Chapter 9
Using Contextually Closed Queries for Local Closed-World Reasoning in Rough Knowledge Databases

Patrick Doherty,[1] Jarosław Kachniarz,[2] Andrzej Szałas[3]

[1] Department of Computer and Information Science, Linköping University, 58183 Linköping, Sweden
patdo@ida.liu.se
[2] Soft Computer Consultants, 34350 US19N, Palm Harbor, FL 34684, USA
jk@softcomputer.com
[3] The College of Economics and Computer Science, Wyzwolenia 30, 10-106 Olsztyn, Poland
andsz@ida.liu.se

Summary. Representing internal models of aspects of an autonomous agent's surrounding environment or of its own epistemic state and developing query mechanisms for these models based on efficient forms of inference are fundamental components in any deliberative/reactive system architecture used by an agent in achieving task goals. The problem is complicated by the fact that the models in question necessarily have to be incomplete due to the complexity of the environments in which such agents are intended to operate. Consequently, the querying mechanisms must be framed in the context of an *open-world assumption*. We propose an architecture for such a system that involves generalizing classical deductive databases to rough knowledge databases (RKDB), where relations in the database are defined as rough sets. We also propose the use of *contextually closed queries* (CCQs) where a context for a query and a local minimization policy are provided in terms of integrity constraints and techniques from circumscription. The concept of a contextually closed query is a generalization of querying in the context of a local closed-world assumption (LCW) previously proposed in the literature. CCQs have the effect of dynamically reducing the boundary regions of relations relative to a particular set of integrity constraints associated with the query before actually querying the RKDB. The general problem of querying the RKDB using CCQs is co-NPTIME complete, but we isolate a number of important practical cases where polynomial time and space complexity is achieved.

1 Introduction

Consider an autonomous system, such as a ground robot or an unmanned aerial vehicle (UAV), operating in a highly complex and dynamic environment. For systems of this sort to function intelligently and robustly, it is useful to have both deliberative and reactive capabilities. Such systems combine the use of reactive and deliberative capabilities in achieving task goals. Reactive capabilities are necessary so that the system can react to contingencies that arise unexpectedly and demand immediate

response with little room for deliberation as to what the best response should be. Deliberative capabilities are useful in the sense that internal representations of aspects of the system's operational environment can be used to predict the course of events in the near or intermediate future. These predictions can then be used to determine more selective actions or better responses in the present, which will potentially save the system time, effort, and resources in the course of achieving task goals.

Due to the complexity of the operational environments in which such robotics systems generally operate and the inaccuracy of sensor data about the environment acquired through different combinations of sensors, these systems cannot be assumed to have complete information or models about their surrounding environment nor the effects of their actions on these environments. On the other hand, the deliberative component is dependent on synthesizing, managing, updating, and using of incomplete qualitative models of the operational environment represented internally in the system architecture. These internal models are used for reasoning about the system's environment and the effects of its actions on the environment while the system attempts to achieve task goals. In spite of the lack of complete information, such systems quite often have, or can acquire, additional information that can be used in certain contexts to assume additional knowledge about the incomplete parts of the specification. This information may be of a normative or default nature, may include rules of thumb particular to the operational domain in question, or may include knowledge implicit in the result of executing a sensing action.

One potentially useful approach that can be pursued in developing of on-line reasoning capabilities and representation of qualitative models of aspects of an autonomous system's operational environment is using traditional database technology combined with techniques originating from artificial intelligence research with knowledge-based systems. There are a number of different compositions of technologies that may be pursued, ranging from more homogeneous logic programming based deductive database systems to heterogeneous systems that combine the use of traditional relational database technology with specialized front-end reasoning engines.

The latter approach will be pursued in this chapter, but with a number of modifications of the standard deductive database framework. These modifications are made necessary by the requirement of representing and reasoning about incomplete qualitative models of the operational environments in which autonomous systems are embedded. A number of fundamental generalizations of standard semantic concepts used in the traditional deductive database approach will be made:

- The extensional database (EDB) which represents and stores base relations and properties about the external environment, or the system's internal environment, will be given formal semantics based on the use of rough sets [14, 19]. The extension of a database relation or property will contain explicit positive and negative information in addition to implicitly represented boundary information

that is defined as the difference between upper and lower approximations of the individual relations and properties.

- The intensional database (IDB) will contain two rule sets generating implicit positive and negative information, respectively, via application of the rule sets in the context of the facts in the EDB. The *closed-world assumption* will *not* be applied to the resulting information generated from the EDB/IDB pair.
- An *open-world assumption* will be applied to the extensional and intensional database pair, which can be locally closed dynamically by using of *contextually closed queries*. A CCQ consists of the query itself, a context represented as a set of integrity constraints,[1] and a local closure policy specified in terms of the minimization/maximization of selected relations. The contextually closed query layer (CCQ layer) represents the closure mechanism and is used to answer individual CCQs.

In effect, the CCQ layer permits the representation of additional normative, default, or closure information associated with the operational environment at hand and the particular view of the environment currently used by the querying agent. Together with the rough set semantics for relations, a rough set knowledge base in this context represents an incompletely specified world model with dynamic policies that permit the local closure of parts of the world model when querying it for information.

The combination of the EDB, IDB, and CCQ layer will be called the rough knowledge database. The computational basis for the inference engine used to query the RKDB will be based on the use of circumscription, quantifier elimination, and the ability to automatically generate syntactic characterizations for the upper and lower bounds of rough relations in the RKDB.

1.1 Open-and Closed-World Reasoning

What is meant intuitively by open- and closed-world reasoning? In traditional databases, reasoning is often based on the assumption that information stored in a specific database contains a complete specification of the application environment at hand. If a tuple is not in a base relational table, it is assumed that it does not have that specific property. In deductive databases, if the tuple is not in a base relational table or any intensional relational tables generated implicitly by the application of intensional rules, it is again assumed that it does not have these properties. Under this assumption, an efficient means of representing negative information about the world depends on applying the *closed-world assumption* (CWA) [1, 20]. In this case, atomic information about the world, absent in a world model (represented as a database), is assumed to be false.

[1] We accept a paradigm, according to which integrity constraints are statements about database contents expressed as classical first-order formulas (see, e.g. [1]), that are to be satisfied by the instances of the database. However, since we also deal with incomplete information, we assume that the required satisfiability is restricted to tuples containing only complete information.

On the other hand, for many applications such as the autonomous systems applications already mentioned, the assumption of complete information is not feasible nor realistic, and the CWA cannot be used. In such cases, an *open-world assumption* (OWA), where information not known by an agent is assumed to be unknown, is often accepted, but this complicates both the representational and implementational aspects associated with inference mechanisms and the use of negative information. The CWA and OWA represent two ontological extremes. Quite often, a reasoning agent does have or acquires additional information that permits the application of the CWA *locally* in a particular context. In addition, if it does have knowledge of what it does not know, this information is valuable because it can be used in plan generation to acquire additional information by using of sensors.

In such a context, various forms of LCW assumptions have been defined (see, e.g., [8, 10]), and planning systems have been proposed (see, e.g., [11, 12]). The starting point for the approach proposed by in [8] several authors of this chapter is based on the approach to query answering using LCW assumptions described in [10], where the authors present a sound, but incomplete, tractable algorithm for LCW reasoning intended for use in the XII Planner [13]. The approach described in [10] was substantially strengthened in [8] by

- Providing formal semantics for the case where LCW assumptions and queries are expressed by arbitrary first-order formulas. The semantics is based on the use of formula circumscription and depends on minimizing formulas expressing LCW constraints.
- Isolating a more expressive language for LCW assumptions which subsumes that used in [10], permits limited use of negation and disjunction and still retains tractability.
- Providing a sound and complete, tractable deduction method for the more expressive language.

The semantics of LCW constraints, as defined in [8], depends on minimizing LCW constraints where it is specified that all relations in a constraint vary. The minimization process results in changing the varied database relations as a side effect of the process. Queries are then posed to the changed database. Initial practice in using the strengthened version of LCW assumptions showed that a finer granularity in the minimization policy for LCW assumptions was desirable as was a more intuitive methodology for expressing LCW policies to understand the results and provide intuitive semantics for the database changes. These desiderata have led to the proposal for the modifications and generalizations of deductive database technology described above.

1.2 A New Approach to Rough Set Based LCW Techniques

In the current chapter, we propose semantics and methodology for LCW reasoning that provides a more intuitive and general framework for integrating LCW reasoning

in knowledge databases used by intelligent agents. The new approach differs from and subsumes that of [8] in the following manner:

- It generalizes to deductive databases, whereas the previous approach described in [8] is basically restricted to relational databases.
- Integrity constraints, absent in the previous approach, take on an important role in characterizing LCW assumptions in a principled manner. In most knowledge databases, the relationships between pieces of information are expressed by integrity constraints (e.g., defined by classical first-order formulas). When applying LCW policies locally to particular relations, one minimizes those relations. However, in such cases, the integrity constraints have to be preserved. This can result in implicit changes to some additional relations. However, the integrity constraints are still preserved, thus the knowledge structure represented continues to satisfy the desired properties. Such information was missing in the previous approach, thus it was much more difficult to understand the changes in the resulting database and to develop pragmatic implementation techniques for modifying and querying the knowledge database.
- Integrity constraints and local closure policies are decoupled from the knowledge database itself and associated dynamically with individual agent queries. The agents themselves possess local views and preferences about the world model that may or may not be shared by other agents or even the same agent using a different query.
- The formula-circumscription technique used in the previous approach is replaced by integrity constraints and standard circumscription. This modification permits selected fixing, varying, and minimizing of specific relations in integrity constraints, whereas the previous approach forced varying on all predicates in an LCW constraint. This provides the user with more flexibility in defining LCW constraints and brings the new approach closer to the methodology used in circumscription-based knowledge representation. It should be emphasized that the implementation is not always dependent on circumscription.
- At the semantic level we use rough sets to represent database information as a natural tool.[2] Rough sets contain information about tuples known to be in a relation (the lower approximation of the relation), tuples known not to be in the relation (the complement of the upper approximation of the relation) and tuples for which it is unknown whether they belong to the relation (the difference between the upper and lower approximation of the relation).

1.3 The Structure of This Chapter

In Sect. 2, we provide some notation and a number of definitions. In Sect. 3, we describe the basic architecture for rough knowledge databases consisting of the extensional, intensional, and contextual closure query layer. In Sect. 4, a detailed example from the domain of unmanned aerial vehicles is provided to demonstrate the need

[2] As discussed, e.g., in [14], rough relations appear in databases in many important contexts.

for the reasoning mechanisms we propose. In Sect. 5, we provide specifications of the languages used for the three rough knowledge database layers, and in Sect. 6, we provide the formal semantics for each of the three layers. In Sect. 7, we provide a high-level specification for an algorithm for computing queries to rough knowledge databases and consider complexity and expressiveness issues. In Sect. 8, we isolate a number of important special cases based on restrictions in using language at the three database layers which guarantee efficient mechanisms for computing queries for these cases. In Sect. 9, we conclude with a summary of results and some considerations on future work.

