Reasoning on Class Relations: An Overview

Stephan Mäs

Abstract Class relations are used whenever the semantics of entire classes are described, independently of single entities. This chapter focuses on class relations that define cardinality restrictions for a certain instance relation (e.g., a topological relation) between all entities of the involved classes. Typical examples are spatial semantic integrity constraints or ontologies of geospatial entity classes. Reasoning on such class relations allows for the detection of inconsistencies and redundancies in sets of class relations. Therefore the logical properties of the applied instance relations and those of the cardinality restrictions have to be considered, in particular symmetry and compositions, but also other inferences. The chapter provides a summary of research and a discussion of open issues for future work on class relation reasoning.

Keywords Class relations · Spatial reasoning · Composition · Inheritance · Semantic integrity constraints

1 Introduction

The inclusion of spatial and temporal concepts, rules and relations should be a main consideration when designing geographic ontologies (Agarwal 2005). The corresponding formalization of spatial and temporal relations and hierarchies to enable a consistent representation of real world phenomena is still one of the major research

S. Mäs (✉)
Department of Geosciences, Professorship of Geoinformation Systems, Technische Universität Dresden, Helmholtzstraße 10, 01069 Dresden, Germany
e-mail: Stephan.Maes@tu-dresden.de

M. Raubal et al. (eds.), *Cognitive and Linguistic Aspects of Geographic Space*, Lecture Notes in Geoinformation and Cartography, DOI: 10.1007/978-3-642-34359-9_13,

themes in GI Science. While research on spatio-temporal relations and their logical properties has been relatively intense in the last two decades (Cohn and Hazarika 2001), the definition of corresponding class relations and their logical properties is a relatively new field. Nevertheless, it is of particular interest for geographical information, since the concepts of such data often refer to spatial relationships which can be represented by class relations (e.g. "*alluvial forests are surrounded by a floodplain*"). As already argued by Donnelly and Bittner (2005) the facilitation of interoperability requires a clear distinction if a relation holds among classes of entities (i.e., universals, feature classes or types) or among concrete entities (i.e., instances, objects or individuals), in particular when the logical properties of the relations are analyzed. Typical class relations are inheritance/generalization relations (Brachman 1983; Baumeister and Seipel 2006). This chapter focuses on class relations that define cardinality restrictions for a certain relation between all entities of the involved classes. Thereby the class relations neither specify the exact number of instances of the classes nor the particular relations between single entities. The constrained instance relation can be of any kind; for geographical information typical examples are the topological (Egenhofer and Herring 1991) or metric relations (Frank 1992) between spatial entities.

The following example shall illustrate the definition of such class relations and the feasibility to infer implicit knowledge with them. The entity relationship diagram in Fig. 1a contains the three classes A*irport*, *Forest* and *Airport Tower*. Among the classes three class relations are defined:

- *airports* and *forests* are either disjoint or meet.
- every *airport* contains at least one *airport tower* and every *airport tower* is contained by an *airport*.
- *forests* and *airport towers* are disjoint.

Such relations are commonly defined in an ontology or as semantic integrity constraints as part of a data model (Tarquini and Clementini 2008; Mäs et al. 2005; Mäs and Reinhardt 2009). In Fig. 1b the last of the three class relations is left out. Considering that the applied instance relations are the topological relations between areal entities defined in Egenhofer and Herring (1991), it is relatively obvious that the two diagrams have the same restriction on the relation between

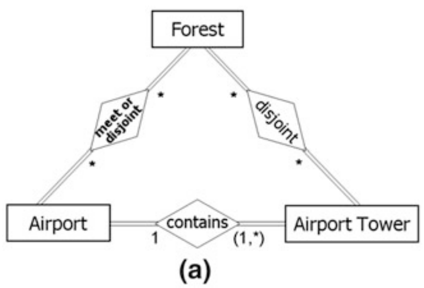

Fig. 1 Entity-Relationship Diagram of the three entity classes and their class relations

the classes *Forests* and *Airport Towers.* Since every *forest* meets or is disjoint from every *airport* the entities of these classes can have intersecting boundaries, but their interiors are disjoint. Since every *airport tower* is contained by an *airport* an *airport tower* has an intersecting interior with this *airport*, but no intersecting boundary. Therewith is no intersection between any *airport tower* and any *forest* possible, even if the third class relation is not explicitly defined. For quality assurance this means that the third class relation (i.e., semantic integrity constraint) does not need to be proven if a data set is conform to the first two. This shows that such a detection and removal of redundant constraints enables the reduction of the calculation costs of a quality check. In practice, this can be of great value, since the constraint sets used, for example, by utility companies or public agencies can easily contain hundreds of constraints.

