarrow \mathbf{not}\ (eval_G : sat). \tag{C.26}$$

Context C_{check} has the following knowledge base kb_{check}:

$$\bot \leftarrow sat\_or\_nsat_F,\ \text{not}\ nsat_G. \tag{C.27}$$

If G is satisfiable, M is consistent, so $E^{\pm}_m(M) = \emptyset$. If F and G are unsatisfiable, the belief *sat* is never accepted at C_{eval_F}; therefore the bridge rule r_β is sufficient for creating inconsistency in M (i.e., $M[\{r_\beta\}] \models \bot$) and forcing r_γ to become applicable is the only way to restore consistency. Therefore if F and G are unsatisfiable, $(\{r_\beta\}, \{r_\gamma\}) \in E^{\pm}(M)$. If F is satisfiable and G is unsatisfiable, the belief *sat* may or may not be accepted at C_{eval_F}, depending on the input C_{eval_F} gets from C_{gen_U}. Therefore both bridge rules r_α and r_β are required for ensuring inconsistency in M, and they are also sufficient. Again, forcing r_γ to become applicable is the only way to restore consistency. Therefore if F is satisfiable and G is unsatisfiable, then $(\{r_\alpha, r_\beta\}, \{r_\gamma\}) \in E^{\pm}(M)$. Thus $(\{r_\alpha, r_\beta\}, \{r_\gamma\})$ is a minimal inconsistency explanation for M iff F is satisfiable and G is unsatisfiable. Note that if G is satisfiable, no explanations exist, while if F is unsatisfiable, the above explanation exists but is no longer minimal. Therefore recognizing a minimal inconsistency explanation in an MCS with $\mathcal{CC}(M) = \mathbf{P}$ is hard for $\mathbf{D^P}$.

($\mathbf{D}(\mathcal{CC}(M))$-*hardness*) We use the same topology as for the MCSD$_m$ hardness proof, i.e., the complete topology shown in Fig. 3b. We also use a very similar reduction. The only change is in the checking context C_{check}. We reduce two context complexity check instances (H_a, S_a), C_a with IN_a, OUT_a and (H_b, S_b), C_b with IN_b, OUT_b to an MCS $M = (C_{a'}, C_{b'}, C_{check})$. The bridge rules br_{check} are as follows.

$$r_\alpha: \quad (check : equal_{S'_a}) \leftarrow l_1, \dots, l_j, \dots, l_{|OUT_a|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_a \wedge s_j \in S_a \\ \mathbf{not}\ s_j & \text{if } s_j \in OUT_a \wedge s_j \notin S_a. \end{cases} \tag{C.28}$$

$$r_\beta: \quad (check : make\_inc) \leftarrow l_1, \dots, l_j, \dots, l_{|OUT_a|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_a \wedge s_j \in S_a \\ \mathbf{not}\ s_j & \text{if } s_j \in OUT_a \wedge s_j \notin S_a. \end{cases} \tag{C.29}$$

$$r_\gamma: \quad (check : equal_{S'_b}) \leftarrow l_1, \dots, l_j, \dots, l_{|OUT_b|}.,$$
$$\text{where } l_j \text{ is } \begin{cases} s_j & \text{if } s_j \in OUT_b \wedge s_j \in S_b \\ \mathbf{not}\ s_j & \text{if } s_j \in OUT_b \wedge s_j \notin S_b. \end{cases} \tag{C.30}$$

$$r_{en}: \quad (check : en) \leftarrow. \tag{C.31}$$

Note that r_α and r_β have the same body but different heads, moreover only r_β differs from the MCSD$_m$-reduction. The knowledge base kb_{check} is as follows:

$$n\_equal_a \leftarrow \text{not}\ equal_{S'_a}. \tag{C.32}$$

$$n\_equal_a \leftarrow make\_inc. \tag{C.33}$$

$$\bot \leftarrow en, n\_equal_a,\ \text{not}\ equal_{S'_b}. \tag{C.34}$$

The bridge rule r_{en} ensures that C_{check} fulfills our assumption that a context without input is consistent. Wlog. we assume that C_a and C_b accept some belief set given input H_a and H_b, respectively. The bridge rule r_α adds $equal_{S'_a}$ to C_{check} iff the first instance (H_a, S_a), C_a is a 'yes' instance. Under the same condition, r_β adds *make_inc*. The bridge rule r_γ adds $equal_{S'_b}$ to C_{check} iff the second instance (H_b, S_b), C_b is a 'yes' instance. In that case, M is consistent, i.e., $E^{\pm}_m(M) = \emptyset$, because r_γ becomes applicable and this deactivates constraint (C.34) such that C_{check} can no longer become inconsistent. If both instances are 'no' instances, M is inconsistent and for explaining this inconsistency it is sufficient to have r_{en} present and the heads of the bridge rules r_α and r_γ absent. Therefore, in that case, $(\{r_{en}\}, \{r_\alpha, r_\gamma\})$ is a minimal inconsistency explanation for M. Finally, if (H_a, S_a), C_a is a 'yes' instance and (H_b, S_b), C_b is a 'no' instance, M is inconsistent and for this inconsistency it is sufficient to have r_{en} and r_β present and heads of bridge rules r_α and r_γ absent, so $(\{r_{en}, r_\beta\}, \{r_\alpha, r_\gamma\}) \in E^{\pm}_m(M)$. Therefore $(\{r_{en}, r_\beta\}, \{r_\alpha, r_\gamma\}) \in E^{\pm}_m(M)$ iff the first instance is a 'yes' instance and the second instance is a 'no' instance. Note that if the second instance is a 'yes' instance, no explanations exist, while if the first instance is a 'no' instance, the above explanation exists but is no longer minimal. Therefore we have established that MCSE$_m$ is hard for $\mathbf{D}(C)$ where $\mathcal{CC}(M) = C$ if C is a class with complete problems that is closed under conjunction and projection. □

Semantics of HEX*-programs* The (ordinary) *Herbrand base* HB^o_P of a HEX-program P is the set of all ordinary atoms $p(c_1, \ldots, c_n)$ occurring in P. An *interpretation* I of P is any subset $I \subseteq HB^o_P$; I satisfies (is a *model* of)

- an atom α, denoted $I \models \alpha$, if $\alpha \in I$ for an ordinary atom α, or if $f_{\&g}(I, \vec{v}, \vec{w}) = 1$ in the case where $\alpha = \&g[\vec{v}](\vec{w})$ and $F_{\&g} : 2^{HB^o_P} \times \mathcal{C}^n \times \mathcal{C}^m \to \{0, 1\}$ is a (fixed) $(|\vec{v}|+|\vec{w}|+1)$-ary Boolean function associated with $\&g$;
- a rule r of form (2) $(I \models r)$, if either $I \models \alpha_i$ for some α_i, or $I \models \beta_j$ for some $j \in \{m+1, \ldots, n\}$, or $I \not\models \beta_i$ for some $i \in \{1, \ldots, m\}$;
- a program P $(I \models P)$, iff $I \models r$ for all $r \in P$.