2 Preliminaries

We deal with the first-order language with equality, F_{I}, over a fixed vocabulary without function symbols, where Const is a finite set of *constant symbols*, V_{I} is a finite set of *first-order variables* and Rel is a finite set of *relation symbols*. Any rough relation $R()$ is defined by

- *The positive part of the relation*, containing positive information and denoted by $R^{+}()$ [it is simply the lower approximation of $R()$].
- *The negative part of the relation*, containing negative information and denoted by $R^{-}()$ [it is the complement of the upper approximation of $R()$].
- *The boundary region of the relation*, containing the unknown facts and denoted by $R^{\pm}()$ [it is the difference of the upper and the lower approximation of $R()$].

By F_{II} we denote the second-order language based on an alphabet whose symbols are those of Rel, together with a denumerable set V_{II} of n-ary predicate variables (for each $n \geq 0$). In the rest of the chapter, we shall use second-order circumscription. Our definition follows [15].

Definition 1. Let $\bar{P}$ be a tuple of distinct predicate constants, $\bar{S}$ be a tuple of distinct predicate constants disjoint with $\bar{P}$, and let $T(\bar{P},\bar{S})$ be a finite theory in the language F_{I}. The *second-order circumscription of $\bar{P}$ in $T(\bar{P},\bar{S})$ with variable $\bar{S}$*, written $CIRC(T(\bar{P},\bar{S});\bar{P};\bar{S})$, is the sentence (in the language F_{II})

$$T(\bar{P},\bar{S}) \wedge \forall\bar{\Phi}\forall\bar{\Psi}.\{[T(\bar{\Phi},\bar{\Psi}) \wedge \bar{\Phi} \leq \bar{P}] \rightarrow \bar{P} \leq \bar{\Phi}\}, \tag{1}$$

where $\bar{\Phi}$ and $\bar{\Psi}$ are tuples of predicate variables similar to $\bar{P}$ and $\bar{S}$, respectively,[3] $\bar{\Phi} \leq \bar{P}$ stands for

$$\bigwedge_{i=1}^{n}[\forall\bar{x}.\Phi_i(\bar{x}) \rightarrow P_i(\bar{x})] \text{ and } \bar{P} \leq \bar{\Phi} \text{ stands for } \bigwedge_{i=1}^{n}[\forall\bar{x}.P_i(\bar{x}) \rightarrow \Phi_i(\bar{x})].$$

[3] A tuple of predicate expressions $\bar{X}$ is said to be similar to a tuple of predicate constants $\bar{Y}$ iff $\bar{X} = (X_1,\ldots,X_n)$, $\bar{Y} = (Y_1,\ldots,Y_n)$ and, for all $1 \leq i \leq n$, X_i and Y_i are of the same arity.

In the following, we shall often write $CIRC(T;\bar{P};\bar{S})$ instead of $CIRC(T(\bar{P},\bar{S});\bar{P};\bar{S})$. We also allow minimization of negative literals. Definition 1 can easily be adjusted to this case, since minimizing a literal, say $\neg R$, means maximizing R. Thus it suffices to replace inequalities $\leq$ of (1) by inequalities $\geq$, i.e., to reverse the corresponding implication. We also require the following definition (see also [7,8]).

Definition 2. Let $\bar{x}$ be a tuple of first-order variables, $\bar{Q}$ be a tuple of relation symbols, $\Phi(\bar{Q})$ be a first-order formula positive w.r.t. all symbols in $\bar{Q}$, and $\Psi(\neg\bar{Q})$ be a first-order formula negative w.r.t. all symbols in $\bar{Q}$. Then

- By a *semi-Horn formula*, we shall understand any formula of the following form

$$[\forall\bar{x}.\Phi(\bar{Q}) \rightarrow \bar{Q}(\bar{x})] \wedge \Psi(\neg\bar{Q}).$$

- By a *weak semi-Horn formula*, we shall understand any formula of the following form:

$$[\Phi(\bar{Q}) \rightarrow Q_i(\bar{x})] \wedge \Psi(\neg\bar{Q}).$$

- By a *weak Ackermann formula* we shall understand a weak semi-Horn formula, in which Φ does not contain the relation Q_i.

Observe that in Definition 2, one can replace all occurrences of all relations of $\bar{Q}$ by their negations. This is useful for the application of dual forms of quantifier elimination techniques used in our algorithms.

3 The Architecture of Rough Knowledge Databases

Let us now discuss the architecture of rough knowledge databases as understood in this chapter. The kernel of the database is the so-called extensional database (see Fig. 1). We assume that the extensional database contains positive and negative facts. The facts that are not explicitly listed in the extensional database are assumed to be unknown in this layer of the database. Thus, in the extensional database layer we accept the open-world assumption. The intensional database layer provides rules that define some new relations, but also rules allowing one to extend the positive and negative parts of the extensional relations.[4] The outermost, most advanced layer, which we call the *contextual closure query layer* (CCQ layer), consists of the CCQ inference mechanism which represents the query/answer mechanism used by individual CCQs applied to the two lower layers of the RKDB.

The extensional database consists of rough relations. According to the methodology developed in [5], the rules of the intensional database function as rough set transducers (see also Sect. 6.3), transforming combinations of rough extensional relations

[4] We assume here that extensional and intensional databases are consistent. Let us note that the consistency condition, expressed in the language we consider, is tractable.

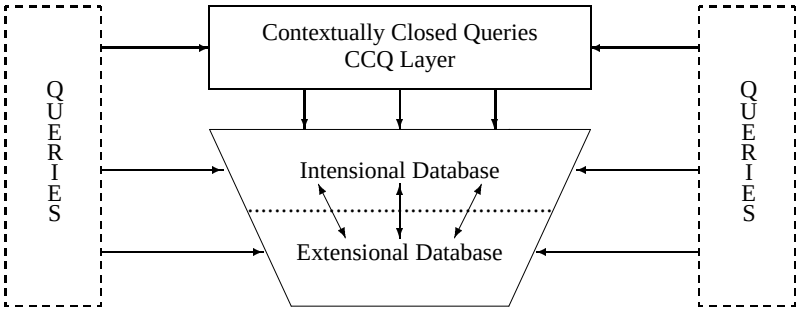

Fig. 1. The architecture of knowledge databases

into new relations that satisfy the constraints of the intensional rules. As in the extensional database, the open-world assumption is accepted in the intensional layer. Local closure context policies (LCC policies) allow us to minimize chosen relations (or their complements), while at the same time preserving the integrity constraints (ICs). Queries are posed via the outermost layer, but in some applications, it might be useful to submit queries to the intensional or even extensional layer. This feature can be provided in an obvious manner. We will focus on the CCQ layer.

4 A UAV Scenario Sensor Usage Example

The WITAS[5] Unmanned Aerial Vehicle Project [3] is a long-term basic research project located at Linköping University (LiU), Sweden, and the authors are participants in the project. The current work with rough knowledge databases and LCC reasoning is intended to be used in an on-line query-answering system which is part of the UAV's system architecture.

The long-term goal of the WITAS UAV Project is the development of the technologies and functionalities necessary for successfully deploying a fully autonomous UAV operating over road and traffic networks. While operating over such an environment, the UAV should be able to navigate autonomously at different altitudes (including autonomous takeoff and landing); plan for mission goals such as locating, identifying, tracking, and monitoring different vehicle types, and construct internal representations of its focus of attention for use in achieving its mission goals. Additionally, it should be able to identify complex patterns of behavior such as vehicle overtaking, traversing of intersections, parking lot activities, etc.

In the current project, we are using a Yamaha RMAX helicopter as the experimental physical platform on which to pursue our research. The helicopter is equipped with

[5] The Wallenberg Laboratory for Information Technology and Autonomous Systems (Pronounced *Vee-Tas*).

a sensor platform that includes a geographical positioning system (GPS), an inertial navigation system (INS), elevation sensors, and a magnetic compass, in addition to a video camera.

The system architecture for the UAV consists of both deliberative and reactive components, and the communication infrastructure for the components is based on the use of the standard object broker CORBA. There are a number of deliberative services such as task planners, trajectory planners, prediction mechanisms, and chronicle recognizers, that are dependent on internal qualitative representations of the environment over which the UAV operates. The knowledge representation components include a soft-real time database called the dynamic object repository, a standard relational database, a geographic information system containing road and geographic data, and a number of front-end query-answering systems that serve as inference engines and may be used by other components in the architecture. The research described in this chapter provides a basis for one of the inference engines. In addition to these components, there is an image processing module used for low and intermediate level vision tasks and a helicopter control module which is used to position the helicopter and camera dynamically and maintain positions during the execution of task goals which may include highly dynamic tasks such as tracking vehicles through a small village with building obstacles.
Let's examine a particular scenario from the UAV operational environment representative of the use of LCC reasoning in the UAV context.

Suppose the UAV receives the following mission goal from its ground control operator:

> Identify and track *all* moving vehicles in region X, and log the estimated velocities and positions of *all* small blue vehicles identified for the duration of their stay in region X, or until the UAV is low on fuel.

Achieving a mission goal such as this in a fully autonomous mode is extremely complex and would involve the concurrent use of many of the deliberative and reactive services in the architecture, in addition to a great deal of sophisticated reasoning about the operational environment. Both hard and soft real-time constraints must also be taken to consideration, particularly for query-answering during a plan execution phase. In this example, we will focus on a particular type of reasoning capability made possible by the combined use of LCC reasoning and rough knowledge databases.

The first step in achieving the mission goal would be to generate a task plan which would include the following steps:

1. Fly to a position that permits viewing region X and possibly an area surrounding the region.
2. Focus the camera on region X, and maintain position, focus, and coverage.

3. Initiate the proper image processing algorithms for identifying moving vehicles in region X.
4. Use the sensor data gathered in the previous step to produce knowledge as to what is seen or not seen by the UAV in region X.
5. Use the acquired knowledge to plan for the next series of actions which involve tracking, feature recognition, and logging.
6. Maintain execution of the necessary services and processes until the mission goal is completed.

We will concentrate on steps 3 and 4 whose successful completion is dependent on a combination of the open-world assumption, LCC reasoning, and rough knowledge database representation of relations and properties.