Similar examples can be found in many disciplines, also in "non-spatial" ontologies such as in biomedicine (Donnelly et al. 2006): from "every *vertebra* has some *cartilage* as a proper part" and "every *vertebra* is a proper part of some *vertebral column* and every *vertebral column* has some *vertebra* as a proper part", it follows that "every *vertebral column* has some *cartilage* as a proper part". For a triple of class relations, like those in the two examples, it is easy to imagine that the third relation could also be in contradiction to the restrictions implied by the other two. Also two relations that apply to the same classes can specify contradicting restrictions. Therefore, the internal consistency of such sets of class relations must be assured.

Most ontologies deal with the description of classes and therefore with the formal description of relations among those classes. Nevertheless, class relations other than inheritance/generalization relations are hardly used for the inference of implicit knowledge or analyzed for conflicting assertions. The examples illustrate the need of methodologies to compare, manage and consistently integrate class relations. A consistency check of class relations must include implicitly defined constraints, because the constraints do not necessarily directly contradict. Conflicts might, for example, result from other constraints defined within a triple of classes. The objective of this chapter is to summarize available approaches for the detection of explicit and implicit redundancies and contradictions and inequalities in sets of class relations.

A class relation can be defined in terms of an instance relation (e.g., the topological relation "*disjoint"*) or a disjunction of instance relations (e.g., "*meet or disjoint"*). It is obvious that the reasoning properties of such class relation are influenced by those of the applied instance relation. Therefore it is reasonable to use and extend the reasoning properties and methods of instance relations for reasoning based on class relations. The formal theory of relations of individuals is the necessary foundation for the formal theory of class relations (Donnelly et al. 2006). Nevertheless, a relation among classes is not necessarily subject to the same logical properties as a relation between instances. The cardinality restrictions must also be considered for reasoning.

Spatial reasoning approaches often refer to qualitative descriptions of spatial entities and their relations. Qualitative representations are characterized by making only as many distinctions in the domain as necessary in a given context (Hernández 1994). Typical examples are spatial relations such as "*a is inside b*", "*c is north of d*" and "*e is longer than f*". The different aspects of space, like topology, orientation, distance and shape, are usually represented by different spatial relations (Cohn and Hazarika 2001). To solve reasoning problems based on such knowledge representations, special purpose inference mechanisms have been developed. An advantage of such approach is that certain constraints which always hold in the spatial domain do not have to be modeled and verified in each situation anew (Freksa 1991). For example, once the composition of a set of topological relations has been calculated and verified, the corresponding composition table (Egenhofer 1994; Grigni et al. 1995) can be used whenever a set of these relations is analyzed. Such compact representation of knowledge should be an integral part of a spatial reasoning system. The class relation reasoning approaches discussed in this chapter also refer to qualitative descriptions and make use of the knowledge about the logical properties of the spatial relations.

In the following section a set class relations is defined, that allows for a qualitative representation of cardinality restrictions. This set is then exemplarily used to explain the reasoning properties of class relations like their symmetry, composition and conceptual neighborhood, and to demonstrate how these inference methods link to the logical properties of the instance relations. After that some open issues for class relation reasoning are discussed.

2 Cardinality Properties and Class Relation Definition

Class relations are used whenever the semantics of entire classes are described, independently of any knowledge about specific single entities. A class relation is defined in terms of an instance relation or a disjunction of instance relations in combination with a cardinality restriction. Cardinalities express the number of elements of a set. Class relations define a cardinality restriction for a certain relation between the individuals of one or more classes (Mäs 2009a).

For the definition of class relations some basic assumptions have to be fulfilled. First, every instance has to be a member of some class. Second, the involved classes must have at least one instance, that is, empty classes are not feasible. Since class relations are linked to individual relations, the third condition specifies that if a class relation is defined, there exists at least one corresponding individual relation among the instances of the classes involved. This chapter is restricted to binary relations defined between entity classes. Relations between three or more classes are not considered.