The *FLP-reduct* [45] of a program P w.r.t. an interpretation I is the set $fP^I \subseteq P$ of all rules r of form (2) in P such that $I \models \beta_i$ for all $i \in \{1, \ldots, m\}$ and $I \not\models \beta_j$ for all $j \in \{m+1, \ldots, n\}$, i.e., I satisfies the body of (2). Then I is an *answer set of* P iff I is a $\subseteq$-minimal model of fP^I. We denote by $\mathcal{AS}(P)$ the collection of all answer sets of P. For P without external atoms, this coincides with answer sets as in [52], for a discussion on the relation between FLP-reduct and GL-reduct see [45].

For proving the correctness of our HEX encodings, we use some lemmas.

Lemma 7. *Let $P = R \cup C$ be a* HEX *program consisting of an ordinary* HEX*-program R and a set of constraints C which contain external atoms. Then for every $I \in \mathcal{AS}(P)$ it holds that $I \in \mathcal{AS}(R)$ and I does not satisfy the body of any constraint in C.*

Proof. From $I \in \mathcal{AS}(P)$ we know that $I \models P$ and therefore $I \models R$ and $I \models C$. From the latter we infer that I does not satisfy the body of any constraint in C (i.e., the second claim). Thus the reduct fP^I does not contain any constraint from C. Hence $fP^I = fR^I$ and I is a minimal model of fR^I as it is a minimal model of fP^I. □

Lemma 8. *Let P be a* HEX *program, and let $I \in \mathcal{AS}(P)$ be an answer set of P. Then for every atom $a \in I$ it holds that there is a rule $r \in P$ of form* (2) *with $a \in \{a_1, \ldots, a_k\}$ and I satisfies the body of r.*

Proof. Assume towards a contradiction that $I \in \mathcal{AS}(P)$, $a \in I$ and no rule $r \in P$ is such that a is in the head of r and I satisfies the body of r. Due to the latter assumption, no rule that contains a in the head is contained in fP^I. Since I is an answer set of P, $I \models fP^I$, therefore the bodies of all rules in fP^I are satisfied by I. Hence every rule in fP^I has a nonempty intersection of its head with I (otherwise $I \not\models fP^I$). Because no rule in fP^I contains a in the head, it follows that $I \setminus \{a\} \models fP^I$, therefore I is no minimal model of fP^I and no answer set, which is a contradiction. □

Proof of Theorem 4. (i) Given $I \in \mathcal{AS}(P^D_p(M))$, due to Lemma 7 we have (a) $I \in \mathcal{AS}(R)$ where R contains rules (3), (4), (5), and (6); and (b) no constraint (7) has a satisfied body. In R, only rules (3) have d_1 and d_2 atoms in the head, therefore $(D_{I,1}, D_{I,2})$ is such that $D_{I,1}, D_{I,2} \subseteq br(M)$. In R, only rules (4) have $pres_i$ and abs_i atoms in the head, therefore $A_i(I) \subseteq OUT_i$ for each context $C_i \in c(M)$. Hence $\mathcal{A}(I)$ is an output-projected belief state of M. Due to Lemma 8, I contains $b_i(s)$ iff at least one of the following is true: $d_2(r) \in I$ and accordingly $r \in D_{I,2}$, or there is at least one bridge rule $r \in br(M)$ such that $d_1(r) \notin I$ and in the corresponding rule (5) we have that for all i, $1 \le i \le j$, $pres_{c_i}(p_i) \in I$, and for all l, $j < l \le m$, $pres_{c_l}(p_l) \notin I$; this holds iff $r \notin D_{I,1}$ and $r \in app(br_i(M), \mathcal{A}(I))$, which holds iff $r \in app(br_i(M) \setminus D_{I,1}, \mathcal{A}(I))$. Therefore, for each context $C_i \in c(M)$ we have $B_i(I) = \{\varphi(r) \mid r \in app(br_i(M) \setminus D_{I,1}, \mathcal{A}(I))\} \cup \{\varphi(r) \mid r \in D_{I,2}\}$. The condition-free bridge rules are always applicable, therefore $B_i(I) = \{\varphi(r) \mid r \in app(br_i(M[br(M) \setminus D_{I,1} \cup cf(D_{I,2})]), \mathcal{A}(I))\}$. Note that in this expression, first all bridge rules of M are modified using $D_{I,1}$ and $D_{I,2}$, then the bridge rules of context C_i of the result are extracted using $br_i(\cdot)$. From (b) we know that for every context $C_i \in c(M)$, the external atom in (7) returns false, therefore $A_i(I) \in \mathbf{ACC}_i(kb_i \cup B_i(I))|_{OUT_i}$ for every $C_i \in c(M)$. Substituting $B_i(I)$ we obtain $A_i(I) \in \mathbf{ACC}_i(kb_i \cup \{\varphi(r) \mid r \in app(br_i(M[br(M) \setminus D_{I,1} \cup cf(D_{I,2})]), \mathcal{A}(I))\})|_{OUT_i}$. Therefore $\mathcal{A}(I)$ is an output-projected equilibrium of MCS $M[br(M) \setminus D_{I,1} \cup cf(D_{I,2})]$ and it holds that $(D_{I,1}, D_{I,2}) \in D^{\pm}(M)$.

(ii) Given a diagnosis $(D_1, D_2) \in D^{\pm}(M)$ and given an output-projected equilibrium $T = (T_1, \dots, T_n)$ of $M' = M[br(M) \setminus D_1 \cup cf(D_2)]$ we assemble the interpretation

$$I = \{d_1(r) \mid r \in D_1\} \cup \{d_2(r) \mid r \in D_2\} \cup \{um(r) \mid r \notin (D_1 \cup D_2)\} \cup$$
$$\{a_i(p) \mid p \in T_i\} \cup \{\bar{a}_i(p) \mid p \in OUT_i \setminus T_i\} \cup \{b_i(s) \mid s \in H_i\}$$

where $H_i = app(br_i(M'), T)$. Since T is an output-projected equilibrium, I satisfies constraints (7), therefore they are not part of the reduct $fP_p^D(M)^I$. By construction of I, those rules in (5) where $r \in D_1$ or r is not applicable in $\mathcal{A}(I)$ have an unsatisfied rule body, so these rules are not part of the reduct. Those rules in (6) where $r \in D_2$ have a satisfied rule body, so these rules are always part of the reduct. Other rules in (5) or (6) are satisfied by I as their body is not satisfied. For each applicable bridge rule r, the according head atom $b_i(s)$ is part of I, and $P_p^D(M)$ contains no cyclic dependencies between rules (hence neither does the reduct $fP_p^D(M)^I$). Therefore I is a minimal model of rules (5) and (6) in the reduct. Rules (3) and (4) are contained in the reduct, and I by construction is a minimal model of these rules. Therefore, I is a model of $P_p^D(M)$ and a minimal model of $fP_p^D(M)^I$, hence $I \in \mathcal{AS}(P_p^D(M))$. □

For the following proofs we assume $M = (C_1, \dots, C_n)$ to be an arbitrary but fixed MCS and $P_P^E(M)$ to be the explanation encoding for M.