Observe that the mission goal above contains two universal statements, the first asks to "identify and track *all* moving vehicles in region X," and the second asks to "log the estimated velocities and positions of *all* small blue vehicles identified." The meaning of the second universal is naturally dependent on the meaning of the first universal. To achieve the mission goal, the inferencing mechanism used by the UAV during plan generation and plan execution must be able to circumscribe (in the intuitive sense) the meaning of "*all* moving vehicles in region X" and that of "*all* small blue vehicles identified."

What the UAV *can perceive as moving*, given the constraints under which it is operating, the character of the dynamics of its current operational environment, and the capabilities of its sensor and image processing functionalities in this context, is not necessarily the same thing as what *is actually moving* in region X. An additional problem, of course, is that the inferencing mechanism cannot appeal to the use of the closed-world assumption. If it could, it would register moving objects in region X and *assume* via application of the CWA that no other objects are moving. One cannot appeal to this mechanism because the open-world assumption is being used. Even if one could, this would be erroneous. Certainly, there may be vehicles in region X that are moving but can not be perceived due to limitations associated with the UAV's capabilities, and there may also be vehicles outside region X that are moving.

The key to solving this particular representational problem is to note that sensing actions, such as step 3 in the plan sketch above, implicitly generate local or contextual closure information (LCC policies) and that the UAV agent can query the rough knowledge database using the particular contextual closure that exists for the purpose at hand. For example, the sensing action in step 3 above not only generates information about specific moving individuals the UAV can perceive with its current sensor and image processing capabilities, but it also generates knowledge that this is all the UAV can see in the region of interest (ROI), region X. The nature of this information is that it is specific knowledge of what the UAV agent does not see rather than information derived via an assumption such as the CWA.
Of course, one has to (or more specifically, the UAV agent has to) supply the con-

textual closure information. This will be supplied in terms of one or more integrity constraints and an LCC policy consisting of particular LCC assumptions pertaining to the minimization, maximization, fixing, or varying of specific relations. The specific closure context for this situation could be paraphrased as follows:

> After sensing region X with a camera sensor, assume that all moving vehicles in the ROI (X) have been perceived except for those with a signature whose color feature is roadgray.

In the following example, we provide the particulars for representing the scenario above and reasoning about it using the proposed approach.

Example 1. [A UAV Scenario: Identify, Track, and Log]
Consider the situation where a UAV observes and classifies cars with different signatures based on color.[6] For the example, the domains considered consist of

- $Cars = \{c_1, c_2, c_3, c_4, c_5, c_6\}$.
- $Regions = \{r_1, r_2, r_3, r_4\}$.
- $Signatures = \{\mathsf{blue}, \mathsf{roadgray}, \mathsf{green}, \mathsf{yellow}\}$.

The following relations are also defined:[7]

- $Moving(c)$ the object c is moving.
- $InROI(r)$ the region r is in the region of interest.
- $See(c,r)$ the object c is seen by the UAV in region r.
- $In(c,r)$ the object c is in region r.
- $ContainedIn(r,r')$ region r is contained in region r'.
- $Sig(c,s)$ the object c has signature s.

Suppose the actual situation in the operational environment over which the UAV is flying is as depicted in Fig. 2. For the mission goal, the UAV's initial region of interest (ROI) is region r_3.

At mission start, the following facts are in the UAV's on-line extensional database (EDB):

$$\{ContainedIn(r_1,r_2), ContainedIn(r_2,r_3)\}.$$

During mission preparation, the ground operator relays the following information to the UAV agent which is placed in the UAV's on-line EDB:

$$\{In(c_2,r_2), In(c_3,r_3), Moving(c_2),$$
$$Sig(c_2,\mathsf{roadgray}), Sig(c_3,\mathsf{green}), Sig(c_6,\mathsf{roadgray}), InROI(r_3)\}.$$

[6] In an actual scenario, a vehicle signature would be more complex and contain features such as width, height, and length, or vehicle type.

[7] In addition, a number of type properties, such as $Car()$, $Region()$, etc. would also be defined.

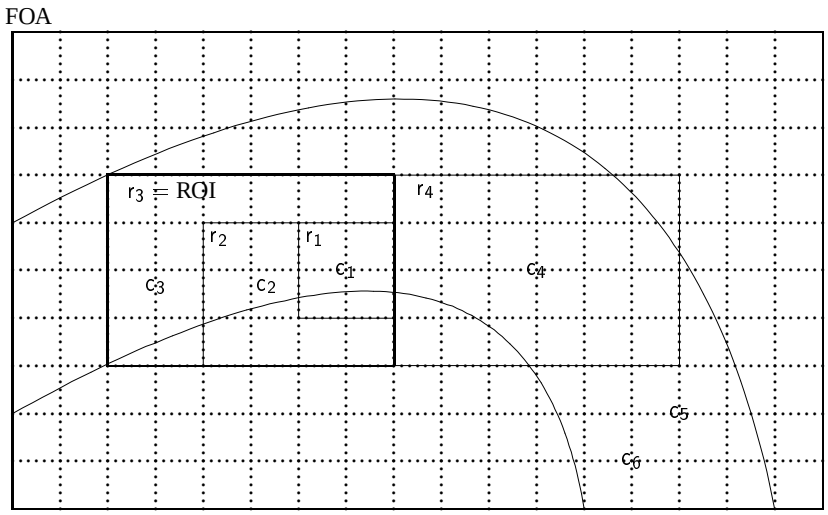

Fig. 2. The situation considered in Example 1.

The following rules are associated with the intensional database:

$$ContainedIn(r,s) \leftarrow \exists t.[ContainedIn(r,t) \wedge ContainedIn(t,s)], \tag{2}$$

$$InROI(s) \leftarrow \exists r.[ContainedIn(s,r) \wedge InROI(r)]. \tag{3}$$

The current EDB, together with the intensional database (IDB), would allow the UAV agent to infer the following additional facts:

$$\{ContainedIn(\mathsf{r}_1,\mathsf{r}_3), InROI(\mathsf{r}_2), InROI(\mathsf{r}_1)\}.$$

Observe that complete information about the *ContainedIn* and *InROI* relations is not yet assumed due to the application of an open-world assumption in the EDB and IDB.

Assume that the UAV generates a plan similar to that described at the beginning of this section and then executes steps 1–3 in this plan. Given its sensor capabilities under current weather conditions, suppose that the UAV agent can assert the following new set of facts in the EDB generated from its sensor actions and image processing facilities (step 3 in the plan):

$$\{In(\mathsf{c}_1,\mathsf{r}_1), In(\mathsf{c}_4,\mathsf{r}_4), Moving(\mathsf{c}_1), Moving(\mathsf{c}_4), Moving(\mathsf{c}_5)$$
$$Sig(\mathsf{c}_1,\mathsf{blue}), Sig(\mathsf{c}_4,\mathsf{yellow})\}.$$

After executing the sensor action in step 3, the UAV's on-line EDB contains the

following facts:

$$\begin{aligned}&\{In(\mathsf{c}_1,\mathsf{r}_1), In(\mathsf{c}_2,\mathsf{r}_2), In(\mathsf{c}_3,\mathsf{r}_3), In(\mathsf{c}_4,\mathsf{r}_4),\\ &Moving(\mathsf{c}_1), Moving(\mathsf{c}_2), Moving(\mathsf{c}_4), Moving(\mathsf{c}_5)\\ &Sig(\mathsf{c}_1,\mathsf{blue}), Sig(\mathsf{c}_2,\mathsf{roadgray}), Sig(\mathsf{c}_3,\mathsf{green}), Sig(\mathsf{c}_4,\mathsf{yellow}),\\ &Sig(\mathsf{c}_6,\mathsf{roadgray}), ContainedIn(\mathsf{r}_1,\mathsf{r}_2), ContainedIn(\mathsf{r}_2,\mathsf{r}_3),\\ &InROI(\mathsf{r}_3)\}.\end{aligned} \tag{4}$$

At this point, observe that, due to the open-world assumption, it is unknown whether c_3 is moving and it is unknown what region c_5 is in or what color it is. Additionally, it is unknown what region c_6 is in or whether it is moving.

Before proceeding with the execution of the rest of the plan, the UAV must take stock of what it knows about the ROI, r_3. In other words, the UAV agent must query the RKDB with a particular policy of contextual closure to determine not only what it sees, but *all* that it sees under the current circumstances. The following closure context discussed above,

> after sensing region X with a camera sensor, assume that all moving vehicles in the ROI X have been perceived except for those with a signature whose color feature is roadgray,

can be represented as the following integrity constraint:

$$\begin{aligned}\forall x,r,z.[&Moving(x) \wedge In(x,r) \wedge InROI(r) \wedge Sig(x,z) \wedge\\ &z \neq \mathsf{roadgray}] \rightarrow See(x,r),\end{aligned} \tag{5}$$

together with the following LCC policy:[8]

$$\mathsf{LCC}[See(x,r), ContainedIn(x,y); Moving()] : (5). \tag{6}$$

This combination states that relations $See(x,r)$ and $ContainedIn(x,y)$ are minimized, relation $Moving()$ is allowed to vary, and all other relations are fixed. The integrity constraint (5) is to be preserved. In essence, the UAV agent is assuming complete information locally about the $ContainedIn()$ and $See()$ relations by minimizing them. In addition, new information about moving may also be derived, but the only information about $Sig()$ and the other fixed relations that can be inferred is what is already in the EDB. That is the effect of *fixing* relations in this context.

Another way to view the integrity constraint (5) of the contextual closure is as the equivalent:

$$\begin{aligned}\forall x,r,z.[&In(x,r) \wedge InROI(r) \wedge Sig(x,z) \wedge z \neq \mathsf{roadgray} \wedge\\ &\neg See(x,r)] \rightarrow \neg Moving(x),\end{aligned} \tag{7}$$

[8] The formal definition of an LCC policy is provided in Sect. 5.3, but we first treat it informally here.

which states that "if an object is in the ROI and it has a visible signature relative to the current capabilities of the UAV agent's sensors, if the UAV agent does not see it, then it is not moving." The integrity constraint is intended to represent strong coupling between moving and seeing due to the character of the sensor capabilities in this context.