In the following definitions the lowercase letters *('a', 'b'* and *'c')* denote variables for instances or individuals. Every instance must belong to an entity class. For entity classes the capital letters *('A', 'B'* and *'C')* are used as variables. '*Inst(a, A)*' is the instantiation relation, meaning that individual *'a'* is an instance of class *'A'*. The claim *'r(a, b)'* means instance *'a'* has the relation *'r'* to instance *'b'*; *'a'* and 'b' are said to participate in the relationship instance *'r'*. The meta-variable *'r'* can stand for any binary relation between instances (e.g., a spatial or temporal relation) or for a disjunction of such relations. The validity of the binary relation depends on the properties of the instances (e.g., for spatial relations on the geometries of the instances). Instance relationships can be associated with a class relation *'R'*. For class relation definitions *'*$R_{<cp>}(A, B)$*'* denotes that *'R'* relates the classes *'A'* and *'B'*. The meta-variable *'R'* can stand for any class relationship. Every *'R'* is defined in terms of an instance relation *'r'* (same letter(s) in lower case). In formulas this is made explicit by the claim '*InstR(r, R)*'. If a class relation is defined by an *'*$R_{<cp>}(A, B)$*'*, at least one *'r'* must exist between the instances of *'A'* and *'B'*, independently of the cardinality restriction. For example, if *'*$MEET_{<cp>}(A, B)$*'* is defined, at least one *'meet(a, b)'* must exist. The placeholder '*<cp>*' stands for the cardinality properties of the class relation. In the following, class relations that are not linked to a particular instance relation are referred to as **abstract class relations** (e.g., '$R_{LD\ RD\ LT}(A, B)$'). These are only used to define generic reasoning rules. Only relations that incorporate a concrete instance relation are called class relations (e.g. '$DISJOINT_{LT}(A, B)$').

A first approach for the formal definition of such class relations has been made by Donnelly and Bittner (2005). It was based on totality cardinality restrictions of the involved classes:

$$LT(A,B,r) := \forall a(Inst(a,A) \rightarrow \exists b(Inst(b,B) \cap r(a,b))). \tag{1}$$

$$RT(A,B,r) := \forall b(Inst(b,B) \rightarrow \exists a(Inst(a,A) \cap r(a,b))). \tag{2}$$

The cardinality restrictions Eqs. (1) and (2) define a totality for the class *'A'* and *'B'* respectively. Equation (1) holds if every instance of *'A'* has the relation *'r'* to some instance of *'B'*. In set theory such relations are called **left-total**.

Equation (2) holds if for each instance of *'B'* there is some instance of *'A'* which stands in relation *'r'* to it. This means that every instance of *'B'* has the converse relation of *'r'* to some instance of *'A'*. In this case the relation is **right-total**.

In order to improve expressiveness this approach has been extended by unambiguousness cardinality restrictions in Mäs (2007):

$$LD(A,B,r) := \forall a,b,c\begin{pmatrix} Inst(a,A) \cap Inst(b,B) \cap Inst(c,A) \\ \cap\ r(a,b) \cap r(c,b) \rightarrow a = c \end{pmatrix} \cap Ex(A,B,r). \tag{3}$$

$$RD(A,B,r) := \forall a,b,c\begin{pmatrix} Inst(a,A) \cap Inst(b,B) \cap Inst(c,B) \\ \cap\ r(a,b) \cap r(a,c) \rightarrow b = c \end{pmatrix} \cap Ex(A,B,r). \tag{4}$$

$$Ex(A,B,r) := \exists a \exists b(Inst(a,A) \cap Inst(b,B) \cap r(a,b)).$$

Class relations which hold Eq. (3) are **left-definite** and specify that for no instance of *'B'* there is more than one instance of *'A'* which stands in relation *'r'* to it. This property restricts the number of *'r'* relations an instance of *'B'* can participate in; the instances of *'A'* are not restricted. The last term *'Ex(A, B, r)'* ensures that at least one instance relation *'r'* does exist between the instances of *'A'* and *'B'*.

Equation (4) specifies that no instance of *'A'* participates in a relationship *'r'* to more than one instance of *'B'*. When this cardinality property is defined in a class relation all instances of *'A'* are restricted while the instances of *'B'* are not affected. The corresponding class relations are **right-definite**.