Given a HEX rule r of form (2), we write $B_{\mathrm{HEX}}(r) = \{\beta_1, \dots, \beta_n\}$ and $H_{\mathrm{HEX}}(r) = \{\alpha_1, \dots, \alpha_k\}$ to denote body and head of r respectively. For an interpretation I and a HEX rule r, we write $I \models B_{\mathrm{HEX}}(r)$ iff $I \models \beta_i$ for all $i \in \{1, \dots, m\}$ and $I \not\models \beta_j$ for all $j \in \{m+1, \dots, n\}$. Similarly, we write $I \models H_{\mathrm{HEX}}(r)$ iff $I \models \alpha_i$ for some $i \in \{1, \dots, k\}$.

For referring to a specific rule of $P_P^E(M)$, we write $tr_N(v_1, \dots, v_\ell)$ where N is the rule of form (N) instantiated with $v_1, \dots, v_\ell$. We denote by $TR_n(M)$ the set of all instantiations of a rule w.r.t. an MCS M. For example, let $r_7 \in br(M)$, then $tr_{10}(r_7)$ denotes the HEX rule $r1(r_7) \leftarrow e1(r_7).$, while $TR_{10}(M) = \{tr_{10}(r) \mid r \in br(M)\}$. For brevity, we write only those values necessary to identify the instantiation, e.g., for rules of form (15) we write $tr_{15}(r)$ where $r \in br(M)$; for a rule of form (23), we write $tr_{23}(i, b)$ where $(i : b)$ is the head of some $r \in br(M)$.

We say an interpretation I *consistently encodes* an explanation candidate $E = (E_1, E_2)$ where $E_1 = \{r \in br(M) \mid e1(r) \in I\}$, $E_2 = \{r \in br(M) \mid e2(r) \in I\}$, for all $r \in br(M)$: (i) $e1(r) \in I$ iff $ne1(r) \notin I$, and (ii) $e2(r) \in I$ iff $ne2(r) \notin I$.

Lemma 9. *Every answer set I of $P_P^E(M)$ consistently encodes an explanation candidate.*

Proof. Let I be an answer set of $P_P^E(M)$. Then, by definition I must be a minimal model of $fP_P^E(M)^I$. Assume for contradiction that I does not consistently encode an explanation candidate. Then, for some $r \in br(M)$ one of the following cases holds.

(i) $e1(r) \in I$ and $ne1(r) \in I$: Consider $I' = I \setminus \{e1(r)\}$. For all $tr \in fP_P^E(M)^I$ with $e1(r) \notin H_{\mathrm{HEX}}(tr)$ it holds that $I' \models tr$ since $I \models tr$. There is only one rule tr' such that $e1(r) \in H_{\mathrm{HEX}}(tr')$, namely $tr' = tr_8(r)$. Since $ne1(r) \in I'$ and $ne1(r) \in H_{\mathrm{HEX}}(tr_8(r))$ it holds that $I' \models tr$, hence $I' \models fP_P^E(M)^I$. Since $I' \subset I$ this contradicts that I is a minimal model of $fP_P^E(M)^I$.
(ii) $e1(r) \notin I$ and $ne1(r) \notin I$. Since $B_{\mathrm{HEX}}(tr_8(r)) = \emptyset$, it holds that $tr_8(r) \in fP_P^E(M)^I$ while $I \not\models H_{\mathrm{HEX}}(tr_8(r))$. Hence, in contradiction to the assumption, it holds that $I \not\models fP_P^E(M)^I$.
(iii) $e2(r) \in I$ and $ne2(r) \in I$: This is similar to case (i), just replace $e1$ by $e2$ and $tr_8(r)$ by $tr_9(r)$.
(iv) $e2(r) \notin I$ and $ne2(r) \notin I$: This is similar to case (ii), just replace $e1$ by $e2$ and $tr_8(r)$ by $tr_9(r)$.

Since each case yields a contradiction, it follows that I consistently encodes an explanation candidate. □

Lemma 10. *If I is an answer set for $P_P^E(M)$ and $E = (E_1, E_2)$ is the explanation candidate consistently encoded by I, then $fP_P^E(M)^I$ exactly contains*

1. $TR_{14}(M) \cup \dots \cup TR_{31}(M)$.
2. $\{tr_{10}(r) \mid r \in E_1\} \cup \{tr_{11}(r) \mid r \in br(M) \setminus E_1\} \cup \{tr_{12}(r) \mid r \in E_2\} \cup \{tr_{13}(r) \mid r \in br(M) \setminus E_2\}$.

Proof. Let I be an answer set for $P_P^E(M)$ encoding an explanation candidate $E = (E_1, E_2)$.

1. By the constraint rule (32), it holds that $spoil \in I$, thus rules $TR_{28}(M) \cup \dots \cup TR_{31}(M)$ are in $fP_P^E(M)^I$. Let $tr \in TR_{28}(M) \cup \dots \cup TR_{31}(M)$, then it holds that $I \models B_{\mathrm{HEX}}(tr)$, hence it follows that $I \models H_{\mathrm{HEX}}(tr)$. Therefore, $I \models B_{\mathrm{HEX}}(tr')$ and $tr' \in fP_P^E(M)^I$, where $tr' \in TR_{14}(M) \cup \dots \cup TR_{27}(M)$. Specifically, it holds for $tr_{24}(i)$, where $1 \le i \le n$, that $I \models B_{\mathrm{HEX}}(tr_{24}(i))$, because $spoil \in I$ which implies that $f_{\&con\_out_i'}(I, spoil, pres_i, in_i) = 0$.

2. Let $r \in E_1$. Then $e1(r) \in I$ and $ne1(r) \notin I$ since I consistently encodes E. Thus, $I \models B_{\mathrm{HEX}}(tr_{10}(r))$, therefore $tr_{10}(r) \in fP_P^E(M)^I$. Furthermore, $I \not\models B_{\mathrm{HEX}}(tr_{11}(r))$, hence $tr_{11}(r) \notin fP_P^E(M)^I$.

Let $r \in br(M) \setminus E_1$. Then $e1(r) \notin I$ and $ne1(r) \in I$ since I consistently encodes E. Thus, $I \models B_{\mathrm{HEX}}(tr_{11}(r))$, therefore $tr_{11}(r) \in fP_P^E(M)^I$. Furthermore, $I \not\models B_{\mathrm{HEX}}(tr_{10}(r))$, hence $tr_{10}(r) \notin fP_P^E(M)^I$.