After applying the LCC policy in the CCQ layer which includes integrity constraints, the resulting EDB/IDB combination would contain (explicitly and implicitly) the following facts:

$$\begin{aligned}
&\{\mathit{In}(\mathsf{c}_1,\mathsf{r}_1),\mathit{In}(\mathsf{c}_2,\mathsf{r}_2),\mathit{In}(\mathsf{c}_3,\mathsf{r}_3),\mathit{In}(\mathsf{c}_4,\mathsf{r}_4),\\
&\mathit{Moving}(\mathsf{c}_1),\mathit{Moving}(\mathsf{c}_2),\mathit{Moving}(\mathsf{c}_4),\mathit{Moving}(\mathsf{c}_5),\neg\mathit{Moving}(\mathsf{c}_3),\\
&\mathit{Sig}(\mathsf{c}_1,\mathsf{blue}),\mathit{Sig}(\mathsf{c}_2,\mathsf{roadgray}),\mathit{Sig}(\mathsf{c}_3,\mathsf{green}),\mathit{Sig}(\mathsf{c}_4,\mathsf{yellow}),\\
&\mathit{ContainedIn}(\mathsf{r}_1,\mathsf{r}_2),\mathit{ContainedIn}(\mathsf{r}_2,\mathsf{r}_3),\mathit{ContainedIn}(\mathsf{r}_1,\mathsf{r}_3),\\
&\neg\mathit{ContainedIn}(r,r') \text{ for all pairs } r,r' \text{ other than listed above},\\
&\mathit{InROI}(\mathsf{r}_1),\mathit{InROI}(\mathsf{r}_2),\mathit{InROI}(\mathsf{r}_3)\}.
\end{aligned}$$

It is useful to note the following about the UAV agent's knowledge about the ROI, resulting from its sensing action in step 3 and subsequent reasoning about it. It still has incomplete information about the relations $\mathit{In}()$, $\mathit{Sig}()$, and $\mathit{InROI}()$. For example, it is unknown what signature object c_5 has or where it is. The UAV agent now knows that object c_3 is not moving and it does have complete information about the $\mathit{ContainedIn}()$ relation.

What about the relation $\mathit{See}()$, which has been minimized? One can now infer the following facts related to the relation $\mathit{See}()$ and the ROI, r_3:

- $\mathit{See}(\mathsf{c}_1,\mathsf{r}_3)$.
- $\neg\mathit{See}(\mathsf{c}_2,\mathsf{r}_3),\neg\mathit{See}(\mathsf{c}_3,\mathsf{r}_3),\neg\mathit{See}(\mathsf{c}_4,\mathsf{r}_3),\neg\mathit{See}(\mathsf{c}_6,\mathsf{r}_3)$.

c_3 is not seen because it is not moving. c_4 is not seen because it is not in the ROI, r_3. c_2 is not seen even though it is moving because of its signature. Most interestingly, it is unknown whether c_5 is seen because the UAV agent could not discern which region c_5 was in nor what its color signature was. In fact, since the UAV agent could identify c_5 as moving, the failure to discern a region for c_5 could be deduced as the reason for this, due to the tight coupling between moving and seeing. This could provide a reason for focusing on c_5 and trying to discern its region. c_6 is not seen because of its signature. What is interesting is that minimization of $\mathit{See}()$ does not change the status of c_6 w.r.t. $\mathit{Moving}()$, i.e., $\mathit{Moving}(\mathsf{c}_6)$ remains unknown.

The fact that $\mathit{See}(\mathsf{c}_5,\mathsf{r}_3)$ and $\mathit{Moving}(\mathsf{c}_6)$ remain unknown informs us of the subtlety of minimization in the context of rough sets. The minimization of a relation in the rough set context does not necessarily create a definition of the relation minimized. What it does do is move tuples in the boundaries of one or more relations into the positive or negative parts of the relation while meeting the conditions of the

integrity constraints, whereas other tuples still remain in the boundaries. This is very important because it satisfies the ontological intuition associated with open-world reasoning.

It also worth emphasizing that if the integrity constraint (5) was defined as an intensional rule, like the following,

$$See(x) \leftarrow Moving(x) \wedge In(x,r) \wedge InROI(r) \wedge Sig(x,z) \wedge z \neq \mathsf{roadgray},$$

then the minimization of $See()$ would not result in changes in the relation $Moving()$ appearing in the body of the rule.

These subtle forms of inferencing are precisely what may be required for realistic inferencing mechanisms in fully autonomous systems, such as that described here, that may often be in situations where there is no communication for longer periods of time between the autonomous agent and ground operators. It remains to be seen whether such complex forms of inferencing can be implemented efficiently. This aspect will be considered in the remainder of the chapter.

5 The Languages of RKDBs

5.1 The Language of Extensional Databases

An extensional database consists of positive and negative facts. Thus, we assume that the language of the extensional database is a set of literals, i.e., formulas of the form $R(\bar{c})$ or $\neg R(\bar{c})$, where $R \in \mathsf{Rel}$ is a relation symbol and $\bar{c}$ is a tuple of constant symbols. It is assumed that the extensional database is consistent, i.e., it does not contain both $R(\bar{c})$ and $\neg R(\bar{c})$, for some relation $R()$ and tuple $\bar{c}$.

5.2 The Language of Intensional Databases

The intensional database is intended to infer new facts, both positive and negative, by applying intensional rules to the EDB. The process is similar to the approach using $\mathsf{Datalog}^{\neg\neg}$ (see, e.g., [1]). The rules have the form,

$$\pm P(\bar{x}) \leftarrow \pm P_1(\bar{x_1}), \ldots, \pm P_k(\bar{x_k}), \tag{8}$$

where $\pm$ is either the empty string or the negation symbol $\neg$ and any variable that appears in the head of a rule [i.e., any variable of $\bar{x}$ in a rule of the form (8)] appears also in the rule's body (i.e., among variables of $\bar{x_1}, \ldots, \bar{x_k}$ in the rule).
The rules can be divided into two layers, the first for inferring positive and the second for inferring negative facts. The first layer of rules (called the *positive* IDB *rule layer*), used for inferring positive facts, has the form,

$$P(\bar{x}) \leftarrow \pm P_1(\bar{x_1}), \ldots, \pm P_k(\bar{x_k}); \tag{9}$$

the second layer of rules (called the *negative* IDB *rule layer*), used for inferring negative facts, has the following form:

$$\neg P(\bar{x}) \leftarrow \pm P_1(\bar{x_1}), \ldots, \pm P_k(\bar{x_k}). \tag{10}$$

5.3 The Language of Integrity Constraints and LCC Policies

Integrity constraints are expressed as formulas of classical first-order logic. Intuitively, they can be considered implicit definitions of intensional relations that will be minimized or maximized by the LCC assumptions in a specific LCC policy. In the following sections, to obtain tractable instances of the general algorithm, we will impose some syntactic restrictions on the syntactic form of ICs together with the LCC assumptions in a specific LCC policy (see Sect. 8).

LCC *policies* are expressions of the form,

$$\mathsf{LCC}[L_1,\ldots,L_p;K_1,\ldots,K_r]\text{:IC}, \tag{11}$$

where $L_1,\ldots,L_p$ are (positive or negative) literals, $K_1,\ldots,K_r$ are relation symbols not appearing in L_i's, and IC is a set of integrity constraints. Literals $L_1,\ldots,L_p$ are minimized and relations $K_1,\ldots,K_r$ are allowed to vary. By an LCC assumption, we mean a minimization or maximization of a single literal from $L_1,\ldots,L_p$ in (11).
In the following sections, we often omit the part ":IC" of (11) if the corresponding integrity constraints are known from the context.

6 The Semantics of RKDBs

6.1 Notational Conventions

Let us denote the facts in the extensional database by EDB and the facts in the intensional database by IDB. Let $R_1,\ldots,R_n$ be all relations in the RKDB. For a specific relation R in the RKDB, we denote the positive atoms of R in the EDB by $EDB^+(R)$ and the negative atoms of R in the EDB by $EDB^-(R)$. Assume that

- By EDB^+, we denote the positive part of the EDB which is $\bigcup_{i=1}^{n} EDB^+(R_i)$.
- By EDB^-, we denote the negative part of the EDB which is $\bigcup_{i=1}^{n} EDB^-(R_i)$.

The EDB is then equivalent to $EDB^+ \cup EDB^-$.

For a specific relation R in the RKDB, we denote the positive atoms of R in the IDB generated by the positive intensional rules of form (9) by $IDB^+(R)$ (where it is assumed that $IDB^+(R) \equiv IDB^+(R) \setminus EDB^+(R)$) and the negative atoms of R in the IDB generated by the negative intensional rules of form (10) by $IDB^-(R)$ (where it is assumed that $IDB^-(R) \equiv IDB^-(R) \setminus EDB^-(R)$). Assume also that

- By IDB^+, we denote the positive part of the IDB which is $\bigcup_{i=1}^{n} IDB^+(R_i)$.
- By IDB^-, we denote the negative part of the IDB which is $\bigcup_{i=1}^{n} IDB^-(R_i)$.

The IDB is then equivalent to $IDB^+ \cup IDB^-$.

Let D be a finite set (a domain of the database). The semantics of constant symbols and variables is given by an assignment of domain values to constants and variables, called a *valuation*:

$$v : \mathsf{Const} \cup V_{\mathsf{I}} \longrightarrow D.$$

The valuation v is then extended to the vectors of constants and variables in the usual way. We also assume that the unique names assumption (UNA) holds, i.e., for all different constants c, c' in the RKDB, we assume that c and c' denote different objects. In other words, the formula $c_i \neq c_j$ is satisfied for each i, j, where $i \neq j$.

In the semantics defined in the following sections, all relations are interpreted as rough sets of tuples, where no form of domain closure is required. The symbol $\Vdash$ will denote the RKDB entailment relation and the symbol $\models$ will denote the classical two-valued entailment relation. By indexing relations with EDB, IDB, and LCC, we indicate that they are considered in the particular context as relations of the extensional, intensional, and CCQ layer of the RKDB, respectively.

6.2 The Semantics of Extensional Databases

The semantics of the extensional database is given by rough sets of tuples. Let $R()$ be a relational symbol appearing in the extensional database. Then $R()$ is interpreted as the rough set whose positive part contains all tuples $v(\bar{c})$ for which literal $R(\bar{c})$ is in the database, and the negative part contains all tuples $v(\bar{c})$ for which literal $\neg R(\bar{c})$ is in the database. All other tuples are in the boundary region of $R()$.

$$EDB \Vdash R(\bar{a}) \text{ iff } R(\bar{a}) \in EDB^{+}(R),$$
$$EDB \Vdash \neg R(\bar{a}) \text{ iff } \neg R(\bar{a}) \in EDB^{-}(R),$$

where $R()$ is a relation of the EDB and $\bar{a}$ is a tuple of constants.

Rough relations for the EDB are then defined as follows:

$$R^{+}_{EDB} = \{v(\bar{a}) : EDB \Vdash R(\bar{a})\},$$
$$R^{-}_{EDB} = \{v(\bar{a}) : EDB \Vdash \neg R(\bar{a})\},$$
$$R^{\pm}_{EDB} = \{v(\bar{a}) : EDB \nVdash R(\bar{a}) \text{ and } EDB \nVdash \neg R(\bar{a})\}.$$

6.3 The Semantics of Intensional Databases

The semantics of the intensional database is given by rough sets of tuples after application of the intensional rules to the extensional database. Intensional rules can be viewed as rough set transducers (see [5]). Basically, a rough set transducer takes rough sets as input and generates new or modified rough sets as output meeting the constraints of the transducer, a set of formulas.