Such cardinality restrictions are well established in data modeling and ontology engineering (Tarquini and Clementini 2008; Donnelly et al. 2006). Since the four cardinality properties are independent of each other they can be combined for the definition of a class relation. For example, a class relation which defines that "every country contains exactly one capital and every capital is contained by exactly one country" requires all four cardinality properties Eq. (5). The class relation "every building is contained by exactly one parcel" would be based on *'LT(Building, Parcel, contains)'* and *'RD(Building, Parcel, contains)'* (Mäs 2007). The other two cardinality properties are invalid Eq. (6).

$$R_{LD\,RD\,LT\,RT}(A,B) := \forall r \begin{pmatrix} InstR(r,R) \rightarrow LD(A,B,r) \cap RD(A,B,r) \cap \\ LT(A,B,r) \cap RT(A,B,r) \end{pmatrix}. \quad (5)$$

$$R_{RD\,LT}(A,B) := \forall r \begin{pmatrix} InstR(r,R) \rightarrow \neg LD(A,B,r) \cap RD(A,B,r) \cap \\ LT(A,B,r) \cap \neg RT(A,B,r) \end{pmatrix}. \quad (6)$$

Therewith a formal definition of a class relation is based on cardinality definitions as well as their negations. An investigation of all possible combinations of the four cardinality properties leads to the following categorization of abstract class relations:

- one abstract class relation where all four properties are valid (5);
- four abstract class relations that combine three of the four defined cardinality properties respectively and exclude the corresponding fourth;
- six abstract class relations where two properties are valid and the other two are excluded (e.g. 6); and
- four abstract class relations where one property is valid and the corresponding other three are excluded.

In addition to these 15 abstract class relations two special cases have been considered in Mäs (2007). The first is a strict case of a *left-total* and *right-total* relation that specifies that all instances of '*A*' must have a relationship instance of '*R*' to all instances of '*B*' (7). For class relations it is frequently occurring, for example if the instances of two classes are allowed to intersect: *'$DISJOINT_{LT\,RT\text{-}all}$(Streets, Lakes)'* (Mäs 2008).

The abstract class relation '*R_{some}(A, B)'* is defined for the situation that none of the four cardinality properties is valid, but nevertheless some instances of '*A*' stand in relation '*r*' to some instances of '*B*' (8). 'R_{some}*(A, B)'* is defined as *not left-total* and *not right-total*, which implies that some instances of '*A*' and '*B*' participate in a relation '*r*' to an instance of '*B*' and '*A*' and some do not. Furthermore the exclusions of '*LD(A, B, r)*' and '*RD(A, B, r)'* specify that some '*A*' and some *'B'* participate in a relation r to at least two instances of '*B*' and '*A*'. All cardinalities from "2" till "all-1" are valid for both classes. Therefore, it is a relatively imprecise representation of all cardinalities that the other 16 abstract class relations do not cover.

$$R_{LT\,RT-all}(A,B) := \forall r \forall a \forall b \begin{pmatrix} InstR(r,R) \cap Inst(a,A) \cap Inst(b,B) \\ \rightarrow r(a,b) \end{pmatrix}. \quad (7)$$

$$R_{some}(A,B) := \forall r \begin{pmatrix} InstR(r,R) \rightarrow \neg LD(A,B,r) \cap \neg RD(A,B,r) \cap \\ \neg LT(A,B,r) \cap \neg RT(A,B,r) \cap Ex(A,B,r) \end{pmatrix}. \quad (8)$$

All together the 17 abstract class relations are a jointly exhaustive set of relations. They enable the definition of class relations based on any binary instance relation. The set of 17 abstract class relations allows for a qualitative description of all possible (indefinitely many) cardinality properties. Only class relations, that base on the four cardinality properties Eqs. (1–4) or $R_{LT\ RT\text{-}all}$*(A, B)* (7) can precisely be defined. Further details on the definition of the abstract class relations can be found in Mäs (2007 and 2009a). Other notations, like for example Entity-Relationship Diagrams, are more expressive with regard to the possible cardinality constraints. However, some of the reasoning concepts investigated in the following sections, require a discrete set of abstract class relations. Also, it is assumed that the introduced set of abstract class relations can precisely represent a majority of the class relations used in practice. Nevertheless, this set of abstract class relations is only exemplarily used here and the reasoning approaches can also be transferred to other sets of relations.

3 Transferring Logical Properties of Instance Relations to Class Relations

In general, the logical properties of class relations, such as their symmetry, transitivity and reflexivity, depend not only on the properties of the applied instance relation, but also on the cardinality restrictions. Donnelly and Bittner (2005) have studied the transfer of logical properties of instance relations to class relations. It has been shown that some, but not all, logical properties of the instance relations transfer to the class relations.