The remaining cases for E_2 are analogous. □

Definition 17. An interpretation I of $P_P^E(M)$ is called *contradiction-free* (regarding *r1*, *nr1*, *r2*, *nr2*, *pres*$_i$, *abs*$_i$) if and only if the following conditions hold:

$$\begin{array}{llll} r1(r) \in I & \textit{iff} & nr1(r) \notin I & \textit{for every } r \in br(M) \\ r2(r) \in I & \textit{iff} & nr2(r) \notin I & \textit{for every } r \in br(M) \\ pres_i(a) \in I & \textit{iff} & abs_i(a) \notin I & \textit{for every } a \in OUT_i,\ 1 \le i \le n \end{array}$$

We say that a contradiction-free interpretation I *consistently encodes* a belief state $S = (S_1, \ldots, S_n)$ and a pair (R_1, R_2) of sets of bridge rules such that: $a \in S_i$ iff $pres_i(a) \in I$, $r \in R_1$ iff $r1(r) \in I$, and $r \in R_2$ iff $r2(r) \in I$.

Notice that rule (32) and Lemma 10 ensure that no answer set I of $P_P^E(M)$ is contradiction-free, because it holds that *spoil* $\in I$ and the rules of $TR_{28}(M) \cup \cdots \cup TR_{31}(M)$ ensure the saturation of I. The notion, however, is useful for reasoning about (minimal) models of $fP_P^E(M)^I$.

$P_P^E(M)$ guarantees that a contradiction-free interpretation I that encodes a belief state S and a pair (R_1, R_2) of sets of bridge rules also contains a representation of the set of heads of bridge rules applicable under S and (R_1, R_2), as the following lemma shows.

Lemma 11. *Let I be a contradiction-free interpretation that encodes the belief state $S = (S_1, \ldots, S_n)$ of M, and let (R_1, R_2) such that $R_1, R_2 \subseteq br(M)$. If I is a minimal model of $P \subseteq P_P^E(M)$ such that $TR_{15}(M) \cup \cdots \cup TR_{23}(M)$ is a subset of P, then $\{b \in IN_i \mid in_i(b) \in I\} = \{\varphi(r) \mid r \in app(br_i(M[R_1 \cup cf(R_2)]), S)\}$ for every $1 \le i \le n$.*

Proof. (⊆): Let $b \in \{b \in IN_i \mid in_i(b) \in I\}$ and let $\{r_1, \ldots, r_k\} = [(i : b)]$ be the set of bridge rules of $M[R_1 \cup cf(R_2)]$ whose head is $(i : b)$. Since $I \models B_{\mathrm{HEX}}(tr_{23}(i, b))$, it must hold for some rule r_j with $1 \le j \le k$ that $r2(r_j) \in I$ or $body(r_j) \in I$.

In the former case it follows that $r_j \in R_2$ and thus $r_j \in app(br_i(M[R_1 \cup cf(R_2)]), S)$, hence $b \in \{\varphi(r) \mid r \in app(br_i(M[R_1 \cup cf(R_2)]), S)\}$.

In the latter case, $body(r_j) \in I$ together with rules $tr_{17}(r_j), \ldots, tr_{20}(r_j)$ implies that each literal in the body of r_j is satisfied by the belief state S. Furthermore, from $I \models tr_{16}(r_j)$ it follows that $r1(r_j) \in I$, hence $r_j \in R_1$. Therefore, $r_j \in app(br_i(M[R_1 \cup cf(R_2)]), S)$, hence $b \in \{\varphi(r_j) \mid app(br_i(M[R_1 \cup cf(R_2)]), S)\}$.

(⊇) Let $b \in app(br_i(M[R_1 \cup cf(R_2)]), S)\}$ and let $\{r_1, \ldots, r_k\} = [(i : b)]$ be the bridge rules in $M[R_1 \cup cf(R_2)]$ whose head is $(i : b)$. By definition of applicability, it must hold for some r_j with $1 \le j \le k$ that either $r_j \in R_2$ or $r_j \in R_1$ and the body of r_j is satisfied w.r.t. S. In the former case $r2(r_j) \in I$ and by $tr_{22}(r_j)$ it must hold that $in_i(b) \in I$, hence $b \in \{b \in IN_i \mid in_i(b) \in I\}$. In the latter case observe that $S \models r_j$ and as I consistently encodes S and (R_1, R_2), it holds that $I \models B_{\mathrm{HEX}}(tr_{15}(r_j))$. Therefore $in_i(b) \in I$, hence $b \in \{b \in IN_i \mid in_i(b) \in I\}$. □

Proof of Theorem 5. Recall the concept of a saturated ("spoiled") interpretation. An interpretation is saturated, if it is a superset of I_{spoil}, which is defined as follows:

$$\begin{aligned} I_{spoil} = {} & \bigl\{r1(r), nr1(r), r2(r), nr2(r), body(r) \bigm| r \in br(M)\bigr\} \cup \\ & \bigl\{in_i(b) \bigm| r \in br(M) \wedge C_h(r) = i \wedge \varphi(r) = b\bigr\} \cup \{spoil\} \cup \\ & \bigcup_{a \in OUT_i} \bigl\{pres_i(a), abs_i(a)\bigr\} \cup \bigcup_{b \in IN_i} \bigl\{in_i(b)\bigr\}. \end{aligned}$$

Soundness (⇐) Let I be an answer set of $P_P^E(M)$. Then by Lemma 9 I consistently encodes an explanation candidate $E = (E_1, E_2)$ where $E_1 = \{r \in br(M) \mid e1(r) \in I\}$ and $E_2 = \{r \in br(M) \mid e2(r) \in I\}$. We show that E is an explanation of M.

Since I is an answer set of $P_P^E(M)$, it is a minimal model of $fP_P^E(M)^I$ and by Lemma 10 all rules of $TR_{14}(M) \cup \cdots \cup TR_{31}(M)$ are in $fP_P^E(M)^I$, so I must be a minimal model of those rules. By rule (32) it follows that *spoil* $\in I$, therefore for each $tr \in TR_{28}(M) \cup \cdots \cup TR_{31}(M)$ it holds that $H_{\mathrm{HEX}}(tr) \in I$ since $I \models B_{\mathrm{HEX}}(tr)$. Therefore, $I_{spoil} \subseteq I$.

Towards a contradiction, assume that E is not an explanation. Then, there exists $(R_1, R_2) \in Rg(E)$ such that $M' \not\models \bot$ holds for $M' = M[R_1 \cup cf(R_2)]$, i.e., M' has an equilibrium $S = (S_1, \ldots, S_n)$.