To provide the semantics of the IDB, we will use the definition of the so-called Feferman–Gilmore translation (see, e.g., [2]) as a basis.

Definition 3. By a *Feferman–Gilmore translation of formula* α, denoted by $FG(\alpha)$, we mean the formula obtained from α by replacing all negative literals of the form $\neg R(\bar{y})$ by $R^-(\bar{y})$ and all positive literals of the form $R(\bar{y})$ by $R^+(\bar{y})$.

Let $\bar{S} = (S_1, \ldots, S_p)$ contain all relation symbols of the form R^+ and R^-, where R is a relation symbol in an IDB rule. For any relation S_i, all rules with S_i^+ (respectively, S_i^-) in their heads should be gathered into a single formula of the form,

$$\forall \bar{y_i}.[S_i^{\pm}(\bar{y_i}) \leftarrow \alpha_i(\bar{y_i})],$$

where

$$\alpha_i(\bar{y_i}) \equiv \bigvee_j \exists \bar{z_j}.\beta_{ij}(\bar{z_j})$$

where $\beta_{ij}(\bar{z_j})$ denotes the bodies of the appropriate rules and $\pm$ stands for $+$ or $-$, respectively.

Denote by $\mu\bar{S}.[\alpha(\bar{S})]$ the least, and by $\nu\bar{S}.[\alpha(\bar{S})]$, the greatest simultaneous fixed-point operator of $\alpha(\bar{S})$ (for the definition of simultaneous fixed-points see, e.g. [9]). Define $\bar{S_{IDB}} \equiv \mu\bar{S}.[FG(\alpha_1), \ldots, FG(\alpha_p)]$. In some cases the IDB might appear inconsistent when there is a relation $R()$ such that $R^+ \cap R^- \neq \emptyset$. In what follows we require that the IDB is consistent, i.e., for all IDB relations $R()$, $R^+ \cap R^- = \emptyset$. This consistency criterion can be verified in time polynomial in the size of the database.

The semantics of IDB rules is then defined as follows:

$$IDB \Vdash R(\bar{a}) \text{ iff } \bar{a} \in EDB^+(R) \cup IDB^+(R),$$
$$IDB \Vdash \neg R(\bar{a}) \text{ iff } \bar{a} \in EDB^-(R) \cup IDB^-(R),$$

where $R()$ is a relation in the EDB or in the head of an intensional rule, $\bar{a}$ is a tuple of constants, and $IDB^+(R)$ and $IDB^-(R)$ are computed from the simultaneous fixed-point definition $\bar{S_{IDB}}$ defined above.

Rough relations for the IDB are then defined as follows:

$$R^+_{IDB} = \{v(\bar{a}) : IDB \Vdash R(\bar{a})\},$$
$$R^-_{IDB} = \{v(\bar{a}) : IDB \Vdash \neg R(\bar{a})\},$$
$$R^{\pm}_{IDB} = \{v(\bar{a}) : IDB \nVdash R(\bar{a}) \text{ and } IDB \nVdash \neg R(\bar{a})\}.$$

Observe that

$$EDB \Vdash R(\bar{a}) \text{ implies } IDB \Vdash R(\bar{a}),$$
$$EDB \Vdash \neg R(\bar{a}) \text{ implies } IDB \Vdash \neg R(\bar{a}).$$

Remark 1. If one wants to distinguish between facts entailed solely by application of intensional rules, this can be done in a straightforward manner, but as a rule, one is interested in querying both the EDB and IDB together, thus the choice of RKDB entailment from the IDB.

6.4 The Semantics of the CCQ Layer and LCC Policies

The inference mechanism associated with the CCQ layer is intended to provide a form of *contextual closure* relative to part of the EDB and IDB when querying the RKDB. A *contextually closed query* consists of

- The *query* itself, which can be any fixed-point or first-order query.
- The *context* represented as a set of one or more integrity constraints.
- A *local closure policy* representing the closure context and consisting of a minimization policy representing the local closure.

An LCC *policy* consists of a context and a local closure policy. LCC policies may also be viewed as rough set transducers with rough relations in the EDB and IDB as input, a transducer consisting of one or more integrity constraints and a minimization policy, and modified rough relations in the RKDB as output.

Let the EDB and IDB be defined as before, let IC denote a finite set of integrity constraints, and let RKDB:LCC$[\bar{L};\bar{K}]$:IC denote querying the three layers of the RKDB with a specific LCC policy LCC$[\bar{L};\bar{K}]$:IC. Then,[9]

$$\begin{aligned}
&\text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \Vdash R(\bar{a}) \text{ iff}\\
&\qquad CIRC(IC \cup IDB \cup EDB;\bar{L};\bar{K}) \models R(\bar{a}),\\
&\text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \Vdash \neg R(\bar{a}) \text{ iff}\\
&\qquad CIRC(IC \cup IDB \cup EDB;\bar{L};\bar{K}) \models \neg R(\bar{a}),
\end{aligned}$$

where the notation is, as in Sect. 6.1, under the assumption that the circumscriptive theory is consistent.

Thus, the CCQ layer has the purpose of dynamically redefining some relations to satisfy ICs in a particular query. A relation R which is minimized, maximized or allowed to vary is defined as the following rough relation:

$$\begin{aligned}
R^{+}_{\text{LCC}} &= \{v(\bar{a}) : \text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \Vdash R(\bar{a})\},\\
R^{-}_{\text{LCC}} &= \{v(\bar{a}) : \text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \Vdash \neg R(\bar{a})\},\\
R^{\pm}_{\text{LCC}} &= \{v(\bar{a}) : \text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \nVdash R(\bar{a}) \text{ and}\\
&\qquad\qquad \text{RKDB:LCC}[\bar{L};\bar{K}]\text{:IC} \nVdash \neg R(\bar{a})\}.
\end{aligned}$$

Intuitively, this means that the positive part of $R()$ contains tuples present in all extensions of $R()$ satisfying the ICs, the boundary part contains tuples present in some extensions of $R()$ satisfying the ICs, but not in all of them, and the negative part of $R()$ contains tuples not present in any extension of $R()$ satisfying the ICs.

The relations that are not minimized, maximized, or allowed to vary are not changed; thus their semantics is that given by the EDB and IDB layers of the RKDB.

[9] Observe that we abuse notation somewhat by using sets of literals, $\bar{L},\bar{K}$ for minimized and varied predicate constants in the circumscription formula $CIRC()$. Formally, we should use predicate constants contained in $\bar{L},\bar{K}$, respectively.

Remark 2. The inference mechanism associated with the CCQ layer is almost always used with both layers of the EDB and IDB. If one wants to apply LCC inference to just the EDB, this can also be done in a straightforward manner.

7 The Computation Method

7.1 The Pragmatics of Computing Contextual Queries

A contextual query in its simplest form involves the (implicit) generation of the extension of a relation R in the context of a set of integrity constraints and a minimization policy and asking whether one or more tuples is a member of that relation. Essentially, we are required to implicitly compute R^{+}_{LCC}, R^{-}_{LCC}, and $R^{\pm}_{\mathsf{LCC}}$ and determine whether the tuple or tuples are in any of the resulting rough set partitions of R. In Sect. 7.2, we will describe an algorithm to do this. Based on this specification, we will be able to show that in some cases, where the LCC policy associated with the query is restricted appropriately, querying the relation R can be done very efficiently. One of the more important results is that one can automatically generate syntactic characterizations of each of the partitions of a rough set relation without actually generating their explicit extensions. The syntactic characterizations can then be used to efficiently query the RKDB.

Since integrity constraints are not associated with the EDB/IDB pair, but with an agent posing a query, the integrity constraints associated with an agent are not necessarily satisfied together with the EDB/IDB. Checking satisfiability is tractable in this context, due to the first-order or fixed-point nature of the integrity constraints and the finiteness of the database. Under additional syntactic restrictions, the satisfiability of the circumscriptive theory can also be guaranteed. In the case of inconsistency, this would lead to the specification and computation of specific update policies which is a topic for future research.

7.2 The Algorithm

The algorithm presented below applies to the general case, i.e., to the problem which is co-NPTIME complete (see Sect. 7.4). However, in Sect. 8, we show specializations of the algorithm to some cases, where PTIME complexity is guaranteed. The inputs to the algorithm are

- An extensional database EDB
- An intensional database IDB
- A set of integrity constraints IC
- An LCC policy $\mathsf{LCC}[\bar{L};\bar{K}]$:IC
- A relation symbol R.[10]

[10] The relation symbol R can be viewed as part of the query which consists of a number of relations that are required to compute the full query.

As output, the algorithm returns the definition of the relation $R()$ obtained by applying the LCC policy and preserving the ICs, according to the semantics defined in Sect. 6.

1. Construct $C \equiv CIRC(IC \cup IDB \cup EDB; \bar{L}; \bar{K})$ representing the given LCC policy applied to the IDB together with the EDB.
2. Eliminate second-order quantifiers from the formula obtained in step 1. In general, the elimination may fail, and the result is the initial second-order formula C, however, if certain restrictions concerning the form of IC are assumed, the elimination of second-order quantifiers is guaranteed (see Sect. 8).
3. Calculate the intersection of all extensions of R satisfying formula C. If there is not any relation $R()$ satisfying C, terminate and return the answer "unsatisfiable," meaning that either the EDB and IDB pair is inconsistent, or the ICs cannot be satisfied.
4. Calculate the union of all extensions of R satisfying formula C.
5. For any tuple $\bar{a}$:
   - if $v(\bar{a})$ is in the intersection calculated in step 3, add $v(\bar{a})$ to $R^{+}()$.
   - if $v(\bar{a})$ is not in the union calculated in step 4, add $v(\bar{a})$ to $R^{-}()$.
   - if none of the above two cases applies, then $v(\bar{a})$ is in $R^{\pm}()$.

In practice, one uses particular second-order quantifier elimination algorithms (see e.g., [6,16,17,21]) that may fail. Since second-order formulas are useless as results, it is reasonable to return the answer "unknown" when the elimination algorithm used in step 2 fails. This implies that the algorithm is only sound relative to the semantics provided in Sect. 6.4.

Observe also that, in practice, it is often better to calculate the definitions of new relations rather than calculating their extensions as in the above algorithm. To achieve this goal one can apply, e.g., techniques proposed in [4,7].