Consider the interpretation $I_{S,(R_1,R_2)}$ corresponding to S and (R_1, R_2), i.e., I' is a contradiction-free interpretation regarding *r1*, *nr1*, *r2*, *nr2*, *pres*$_i$, *abs*$_i$ that consistently encodes S and (R_1, R_2). Let $I_{S,(R_1,R_2),E} = I_{S,(R_1,R_2)} \cup \{e1(r) \in I\} \cup \{ne1(r) \in I\} \cup \{e2(r) \in I\} \cup \{ne2(r) \in I\}$ be the interpretation consistently encoding E, S, and (R_1, R_2). Finally, let $I_{app} = \{in_i(b) \mid b \in app(br_i(M'), S)\} \cup \{body(r) \mid r \in R_1 \wedge S \models r\}$ correspond to the set of bridge rule heads and bodies applicable under S. Combining them, we obtain an interpretation $I' = I_{S,(R_1,R_2),E} \cup I_{app}$. Note that $I' \subset I$, since I is saturated and both I and I' consistently encode E.

As we show in the following, it holds that $I' \models fP_P^E(M)^I$:

- For every $tr \in TR_8(M) \cup TR_9(M)$ it holds that $I' \models tr$ since $I \models tr$ and I agrees with I' on atoms $e1, ne1, e2$, and $ne2$.
- For every $tr \in TR_{10}(M) \cup \cdots \cup TR_{13}(M)$ it holds that $I' \models tr$ since $(R_1, R_2) \in Rg(E)$ and I' consistently encodes (R_1, R_2).
- For every $tr \in TR_{14}(M)$ it holds that $I' \models tr$ since I' consistently encodes S.
- For every $r \in br(M)$ it holds that $I' \models tr_{15}(r)$ since $I_{app} \subseteq I'$ and I_{app} is defined such that $S \mid\!\!\rightsquigarrow r$ and $r \in R_1$ implies that $body(r) \in I'$.
- For every $r \in br(M)$ it holds that $I' \models tr_{16}(r)$ since $body(r) \in I'$ implies $r \in R_1$, hence by I' encoding (R_1, R_2) it follows that $r1(r) \in I'$.
- For every $r \in br(M)$ it holds that $I' \models tr_{17}(r), \ldots I' \models tr_{20}(r)$, because $body(r) \in I'$ only if $S \models r$, hence by I' encoding S the following hold: $H_{\mathrm{HEX}}(tr_{17}(r)) \in I', \ldots, H_{\mathrm{HEX}}(tr_{20}(r)) \in I'$.
- For every $r \in br(M)$ it holds that $I' \models tr_{21}(r)$, respectively $I' \models tr_{22}(r)$ since $S \models r$ and $r \in R_1$, respectively $r \in R_2$, implies that $r \in app(br_i(M'), S)$, hence $in_i(b) \in I'$ where $i \in ci(M)$ and $\varphi(r) = b$.
- For every head $(i : b)$ of a bridge rule it holds that $I' \models tr_{23}(i, b)$, because: if $in_i(b) \in I'$ for some $i \in ci(M)$, then by definition of I' there exists $r \in app(br_i(M'), S)$ such that one of the following holds:
  - $S \models r$ and $r \in R_1$, which implies that $body(r) \in I'$.
  - $r \in R_2$ and therefore $r2(r) \in I'$.
- $I' \models tr_{24}(i)$ holds for all $1 \leq i \leq n$: By definition of I_{app}, it holds that $\{b \mid in_i(b) \in I'\} = app(br_i(M'), S)$ and since I' encodes S, it also it holds that $\{a \mid pres_i(a)\} = S_i$. By assumption S is an equilibrium of M', hence $S_i \in \mathbf{ACC}_i(app(br_i(M'), S))$. Therefore, $f_{\&con\_out'_i}(I', pres_i, in_i) = 1$ and $I' \not\models B_{\mathrm{HEX}}(tr_{24}(i))$.
- For every $tr \in TR_{25}(M) \cup \cdots \cup TR_{27}(M)$ it holds that $I' \models tr$ since I' is conflict-free and $I' \not\models B_{\mathrm{HEX}}(tr)$.
- For every $tr \in TR_{28}(M) \cup \cdots \cup TR_{31}(M)$ it holds that $I' \models tr$ since $spoil \notin I'$.
- Rule (32): is not in the reduct $fP^E_P(M)^I$, hence it needs not be satisfied by I'.

Therefore, all rules of $fP^E_P(M)^I$ are satisfied and it follows that I' is a model of $fP^E_P(M)^I$. Since $I' \subset I$, I is not a minimal model of $fP^E_P(M)^I$, which contradicts that I is an answer set of $P^E_P(M)$. This proves that $E \in E^{\pm}(M)$.

Completeness ($\Rightarrow$) Let $E = (E_1, E_2) \in E^{\pm}(M)$. Then for every $(R_1, R_2) \in Rg(E)$ it holds that $M' \models \bot$ where $M' = M[R_1 \cup cf(R_2)]$, i.e., for every belief state $S = (S_1, \ldots, S_n)$ exists some $1 \leq i \leq n$ such that $S_i \notin \mathbf{ACC}_i(app(br_i(M'), S))$.

We show that $I_E = \{e1(r) \mid r \in E_1\} \cup \{ne1(r) \mid r \in br(M) \setminus E_1\} \cup \{e2(r) \mid r \in E_2\} \cup \{ne2(r) \mid r \in br(M) \setminus E_2\} \cup I_{spoil}$ is an answer set of $P^E_P(M)$.

Since I_E contains respective instances for $e1, ne1, e2$, and $ne2$, $fP^E_P(M)^{I_E}$ contains the following rules: $tr_{10}(r)$ such that $r \in E_1$; $tr_{11}(r)$ such that $r \in br(M) \setminus E_1$; $tr_{12}(r)$ such that $r \in E_2$; and $tr_{13}(r)$ such that $r \in br(M) \setminus E_2$. Furthermore, because I_E contains I_{spoil}, $fP^E_P(M)^{I_E}$ contains all rules in $TR_8(M) \cup TR_9(M) \cup TR_{14}(M) \cup \cdots \cup TR_{31}(M)$. Given that $I_{spoil} \subset I_E$, it is easy to see that I_E is a model of $fP^E_P(M)^{I_E}$. It remains to show that I_E is a $\subseteq$-minimal model of $fP^E_P(M)^{I_E}$.

Assume for contradiction that some $I' \subset I_E$ is a model of $fP^E_P(M)^{I_E}$. Observe that I_E consistently encodes E by definition. Since it must hold that $I' \models tr$ where $tr \in TR_8(M) \cup TR_9(M)$ and $I' \subset I_E$, it follows that I' also consistently encodes E.

Since $fP^E_P(M)^{I_E}$ contains rules $TR_{28}(M) \cup \cdots \cup TR_{31}(M)$ which must be satisfied by I' either $spoil \notin I'$ or all respective heads are in I', which means that I' is saturated. The latter implies that $I' = I_E$, which contradicts the assumption $I' \subset I_E$, it follows that $spoil \notin I'$. This requires that $I' \not\models B_{\mathrm{HEX}}(tr)$ where $tr \in TR_{24}(M) \cup TR_{25}(M) \cup TR_{26}(M) \cup TR_{27}(M)$.

Since it holds that $I \not\models B_{\mathrm{HEX}}(tr_{24}(i))$ for all $1 \leq i \leq n$, there exists a contradiction-free guess regarding $r1, nr1, r2, nr2, pres_i, abs_i$ such that $f_{\&con\_out'_i}(I', pres_i, in_i) = 1$. Let $S = (S_1, \ldots, S_n)$ be the belief state consistently encoded by I' and let (R_1, R_2) be the pair of sets of bridge rules consistently encoded by I'. It holds that $(R_1, R_2) \in Rg(E)$, because $TR_{10}(M)$ and $TR_{12}(M)$ together with the fact that I' is contradiction-free ensure: $e1(r) \in I'$ implies $r1(r) \in I'$ and $r2(r) \in I'$ implies that $ne2(r) \in I'$. In other words, $R_1 \subseteq E_1$ and $R_2 \subseteq br(M) \setminus E_2$, hence $(R_1, R_2) \in Rg(E)$.

By Lemma 11, $\{b \in IN_i \mid in_i(b) \in I'\} = \{\varphi(r) \mid r \in app(br_i(M[R_1 \cup cf(R_2)]), S)\}$ for every $1 \leq i \leq n$, which implies that $S_i \in \mathbf{ACC}_i(\{\varphi(r) \mid r \in app(br_i(M[R_1 \cup cf(R_2)]), S)\})$; i.e., S is an equilibrium of $M[R_1 \cup cf(R_2)]$. Since $(R_1, R_2) \in Rg(E)$, this contradicts that E is an explanation of M. It follows that no $I' \subset I_E$ is a model of $fP^E_P(M)^{I_E}$. Hence I_E is an answer set of $P^E_P(M)$. $\square$

References

[1] M. Arenas, L.E. Bertossi, J. Chomicki, Answer sets for consistent query answering in inconsistent databases, Theory Pract. Log. Program. 3 (4–5) (2003) 393–424.

[2] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, Cambridge, UK, 2003.

[3] S.E.-D. Bairakdar, M. Dao-Tran, T. Eiter, M. Fink, T. Krennwallner, Decomposition of distributed nonmonotonic multi-context systems, in: T. Janhunen, I. Niemelä (Eds.), Logics in Artificial Intelligence – 12th European Conference, Proceedings, JELIA 2010, Helsinki, Finland, September 13–15, 2010, in: LNCS, vol. 6341, Springer, 2010, pp. 24–37.

[4] S.E.-D. Bairakdar, M. Dao-Tran, T. Eiter, M. Fink, T. Krennwallner, The DMCS solver for distributed nonmonotonic multi-context systems, in: European Conference on Logics in Artificial Intelligence (JELIA 2010), vol. 6341, 2010, pp. 352–355.

[5] M. Balduccini, M. Gelfond, Logic programs with consistency-restoring rules, in: International Symposium on Logical Formalization of Commonsense Reasoning, in: AAAI 2003 Spring Symposium Series, 2003, pp. 9–18.

[6] N.D. Belnap, A useful four-valued logic, in: G. Epstein, J.M. Dunn (Eds.), Modern Uses of Multiple-Valued Logic, Reidel Publishing Company, Boston, 1977, pp. 7–37.