7.3 Expressiveness of the Approach

Noted that the approach we consider here subsumes that of [8], namely, consider a policy $\mathsf{LCW}[\beta(\bar{R})]$ of [8], meaning that formula $\beta(\bar{R})$ is to be minimized, whereas all relations in β, i.e. $\bar{R}$, are allowed to vary. This LCW policy is expressible by the following LCC policy:

$$\mathsf{LCC}[S; \bar{R}] : \{\forall \bar{x}.[\beta(\bar{R}) \rightarrow S(\bar{x})]\},$$

where $\bar{x}$ denotes all free variables of $\beta(\bar{R})$ and S is a fresh relation symbol, not appearing among symbols in $\bar{R}$.

An interesting question arises whether the current approach allows one to express all tractable LCC policies, where by a *tractable* LCC *policy*, we mean any LCC policy such that all minimized, maximized, and varied relations are PTIME-computable. The following characterization shows that the method presented is strong enough

to express all tractable LCC policies. In other words, any tractable LCC policy can always be reformulated in the form used in Lemma 1 below. In Sect. 8, we provide additional syntactic characterizations of LCC policies that guarantee tractability.

Lemma 1. Given the LCC semantics for LCC policies provided in Sect. 6.4 and assuming that the database domain is ordered, all tractable LCC policies can be expressed as policies of the form,

$$\mathsf{LCC}[\bar{L};\bar{K}] : \{\beta_i(\bar{x}) \rightarrow L_i(\bar{x}) : L_i \in \bar{L}\},$$

where each $\beta_i(\bar{x})$ is a first-order formula positive w.r.t. L_i.

Proof. Any relation computable in PTIME can be expressed by means of the least fixed-point of a formula of the form,

$$\beta_i(\bar{x}) \rightarrow L_i(\bar{x}), \tag{12}$$

provided that the database domain is ordered (see, e.g., [9]). Since all minimized, maximized, and varied relations are assumed to be tractable, they can be expressed by the least fixed-points of formulas of the form (12), thus also by policy $\mathsf{LCC}[\bar{L};\bar{K}] : \{\beta_i(\bar{x}) \rightarrow L_i(\bar{x}) : L_i \in \bar{L}\}$. □

7.4 Complexity of the Approach

In general, the problem of querying the database in the presence of unrestricted ICs is co-NPTIME complete. On the other hand, some classes of LCC policies for which the computation mechanism is in PTIME can be isolated (see, e.g., [8] and also Sect. 8.3).

8 Important Particular Cases

In this section, we consider a number of restrictions on ICs that allow us to compute explicit definitions of the new relations as first-order and fixed-point formulas. In such cases, computing contextually closed queries is in PTIME.

Let $M(\bar{R})$ stand for $IC \cup EDB \cup IDB$, and assume that the ICs have the following form:

$$\forall \bar{x}.[\alpha(\bar{x}) \rightarrow \beta(\bar{x})], \tag{13}$$

where α and β are first-order formulas.

Definition 4. By a *marking* of relation symbols for the policy $\mathsf{LCC}[\bar{L};\bar{K}]$:IC, we understand a mapping assigning, to any relation symbol, both in the local closure policy $\mathsf{LCC}[\bar{L};\bar{K}]$ and IC in the LCC policy, the least subset of $\{\min, \max\}$ that is closed under the following rules:

1. For any relation symbol S appearing in $\bar{L}$ positively, min is in the set of marks of S.
2. For any relation symbol S appearing in $\bar{L}$ negatively, max is in the set of marks of S.
3. If $\alpha(R) \rightarrow \beta(S)$ is in IC, $R,S \in \bar{L} \cup \bar{K}$ and S occurs in β positively and is marked by min, or S occurs in β negatively and is marked by max; then,
   - If R occurs positively in α, min is in the set of marks of R.
   - If R occurs negatively in α, max is in the set of marks of R.
4. If $\alpha(R) \rightarrow \beta(S)$ is in IC, $R,S \in \bar{L} \cup \bar{K}$ and α contains a positive occurrence of R and R is marked by max, or α contains a negative occurrence of R and R is marked by min then:
   - if S occurs positively in β, then 'max' is in the set of marks of S.
   - if S occurs negatively in β, then 'min' is in the set of marks of S.

An $\mathsf{LCC}[\bar{L};\bar{K}]$:IC policy is called *uniform* if no relation symbol is marked by both max and min.

Example 2. Let us consider the following integrity constraint:

$$[Car(x) \wedge Red(x)] \rightarrow RedCar(x). \tag{14}$$

The marking for the policy,

$$\mathsf{LCC}[\{RedCar(x), Car(x)\}; \{Red(x)\}] : (14),$$

assigns the mark min to all the relation symbols. Thus the policy is uniform. On the other hand, the marking for the policy,

$$\mathsf{LCC}[\{RedCar(x), \neg Car(x)\}; \{Red(x)\}] : (14),$$

assigns the mark min to *Red* and the marks $\{\min, \max\}$ to *Car* and *RedCar*. Thus the latter policy is not uniform.

8.1 The Case of Universal LCC policies

Definition 5. By a *universal* LCC *policy*, we understand any uniform policy,

$$\mathsf{LCC}[\bar{L};\bar{K}]\text{: IC},$$

in which IC is a set of constraints of the following form,

$$\forall \bar{y}. \{[\pm P_1(\bar{x_1}) \wedge \ldots \wedge \pm P_k(\bar{x_k})] \rightarrow \pm P(\bar{x})\}, \tag{15}$$

where $P_1, \ldots, P_k, P$ are relation symbols, $\bar{y}$ is the vector of all variables occurring in $\bar{x_1}, \ldots, \bar{x_k}, \bar{x}$, and $\bar{x} \subseteq \bar{x_1} \cup \ldots \cup \bar{x_k}$.

For universal integrity constraints, we will have a computation method much more efficient than that described in Sect. 7.2. In the rest of this section, we will consider only universal LCC policies $\mathsf{LCC}[\overline{L};\overline{K}]$:IC, for given sets of literals $\overline{L}$,$\overline{K}$ and a set IC of integrity constraints.

In the computation method for universal policies, we first construct minimal rough relations satisfying the EDB, IDB, and the integrity constraints, where minimality is defined w.r.t. the so-called *information ordering* considered by Fitting and van Benthem (see, e.g., [2]) in the context of three-valued logics. The definition of information ordering follows.

Definition 6. Let R and S be rough relations. We define *information ordering*, denoted by $R \sqsubseteq S$, as follows:

$$R \sqsubseteq S \stackrel{\text{def}}{\equiv} R^+ \subseteq S^+ \text{ and } R^- \subseteq S^-.$$

To find minimal w.r.t. $\sqsubseteq$ rough relations satisfying IC, EDB, and IDB, we will use the following tautologies of first-order logic:

$$\begin{aligned} &\forall \overline{x}.\{\alpha(\overline{R}) \rightarrow [\beta(\overline{R}) \vee M(\overline{y})]\} \equiv \forall \overline{x}.\{[\alpha(\overline{R}) \wedge \neg M(\overline{y})] \rightarrow \beta(\overline{R})\}, \\ &\forall \overline{x}.\{[\alpha(\overline{R}) \wedge M(\overline{y})] \rightarrow \beta(\overline{R})\} \equiv \forall \overline{x}.\{\alpha(\overline{R}) \rightarrow [\beta(\overline{R}) \vee \neg M(\overline{y})]\}, \end{aligned} \tag{16}$$

where it is assumed that all double negations $\neg\neg$ are removed.

Definition 7. Let I be an integrity constraint of the form:

$$\forall \overline{x}.\{[\pm R_1(\overline{y_1}) \wedge \ldots \wedge \pm R_m(\overline{y_m})] \rightarrow \pm S(\overline{z})\}. \tag{17}$$

Let $\mathcal{P} = \mathsf{LCC}[\overline{L};\overline{K}]$:$I$ be an LCC policy. By the *expansion of I w.r.t. $\mathcal{P}$*, denoted by $Exp^{\mathcal{P}}(I)$, we understand the least set of constraints of the form,

$$\forall \overline{x}.\left\{\left[\bigwedge_k L_k(\overline{x_k})\right] \rightarrow \pm S(\overline{x_S})\right\},$$

obtained from (17) by applying the tautologies (16), such that any (possibly negated) literal of (17) containing a relation symbol occurring in $\overline{L};\overline{K}$, is a consequent of exactly one constraint.

Example 3. Consider the integrity constraint

$$I \stackrel{\text{def}}{\equiv} \forall x,y.\{[\neg P(x) \wedge S(x,y)] \rightarrow P(y)\},$$

and the policy $\mathcal{P} = \mathsf{LCC}[P;S]$:$I$. The expansion of I w.r.t. $\mathcal{P}$ is defined as the following set of constraints:

$$\begin{aligned} Exp^{\mathcal{P}}(I) = \; & (\forall x,y.\{[\neg P(x) \wedge S(x,y)] \rightarrow P(y)\} \\ & \forall x,y.\{[\neg P(y) \wedge S(x,y)] \rightarrow P(x)\} \\ & \forall x,y.\{[\neg P(x) \wedge \neg P(y)] \rightarrow \neg S(x,y)\}). \end{aligned}$$

In the case of policy $\mathcal{P}' = \mathsf{LCC}[S;\emptyset]{:}I$, the expansion of I is defined as

$$Exp^{\mathcal{P}'}(I) = (\forall x,y.\,\{[\neg P(x) \wedge \neg P(y)] \rightarrow \neg S(x,y)\})\,.$$

Let us fix an LCC policy $\mathcal{P} = \mathsf{LCC}[\bar{L};\bar{K}]{:}\mathrm{IC}$. To compute the definition of minimal w.r.t. $\sqsubseteq$ rough relations, satisfying the constraints IC, EDB, and IDB, we consider the following cases:

- If $S_i \notin \bar{L} \cup \bar{K}$, then the positive part of the resulting relation, $S_i^+()$, contains exactly the tuples present in $EDB^+(S_i) \cup IDB^+(S_i)$, and the negative part of the resulting relation, $S_i^-()$, contains exactly the tuples present in $EDB^-(S_i) \cup IDB^-(S_i)$.
- If $\bar{S} = \bar{L} \cup \bar{K}$, then we consider the set of integrity constraints:

  $$\{FG(\alpha) : \alpha \in Exp^{\mathcal{P}}(I) \text{ and } I \in IC\,\},$$

  where FG is the Feferman–Gilmore translation defined in Definition 3.

  We assume that the following integrity constraints, reflecting the contents of EDB and IDB, are implicitly given:

  $$\begin{aligned}
  &\forall \bar{y}.\left\{EDB^+[S(\bar{y})] \rightarrow S^+(\bar{y})\right\},\\
  &\forall \bar{y}.\left\{EDB^-[S(\bar{y})] \rightarrow S^-(\bar{y})\right\},\\
  &\forall \bar{y}.\left\{IDB^+[S(\bar{y})] \rightarrow S^+(\bar{y})\right\},\\
  &\forall \bar{y}.\left\{IDB^-[S(\bar{y})] \rightarrow S^-(\bar{y})\right\},
  \end{aligned}$$

  where the empty parts $EDB^+(S(\bar{y}))$, $EDB^-(S(\bar{y}))$, $IDB^+(S(\bar{y}))$, and $IDB^-(S(\bar{y}))$ are interpreted as false.

  Now, for each $S_i \in \bar{S}$, gather all the ICs with S_i^+ as the consequent into the following single formula:

  $$\forall \bar{y}.\left\{\left[\bigvee_{1\le k\le k_i} \exists \bar{z}_{ik}.\phi_{ik}(\bar{R}_k)\right] \rightarrow S_i^+(\bar{y})\right\}, \tag{18}$$

  and all the ICs with S_i^- as the consequent into the following single formula:

  $$\forall \bar{y}.\left\{\left[\bigvee_{1\le j\le j_i} \exists \bar{z}_{ij}.\psi_{ij}(\bar{R}_j)\right] \rightarrow S_i^-(\bar{y})\right\}. \tag{19}$$

  The following definitions of the positive and the negative part of the required minimal rough relations wrt policy $\mathcal{P}$, indicated by the index $\mathcal{P}$, can now be derived:

  $$\bar{S}^+_{\mathcal{P}}(\bar{y}) \equiv \mu\bar{S}(\bar{y}).\left[\bigvee_{1\le k\le k_1} \exists \bar{z}_{1k}.\phi_{1k}(\bar{R}_k), \ldots, \bigvee_{1\le k\le k_n} \exists \bar{z}_{nk}.\phi_{nk}(\bar{R}_k)\right] \tag{20}$$

$$\bar{S}^{-}_{\mathcal{P}}(\bar{y}) \equiv \mu\bar{S}(\bar{y}).\left[\bigvee_{1\leq j\leq j_1} \exists\bar{z}_{1j}.\psi_{1j}(\bar{R}_j),\ldots,\bigvee_{1\leq j\leq j_m} \exists\bar{z}_{mj}.\psi_{mj}(\bar{R}_j)\right]. \quad (21)$$

Observe that the syntactic restrictions placed on the ICs guarantee that the formulas under the fixed-point operators are positive, thus, the monotone w.r.t. S and consequently, the fixed-points exist. Observe also, that for nonrecursive universal LCC policies, the fixed-point operators can be removed, and the definitions obtained are classical first-order formulas.[11]

Having computed the suitable parts of the relations in all integrity constraints, one can easily perform a consistency check, indicating whether the ICs can be satisfied by the current contents of the $EDB \cup IDB$. For each relation $R()$, one needs to assure that $R^+ \cap R^- = \emptyset$.

Definition 8. Let $\mathcal{P}$ =LCC$[\bar{L};\bar{K}]$:IC be an LCC policy. The *rough negation for the policy* $\mathcal{P}$, denoted by $\sim_{\mathcal{P}}$, is defined as follows:

- $\sim_{\mathcal{P}}$ satisfies the usual DeMorgan laws for quantifiers, conjunction, and disjunction, and

$$\sim_{\mathcal{P}} \mu\bar{R}.\alpha(\bar{R}) \stackrel{\text{def}}{\equiv} \nu\bar{R}.\sim_{\mathcal{P}} \alpha(\bar{R}),$$
$$\sim_{\mathcal{P}} \nu\bar{R}.\alpha(\bar{R}) \stackrel{\text{def}}{\equiv} \mu\bar{R}.\sim_{\mathcal{P}} \alpha(\bar{R}).$$

- If $S \in \bar{L} \cup \bar{K}$, then,

$$\sim_{\mathcal{P}} S^+() \stackrel{\text{def}}{\equiv} \neg S^+(), \quad \sim_{\mathcal{P}} S^-() \stackrel{\text{def}}{\equiv} \neg S^-(),$$
$$\sim_{\mathcal{P}} \neg S^+() \stackrel{\text{def}}{\equiv} S^+(), \quad \sim_{\mathcal{P}} \neg S^-() \stackrel{\text{def}}{\equiv} S^-().$$

- If $S \notin \bar{L} \cup \bar{K}$, then,

$$\sim_{\mathcal{P}} S^+() \stackrel{\text{def}}{\equiv} S^-(), \qquad \sim_{\mathcal{P}} S^-() \stackrel{\text{def}}{\equiv} S^+(),$$
$$\sim_{\mathcal{P}} \neg S^+() \stackrel{\text{def}}{\equiv} \neg S^-(), \quad \sim_{\mathcal{P}} \neg S^-() \stackrel{\text{def}}{\equiv} \neg S^+().$$

If the ICs are consistent with $EDB \cup IDB$, then the definitions of minimal and maximal rough relations satisfying the ICs and reflecting the semantics introduced in Sect. 6.4 can be calculated as follows:[12]

$$S^+_{min}(\bar{y}) \equiv \bar{S}^+_{\mathcal{P}}, \quad (22)$$
$$S^-_{min}(\bar{y}) \equiv \sim_{\mathcal{P}} S^+_{min}(\bar{y}), \quad (23)$$
$$S^-_{max}(\bar{y}) \equiv \bar{S}^-_{\mathcal{P}}, \quad (24)$$
$$S^+_{max}(\bar{y}) \equiv \sim_{\mathcal{P}} S^-_{max}(\bar{y}). \quad (25)$$

[11] In both cases, however, computing the defined parts of relations can be done in time polynomial in the size of $EDB \cup IDB$.

[12] Observe that the LCC policies provide us with direct information about which relations are to be maximized, which are to be minimized, and which remain unchanged.

For nonrecursive universal policies, the fixed-point operators can be removed, as before.

Observe that definitions of varied predicates can now be computed by noticing that these are the minimal w.r.t. $\sqsubseteq$ rough relations satisfying the ICs in the new context of minimized and maximized relations. It then suffices to apply definitions (20) and (21) with minimized and maximized relations replaced by their definitions obtained as (22–25) , as appropriate.

Example 4. Consider the UAV sensing example introduced in Sect. 4. The definition of minimal $See()$ is given by

$$\begin{aligned} See^+_{min}(x,r) \equiv\ & See^+(x,r) \vee \{\exists z.\, [Moving^+(x) \wedge In^+(x,r) \wedge \\ & InROI^+(r) \wedge Sig^+(x,z) \wedge z \neq \mathsf{roadgray}]\}, \\ See^-_{min}(x,r) \equiv\ & \neg See^+(x,r) \wedge \{\forall z.\, [\neg Moving^+(x) \vee In^-(x,r) \vee \\ & InROI^-(r) \vee Sig^-(x,z) \vee z = \mathsf{roadgray}]\}. \end{aligned}$$

The varied relation $Moving()$ is defined by

$$\begin{aligned} Moving^+_{var}(x) \equiv\ & Moving^+(x), \\ Moving^-_{var}(x) \equiv\ & Moving^-(x) \vee \{\exists r \exists z.\, [In^+(x,r) \wedge InROI^+(r) \wedge \\ & Sig^+(x,z) \wedge z \neq \mathsf{roadgray} \wedge See^-_{min}(x,r)]\}. \end{aligned}$$

Example 5. Consider the problem of determining whether a given car on a road is seen. We assume that large cars are usually seen. Our database contains the following relations:

- $Car()$ containing cars
- $Large()$ containing large objects
- $See()$ containing visible objects
- $Ab()$ standing for abnormal objects, i.e., large but invisible objects.

Define the following integrity constraint IC:

$$\forall x.\, \{[Car(x) \wedge Large(x) \wedge \neg See(x)] \rightarrow Ab(x)\}.$$

We want to minimize abnormality, i.e., to minimize relation *Ab*, while keeping the relations *Car* and *Large* unchanged. The local closure policy is then

$$\mathsf{LCC}[\{Ab(x)\}; \{See(x)\}].$$

According to Lemma 2.4, we obtain the following characterizations of $Ab()$ and $See()$:

$$\begin{aligned} & Ab^+_{min}(x) \equiv Ab^+(x) \vee [Car^+(x) \wedge Large^+(x) \wedge See^-(x)], \\ & Ab^-_{min}(x) \equiv Ab^-(x) \vee [Car^-(x) \vee Large^-(x) \vee \neg See^-(x)], \\ & See^+_{var}(x) \equiv See^+(x) \vee [Car^+(x) \wedge Large^+(x) \wedge Ab^-_{min}(x)], \\ & See^-_{var}(x) \equiv See^-(x). \end{aligned}$$

8.2 The Case of Semi-Horn LCC Policies

Assume that

1. Any integrity constraint IC_j is expressed as a formula of the form

$$\forall \bar{x}.[\beta_j(\bar{x}) \rightarrow S_j(\bar{x})],$$

   where for each $j = 1,\ldots,n$, S_j is a relation symbol and $\beta_j(\bar{x})$ is a first-order formula.
2. Any LCC assumption L_j, in the given LCC policy, has the form S_j or $\neg S_j$, where S_j is a relation symbol.

We now have the the following proposition.

Proposition 1. Under the above assumptions 1 and 2,

$$Circ(M(R);\bar{L};\emptyset) \equiv M(\bar{R}) \wedge \bigwedge_{j=1}^{n} \forall \bar{y}.\left[\neg L_j(\bar{y}) \vee \neg A_j(\bar{y})\right], \tag{26}$$

where $A_j(\bar{y})$ is the following second-order formula:

$$A_j(\bar{y}) \equiv \exists \bar{S}' \exists \bar{K}'.\left\{ M(\bar{R}') \wedge \neg L'_j(\bar{y}) \wedge \bigwedge_{1\leq i\leq n, i\neq j} \forall \bar{x}.\left[\neg L'_i(\bar{x}) \vee L_i(\bar{x})\right] \right\}, \tag{27}$$

in which $\bar{R}' \equiv \bar{S}' \cup \bar{K}' \cup (\bar{R} - \bar{S} - \bar{K})$, L'_j stands for $L_j[\bar{S} \leftarrow \bar{S}', \bar{K} \leftarrow \bar{K}']$ and $M(\bar{R}')$ represents $IC' \cup IDB' \cup EDB'$.

The following lemma holds under assumptions 1 and 2.

Lemma 2.