[7] C. Berge, Hypergraphs, Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1989.
[8] P.A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, I. Zaihrayeu, Data management for peer-to-peer computing: a vision, in: Web DB, 2002, pp. 89–94.
[9] L. Bertossi, Consistent query answering in databases, SIGMOD Rec. 35 (2) (June 2006) 68–76.
[10] L.E. Bertossi, Database Repairing and Consistent Query Answering, in: Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2011.
[11] L.E. Bertossi, A. Hunter, T. Schaub (Eds.), Inconsistency Tolerance, LNCS, vol. 3300, Springer, 2005 [result from a Dagstuhl seminar].
[12] P. Besnard, T. Schaub, Signed systems for paraconsistent reasoning, J. Autom. Reason. 20 (1) (1998) 191–213.
[13] A. Bikakis, G. Antoniou, Distributed defeasible contextual reasoning in ambient computing, in: Ambient Intelligence, 2008, pp. 308–325.
[14] A. Bikakis, G. Antoniou, Defeasible contextual reasoning with arguments in ambient intelligence, IEEE Trans. Knowl. Data Eng. 22 (11) (2010) 1492–1506.
[15] A. Bikakis, G. Antoniou, P. Hassapis, Strategies for contextual reasoning with conflicts in ambient intelligence, Knowl. Inf. Syst. 27 (1) (2011) 45–84.
[16] A. Binas, S.A. McIlraith, Peer-to-peer query answering with inconsistent knowledge, in: Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning, Sydney, Australia, September 16–19, 2008, pp. 329–339.
[17] H.A. Blair, V.S. Subrahmanian, Paraconsistent logic programming, Theor. Comput. Sci. 68 (2) (1989) 135–154.
[18] M. Bögl, T. Eiter, M. Fink, P. Schüller, The MCS-IE system for explaining inconsistency in multi-context systems, in: T. Janhunen, I. Niemelä (Eds.), 12th European Conference on Logics in Artificial Intelligence, JELIA 2010, in: LNAI, Springer, September 2010, pp. 356–359.
[19] A. Borgida, L. Serafini, Distributed description logics: assimilating information from peer sources, J. Data Semant. I (2003) 153–184.
[20] M. Brain, M. Gebser, J. Pührer, T. Schaub, H. Tompits, S. Woltran, Debugging ASP programs by means of ASP, in: International Conference on Logic Programming and Nonmonotonic Reasoning, 2007, pp. 31–43.
[21] G. Brewka, T. Eiter, Equilibria in heterogeneous nonmonotonic multi-context systems, in: AAAI Conference on Artificial Intelligence, 2007, pp. 385–390.
[22] G. Brewka, T. Eiter, M. Fink, Nonmonotonic multi-context systems: a flexible approach for integrating heterogeneous knowledge sources, in: M. Balduccini, T.C. Son (Eds.), Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning, in: LNCS, vol. 6565, Springer, 2011, pp. 233–258.
[23] G. Brewka, T. Eiter, M. Fink, A. Weinzierl, Managed multi-context systems, in: T. Walsh (Ed.), IJCAI 2011, AAAI Press/IJCAI, 2011, pp. 786–791.
[24] G. Brewka, F. Roelofsen, L. Serafini, Contextual default reasoning, in: International Joint Conference on Artificial Intelligence (IJCAI), 2007, pp. 268–273.
[25] T. Bylander, D. Allemang, C. Tanner, J. Josephson, The computational complexity of abduction, Artif. Intell. 49 (1991) 25–60.
[26] D. Calvanese, G.D. Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Inconsistency tolerance in P2P data integration: an epistemic logic approach, Inf. Syst. 33 (4–5) (2008) 360–384.
[27] D. Calvanese, G.D. Giacomo, M. Lenzerini, R. Rosati, Logical foundations of peer-to-peer data integration, in: C. Beeri, A. Deutsch (Eds.), PODS, ACM, 2004, pp. 241–251.
[28] N. Choi, I.-Y. Song, H. Han, A survey on ontology mapping, SIGMOD Rec. 35 (September 2006) 34–41.
[29] E. Dantsin, T. Eiter, G. Gottlob, A. Voronkov, Complexity and expressive power of logic programming, ACM Comput. Surv. 33 (3) (2001) 374–425.
[30] M. Dao-Tran, T. Eiter, M. Fink, T. Krennwallner, Distributed nonmonotonic multi-context systems, in: F. Lin, U. Sattler, M. Truszczynski (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9–13, 2010, AAAI Press, 2010, pp. 60–70.
[31] A. Doan, A.Y. Halevy, Semantic integration research in the database community: a brief survey, AI Mag. 26 (1) (2005) 83–94.
[32] T. Eiter, G. Brewka, M. Dao-Tran, M. Fink, G. Ianni, T. Krennwallner, Combining nonmonotonic knowledge bases with external sources, in: Frontiers of Combining Systems (FroCoS), 2009, pp. 18–42.
[33] T. Eiter, M. Fink, G. Ianni, T. Krennwallner, P. Schüller, Pushing efficient evaluation of HEX programs by modular decomposition, in: J.P. Delgrande, W. Faber (Eds.), Logic Programming and Nonmonotonic Reasoning – 11th International Conference, Proceedings, LPNMR 2011, Vancouver, Canada, May 16–19, 2011, in: LNCS, vol. 6645, Springer, 2011, pp. 93–106.
[34] T. Eiter, M. Fink, T. Krennwallner, C. Redl, P. Schüller, Improving HEX-program evaluation based on unfounded sets, Tech. Rep. INFSYS RR-1843-12-08, Institut für Informationssysteme, TU Wien, 2012.
[35] T. Eiter, M. Fink, J. Moura, Paracoherent answer set programming, in: F. Lin, U. Sattler, M. Truszczynski (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9–13, 2010, AAAI Press, 2010, pp. 486–496.
[36] T. Eiter, M. Fink, P. Schüller, Approximations for explanations of inconsistency in partially known multi-context systems, in: J.P. Delgrande, W. Faber (Eds.), Logic Programming and Nonmonotonic Reasoning – 11th International Conference, Proceedings, LPNMR 2011, Vancouver, Canada, May 16–19, 2011, in: LNCS, vol. 6645, Springer, 2011, pp. 107–119.
[37] T. Eiter, M. Fink, P. Schüller, A. Weinzierl, Finding explanations of inconsistency in multi-context systems, in: F. Lin, U. Sattler, M. Truszczynski (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9–13, 2010, AAAI Press, 2010, pp. 329–339.
[38] T. Eiter, M. Fink, P. Schüller, A. Weinzierl, Finding explanations of inconsistency in nonmonotonic multi-context systems, Tech. Rep. INFSYS RR-1843-12-09, Institut für Informationssysteme, TU Wien, 2012.
[39] T. Eiter, M. Fink, A. Weinzierl, Preference-based inconsistency assessment in multi-context systems, in: T. Janhunen, I. Niemelä (Eds.), Logics in Artificial Intelligence – 12th European Conference, Proceedings, JELIA 2010, Helsinki, Finland, September 13–15, 2010, in: LNCS, vol. 6341, Springer, 2010, pp. 143–155.
[40] T. Eiter, G. Gottlob, On the computational cost of disjunctive logic programming: propositional case, Ann. Math. Artif. Intell. 15 (3–4) (1995) 289–323.
[41] T. Eiter, G. Ianni, R. Schindlauer, H. Tompits, A uniform integration of higher-order reasoning and external evaluations in answer-set programming, in: International Joint Conference on Artificial Intelligence, 2005, pp. 90–96.
[42] T. Eiter, G. Ianni, R. Schindlauer, H. Tompits, Effective integration of declarative rules with external evaluations for semantic-web reasoning, in: European Semantic Web Conference (ESWC), 2006, pp. 273–287.
[43] T. Eiter, A. Polleres, Transforming co-np checks to answer set computation by meta-interpretation, in: F. Buccafurri (Ed.), APPIA-GULP-PRODE, 2003, pp. 410–421.
[44] F. Ensan, W. Du, Aspects of inconsistency resolution in modular ontologies, in: Advances in Artificial Intelligence, 21st Conference of the Canadian Society for Computational Studies of Intelligence, Proceedings, Canadian AI 2008, Windsor, Canada, May 28–30, 2008, in: LNCS, vol. 5032, Springer, 2008, pp. 84–95.
[45] W. Faber, N. Leone, G. Pfeifer, Recursive aggregates in disjunctive logic programs: semantics and complexity, in: European Conference on Logics in Artificial Intelligence (JELIA), 2004, pp. 200–212.
[46] A. Finkelstein, D. Gabbay, A. Hunter, J. Kramer, B. Nuseibeh, Inconsistency handling in multiperspective specifications, IEEE Trans. Softw. Eng. 20 (1994) 569–578.
[47] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, M. Goedicke, Viewpoints: a framework for integrating multiple perspectives in system development, Int. J. Softw. Eng. Knowl. Eng. 2 (1) (1992) 31–58.
[48] D. Gabbay, A. Hunter, Making inconsistency respectable 1: a logical framework for inconsistency in reasoning, in: Foundations of Artificial Intelligence Research, in: LNCS, vol. 535, 1991, pp. 19–32.