1. If the LCC assumption L_j has the form $\neg S_j$ and β_j is a semi-Horn formula, then the elimination of second-order quantifiers is guaranteed in time polynomial in the size of formula β_j.
2. If β_j is expressed as a weak semi-Horn formula, then the elimination of second-order quantifiers is guaranteed in PTIME. The resulting formula $A_j(\bar{y})$ is a classical or fixed-point first-order formula. It is also guaranteed that elimination of second-order quantifiers from the formula $\exists \bar{K} Circ(\bar{K})$ succeeds in time polynomial in the size of formula β_j. However, necessary computations may require calculation of simultaneous fixed-points.
3. If β_j is expressed as a weak Ackermann formula, then the elimination of second-order quantifiers is guaranteed in PTIME, and $A_j(\bar{y})$ is a classical first-order formula. Elimination of second-order quantifiers from the formula $\exists \bar{K} Circ(\bar{K})$ is also guaranteed in time polynomial in the size of formula β_j.

4. For of a consistent EDB, a uniform LCC policy, and nonrecursive integrity constraints, $A_j(\bar{y})$ is expressed by the following formula:

$$A_j(\bar{y}) \equiv \neg L'_j(\bar{y})(\bar{K'} \leftarrow \neg \bar{K^-}, \neg \bar{K'} \leftarrow \neg \bar{K^+}),$$

   in which all L'_j are replaced accordingly by $\beta_j(\bar{K'})$ or $\neg\beta_j(\bar{K'})$, $\neg K_i$ are replaced by $\neg K_i^+$, and K_i by $\neg K_i^-$.
5. If L_j is of the form S_j and $\neg\beta_j$ is a semi-Horn formula, then the elimination of second-order quantifiers is guaranteed. However, the resulting formula $A_j(\bar{y})$ may have an exponential size w.r.t. the size of β_j.

For semi-Horn formulas, the following lemma, simplifying the inference method, holds.

Lemma 3. If $\Psi(\bar{R})$ is a semi-Horn formula, then

$$\Psi(\bar{R}) \Vdash R(\bar{a}) \text{ iff } \bar{R}_{min} \models R(\bar{a}),$$
$$\Psi(\bar{R}) \Vdash \neg R(\bar{a}) \text{ iff } \bar{R}_{max} \models \neg R(\bar{a}),$$

where $\bar{R}_{min}$ (resp. $\bar{R}_{max}$) is a minimal (resp., maximal) relation satisfying $\Psi(\bar{R})$. In this case, both $\bar{R}_{min}$ and $\bar{R}_{max}$ can be computed in PTIME.

Thus, the general computation algorithm presented in Sect. 7.2 can be specialized in the following way: The inputs to the algorithm are the same as in Sect. 7.2, however, the LCC policy LCC$[\bar{L};\bar{K}]$:IC is assumed to satisfy syntactic restrictions as formulated in Lemma 2 by any of the points 1–4, accordingly. Note that assumptions 1 and 2 from the beginning of this section, as required by Lemma 2, should also be satisfied.

The specialized algorithm is formulated as follows:

1. Construct $C \equiv CIRC(IC \cup IDB \cup EDB; \bar{L}; \bar{K})$ representing the given LCC policy applied to the IDB together with the EDB.
2. Eliminate second-order quantifiers from the formula obtained in step 1. The elimination is guaranteed to succeed in PTIME. As a result, a first-order or fixed-point formula is obtained.
3. Calculate the minimal extension of R satisfying formula C. It can be done by computing the minimal R from the second-order formula, $\exists R.C(R)$. As a result, a definition of relation R, which is a definition of $R^+_{var}()$, and a coherence condition are obtained. If the coherence condition is not satisfiable, terminate and return the answer "unsatisfiable," meaning that either the EDB and IDB pair is inconsistent or the ICs cannot be satisfied. All of the above computations can be performed in PTIME.
4. Calculate the maximal extension of R satisfying formula C. It can be done by computing the maximal R from the second-order formula, $\exists R.C(R)$. As a result, a definition of the complement of R^-_{var} is obtained. All of the above computations can be performed in PTIME.

The algorithm presented above executes in PTIME and the necessary second-order quantifier elimination can be performed automatically (using the techniques described in [6,16,17,21]).

8.3 The Case of Nonuniform LCC policies

Observe that in the case of nonuniform LCC policies, we still might obtain tractable subcases of the general case of policies. One such large class is defined in [8]. The other classes are also discussed in Sect. 8.2, in particular, in Lemma 2.

One of the promising methods depends on first computing the corresponding circumscription (applying the Doherty, Łukaszewicz, Szałas (DLS) algorithm of [6]) and then on computing the definitions of the required minimal and maximal relations by using the methodology developed in [4] and [7].

9 Conclusions

We proposed the use of rough knowledge databases to represent incomplete models of aspects of an agent's operational environment or world model. Relations represented as tables were generalized to rough sets with partitions for positive, negative, and boundary information. Then, we introduced the idea of a contextually closed query consisting of a query, a context represented as a set of integrity constraints, and a local closure policy. The LCC policy, consisting of integrity constraints and local closure policy, was applied to the intensional and extensional database layers before actually querying the RKDB. The combination of a contextually closed query and a RKDB provided the basis for an inference mechanism that could be used under the open-world assumption.

The inference mechanism and modeling approach has many applications, particularly in the area of planning with an open-world assumption, where sensor actions and knowledge preconditions are essential components in a plan and an efficient query/answer system is used in both the plan generation and execution process. We demonstrated the idea with a scenario from an unmanned aerial vehicle project. In the general case, the problem of querying the RKDB using CCQs is co-NPTIME complete, but we could isolate a number of important practical cases where polynomial time and space complexity is achieved.

In the future, there are a number of interesting topics to pursue. The use of contextually closed queries in an open-world planner has already been mentioned. Another particularly interesting issue has to do with updating the RKDB. Since each querying agent "carries" its context with it, the issue of satisfiability of the integrity constraints relative to the EDB/IDB pair and satisfiability of the pursuant minimization policy are essential aspects of the approach. We have shown that satisfiability can be checked efficiently. Then, the research question is, "what should be done when

a query is not satisfiable relative to the EDB/IDB pair?" This is a question posed and considered in the area of belief revision and update or in what is more traditionally called view update in the relational database area. One final pragmatic issue involves implementation of the techniques proposed in this chapter in an on-line query/answering system for the WITAS UAV project discussed in the chapter. Parts of a prototype system have already been implemented and empirical experiments in a real-time context are planned for the future.

Acknowledgments

This research has been supported in part by the Wallenberg Foundation, Sweden. Andrzej Szałas has additionally been supported by KBN grant 8T11C 025 19. We thank Jonas Kvarnström for thorough and valuable proofreading of this chapter.

References

1. S. Abiteboul, R. Hull, V. Vianu. *Foundations of Databases*. Addison-Wesley, Boston, 1995.
2. D. Busch. Sequent formalizations of three-valued logic. In P. Doherty, editor, *Partiality, Modality and Nonmonotonicity*, 45–75, CSLI Publications, Stanford, CA, 1996.
3. P. Doherty, G. Granlund, K. Kuchcinski, K. Nordberg, E. Sandewall, E. Skarman, J. Wiklund. The WITAS unmanned aerial vehicle project. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, 747–755, IOS, Amsterdam, 2000.
4. P. Doherty, J. Kachniarz, A. Szałas. Meta-queries on deductive databases. *Fundamenta Informaticae*, 40(1): 17–30, 1999.
5. P. Doherty, W. Łukaszewicz, A. Skowron, A. Szałas. Approximation transducers and trees: A technique for combining rough and crisp knowledge (this book).
6. P. Doherty, W. Łukaszewicz, A. Szałas. Computing circumscription revisited. *Journal of Automted Reasoning*, 18(3): 297–336, 1997. See also Report number LiTH-IDA-R-94-42 of Linköping University, 1994 and *Proceedings of the 14th International Joint Conference on AI (IJCAI'95)*, Morgan Kaufmann, San Francisco, 1995.
7. P. Doherty, W. Łukaszewicz, A. Szałas. Declarative PTIME queries for relational databases using quantifier elimination. *Journal of Logic and Computation*, 9(5): 739–761, 1999. See also: Report number LiTH-IDA-R-96-34 of Linköping University, 1996.
8. P. Doherty, W. Łukaszewicz, A. Szałas. Efficient reasoning using the local closed-world assumption. In A. Cerri, D. Dochev, editors, *Proceedings of the 9th International Conference (AIMSA 2000)*, LNAI 1904, 49–58, Springer, Heidelberg, 2000.
9. H-D. Ebbinghaus, J. Flum. *Finite Model Theory*. Springer, Heidelberg, 1995.
10. O. Etzioni, K. Golden, D.S. Weld. Sound and efficient closed–world reasoning for planning. *Artificial Intelligence*, 89: 113–148, 1997.
11. O. Etzioni, S. Hanks, D.S. Weld, D. Draper, N. Lesh, M. Williamson. An approach to planning with incomplete information. In B. Nebel, C. Rich, W.R. Swartout, editors, *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning, (KR'92)*, 115–125, Morgan Kaufmann, San Francisco, 1992.
12. A. Finzi, F. Pirri, R. Reiter. Open world planning in the situation calculus. In *Proceedings of the 17th National Conference on Artificial Intelligence (NCAI 2000)*, MIT Press, Cambridge, MA, 2000.

13. K. Golden, O. Etzioni, D. Weld. XII: Planning for universal quantification and incomplete information. Technical Report of the University of Washington, Department of Computer Science and Engineering, Seattle, 1994.
14. J. Komorowski, Z. Pawlak, L. Polkowski, A. Skowron. Rough sets: A tutorial. In *[18]*, 3–98, 1999.
15. W. Łukaszewicz. *Non-Monotonic Reasoning - Formalization of Commonsense Reasoning*. Ellis Horwood, Chichester, 1990.
16. A. Nonnengart, H.J. Ohlbach, A. Szałas. Elimination of predicate quantifiers. In H.J. Ohlbach, U. Reyle, editors, *Logic, Language and Reasoning. Essays in Honor of Dov Gabbay, Part I*, 159–181, Kluwer, Dordrecht, 1999.
17. A. Nonnengart, A. Szałas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. In E. Orłowska, editor, *Logic at Work: Essays Dedicated to the Memory of Helena Rasiowa*, 307–328, Physica, Heidelberg, 1998. See also Report number MPI-I-95-2-007 of Max-Planck-Institut fuer Informatik, Saarbruecken, 1995.
18. S.K. Pal, A. Skowron, editors. *Rough Fuzzy Hybridization: A New Trend in Decision-Making*. Springer, Singapore, 1999.
19. Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning About Data*. Kluwer, Dordrecht, 1991.
20. R. Reiter. On closed world data bases. In H. Gallaire, J. Minker, editors, *Logic and Data Bases*, 55–76, Plenum, Dordrecht, 1978.
21. A. Szałas. On the correspondence between modal and classical logic: An automated approach. *Journal of Logic and Computation*, 3:605–620, 1993.