[49] D.M. Gabbay, Labelled deductive systems: a position paper, in: J. Oikkonen, J. Väänänen (Eds.), Proc. of Logic Colloquium '90, Helsinki, Finland, 15–22 July 1990, in: Lecture Notes in Logic, vol. 2, Springer-Verlag, Berlin, 1993, pp. 66–88.
[50] D.M. Gabbay, A. Hunter, Making inconsistency respectable: part 2 – meta-level handling of inconsistency, in: ECSQARU, in: LNCS, vol. 747, Springer, 1993, pp. 129–136.
[51] M. Gebser, J. Pührer, T. Schaub, H. Tompits, A meta-programming technique for debugging answer-set programs, in: D. Fox, C.P. Gomes (Eds.), AAAI, AAAI Press, 2008, pp. 448–453.
[52] M. Gelfond, V. Lifschitz, Classical negation in logic programs and disjunctive databases, New Gener. Comput. 9 (3/4) (1991) 365–386.
[53] C. Ghidini, F. Giunchiglia, Local models semantics, or contextual reasoning=locality+compatibility, Artif. Intell. 127 (2) (2001) 221–259.
[54] F. Giunchiglia, Contextual reasoning, Epistemologia XVI (1993) 345–364.
[55] F. Giunchiglia, L. Serafini, Multilanguage hierarchical logics, or: how we can do without modal logics, Artif. Intell. 65 (1) (1994) 29–70.
[56] G. Gottlob, Complexity results for nonmonotonic logics, J. Log. Comput. 2 (1992) 397–425.
[57] P.W.P.J. Grefen, J. Widom, Integrity constraint checking in federated databases, in: CoopIS, 1996, pp. 38–47.
[58] J.Y. Halpern, Y. Moses, A guide to completeness and complexity for modal logics of knowledge and belief, Artif. Intell. 54 (3) (1992) 319–379.
[59] D. Heimbigner, D. McLeod, A federated architecture for information management, ACM Trans. Inf. Syst. 3 (3) (1985) 253–278.
[60] A. Hunter, Paraconsistent logics, in: Handbook of Defeasible Reasoning and Uncertain Information, Kluwer, 1996, pp. 11–36.
[61] A. Hunter, R. Summerton, Fusion rules for context-dependent aggregation of structured news reports, J. Appl. Non-Class. Log. (2004).
[62] K. Inoue, C. Sakama, Abductive framework for nonmonotonic theory change, in: International Joint Conference on Artificial Intelligence (IJCAI), 1995, pp. 204–210.
[63] A.C. Kakas, R.A. Kowalski, F. Toni, Abductive logic programming, J. Log. Comput. 2 (6) (1992) 719–770.
[64] S. Konieczny, R.P. Pérez, Logic based merging, J. Philos. Log. 40 (2) (2011) 239–270.
[65] D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, D.F. Savo, Inconsistency-tolerant semantics for description logics, in: P. Hitzler, T. Lukasiewicz (Eds.), RR, in: LNCS, vol. 6333, Springer, 2010, pp. 103–117.
[66] D. Lembo, M. Ruzzi, Consistent query answering over description logic ontologies, in: Proc. Conference on Web Reasoning and Rule Systems, in: LNCS, vol. 4524, Springer, 2007, pp. 194–208.
[67] M. Lenzerini, Data integration: a theoretical perspective, in: L. Popa, S. Abiteboul, P.G. Kolaitis (Eds.), PODS, ACM, 2002, pp. 233–246.
[68] N. Leone, G. Greco, G. Ianni, V. Lio, G. Terracina, T. Eiter, W. Faber, M. Fink, G. Gottlob, R. Rosati, D. Lembo, M. Lenzerini, M. Ruzzi, E. Kalka, B. Nowicki, W. Staniszkis, The INFOMIX system for advanced integration of incomplete and inconsistent data, in: SIGMOD, 2005, pp. 915–917.
[69] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, F. Scarcello, The dlv system for knowledge representation and reasoning, ACM Trans. Comput. Log. 7 (3) (2006) 499–562.
[70] N. Leone, R. Rosati, F. Scarcello, Enhancing answer set planning, Tech. Rep. DBAI-TR-2000-37, Institut für Informationssysteme, TU Wien, 2000.
[71] V. Lifschitz, H. Turner, Splitting a logic program, in: International Conference on Logic Programming (ICLP), 1994, pp. 23–37.
[72] J. Lloyd, Declarative error diagnosis, New Gener. Comput. 5 (1987) 133–154, http://dx.doi.org/10.1007/BF03037396.
[73] J. Lobo, C. Uzcátegui, Abductive change operators, Fundam. Inform. 27 (4) (1996) 385–411.
[74] J. Oetsch, J. Pührer, H. Tompits, Catching the ouroboros: on debugging non-ground answer-set programs, Theory Pract. Log. Program. 10 (4–6) (2010) 513–529.
[75] J. Oetsch, J. Pührer, H. Tompits, Stepping through an answer-set program, in: J.P. Delgrande, W. Faber (Eds.), Logic Programming and Nonmonotonic Reasoning – 11th International Conference, Proceedings, LPNMR 2011, Vancouver, Canada, May 16–19, 2011, in: LNCS, vol. 6645, Springer, 2011, pp. 134–147.
[76] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.
[77] P. Peppas, Belief revision, in: Handbook of Knowledge Representation, in: Foundations of Artificial Intelligence, vol. 3, Elsevier, 2008, pp. 317–359.
[78] L.M. Pereira, Rational Logic Programming, 1986.
[79] L.M. Pereira, M. Calejo, A framework for prolog debugging, in: ICLP/SLP, 1988, pp. 481–495.
[80] L.M. Pereira, C.V. Damásio, J.J. Alferes, Debugging by diagnosing assumptions, in: P. Fritzson (Ed.), AADEBUG, in: LNCS, vol. 749, Springer, 1993, pp. 58–74.
[81] L.M. Pereira, C.V. Damásio, J.J. Alferes, Diagnosis and debugging as contradiction removal, in: International Conference on Logic Programming and Nonmonotonic Reasoning, 1993, pp. 316–330.
[82] G. Priest, Reasoning about truth, Artif. Intell. 39 (2) (1989) 231–244.
[83] T. Przymusinski, Stable semantics for disjunctive programs, New Gener. Comput. 9 (3) (1991) 401–424.
[84] R. Reiter, A theory of diagnosis from first principles, Artif. Intell. 32 (1987) 57–95.
[85] F. Roelofsen, L. Serafini, Minimal and absent information in contexts, in: International Joint Conference on Artificial Intelligence (IJCAI), 2005, pp. 558–563.
[86] S. Schenk, On the semantics of trust and caching in the semantic web, in: P.A. Sheth, et al. (Eds.), International Semantic Web Conference, in: LNCS, vol. 5318, Springer, 2008, pp. 533–549.
[87] L. Serafini, M. Homola, Contextualized knowledge repositories for the semantic web, J. Web Semant. 12 (2012) 64–87.
[88] E.Y. Shapiro, Algorithmic Program Debugging, MIT Press, 1983.
[89] A.P. Sheth, J.A. Larson, Federated database systems for managing distributed, heterogeneous, and autonomous databases, ACM Comput. Surv. 22 (3) (1990) 183–236.
[90] T. Syrjänen, Debugging inconsistent answer set programs, in: International Workshop on Nonmonotonic Reasoning (NMR), 2006, pp. 77–83.
[91] A. Weinzierl, Comparing inconsistency resolutions in multi-context systems, in: D. Lassiter, M. Slavkovik (Eds.), New Directions in Logic, Language and Computation, in: LNCS, vol. 7415, Springer, Berlin, Heidelberg, 2012, pp. 158–174.
[92] Y. Zhang, Y. Ding, CTL model update for system modifications, J. Artif. Intell. Res. 31 (2008) 113–155.