For the set of class relations defined in the previous section only the symmetry/ converseness has been sufficiently researched yet. In Mäs (2007) it has been shown, how the converse of a class relation can be defined, if the converse relation of the corresponding instance relation is known. The converse of a class relation bases on the converse of the applied instance relation. If an abstract class relation is *left-total/left-definite* the converse relation is *right-total/right-definite*, and vice versa. The relations 'R_{some}' and '$R_{LT\ RT\text{-}all}$' are symmetric. Table 1 summarizes this correlation between symmetry/converseness of instance relations and those of the corresponding class relations.

The following examples demonstrate the derivation of converse class relations. The class relations base on the symmetric instance relation '*overlap*' and the converse relations '*contains*' and '*inside*':

$$OVERLAP_{RD\,LT}(Watermill,\ Stream)^i := OVERLAP_{LD\,RT}(Stream,\ Watermill).$$

$$CONTAINS_{LD\,RD\,LT\,RT}(Country,\ Capital)^i := INSIDE_{LD\,RD\,LT\,RT}(Capital,\ Country).$$

The examples show that not all class relations are symmetric, even if they are based on symmetric instance relations.

In a recent paper Egenhofer (2011) researched the inference of complements of class relations. Therefore, the applied instance relations must be part of a jointly exhaustive and pair wise disjoint (JEPD) set of relations. The complement of a class relation captures the relations that must hold between all instances of the related classes other than the instance relations covered by the original class relation. For example, the complement of the class relation "*every building is inside or covered by a land parcel*" captures the relations between all buildings and land parcels other than the hosts of the buildings: "*every building meets or is disjoint from land parcels (it is not hosted by)*".

For other logical properties of the class relations, such as antisymmetry, transitivity and reflexiveness a detailed analysis is still missing.

4 Composition of Class Relations

The composition of binary relations enables the derivation of implicit knowledge about a triple of entities. If two binary relations are known, the corresponding third

Table 1 Symmetry/converseness of the class relations (Mäs 2007)

Individual relation '*r*' is…	Class relation '*R*' is…					
	Left-definite	Right-definite	Left-total	Right-total	R_{some}	$R_{LTRT-all}$
	Converse class relation 'R^i' is…					
Symmetric	*R* right-definite	*R* left-definite	*R* right-total	*R* left-total	R_{some}	$R_{LTRT-all}$
Not symmetric	R^i right-definite	R^i left-definite	R^i right-total	R^i left-total	R^i_{some}	$R^i_{LTRT-all}$

r^i converse instance relation
R^i converse class relation (defined in terms of 'r': '*InstR(r^i ,R^i)*')

one can potentially be inferred, or at least some of the possible relations can be excluded. Examples of composition tables of instance relations can be found in Egenhofer (1994) and Grigni et al. (1995) for topological relations between areal entities and in Hernández (1994) and Freksa (1992a) for directional/orientation relations. Many other sets of binary spatial relations also allow for such derivations. As the examples in the introduction of this chapter illustrate, a transfer of this reasoning formalism to the class level is very useful for work with geographical data, but also for many other application domains. In the example two class relations have been defined:

- *airports* and *forests* are either disjoint or meet.
- every *airport* contains at least one *airport tower* and every *airport tower* is contained by an *airport*.

The composition of these two class relations leads to:

- *forests* and *airport towers* are disjoint.

It is obvious that the composition depends on the composition of the applied instance relations, but the cardinality restrictions also have an influence. In Mäs (2008) a two level reasoning formalism has been proposed, which separates the compositions of the abstract class relations from those of the instance relations. Therewith the composition of the abstract class relations can be defined independently of a concrete set of instance relations. The composition table of the 17 abstract class relations is shown in Fig. 2.

To illustrate the two levels reasoning formalism and the use of the composition table the introductory example of Fig. 1 is used. The composition of the instance relations of the three entities forest *f1*, airport *a*1 and airport tower *t1* is in this case (Egenhofer 1994):

$$meet \cup disjoint(f1, a1);\ contains(a1, t1) \rightarrow disjoint(f1, t1).$$

The composition of the abstract class relations is provided by the composition table in Fig. 2 (row 17, column 14):

$$R1_{LT\,RT-all}(Forest, Airport);\ R2_{LD\,LT\,RT}(Airport, Airport\,Tower) \\ \rightarrow R3_{LT\,RT-all}(Forest, Airport\,Tower).$$

The combination of the compositions of the two levels results in:

$$[MEET \cup DISJOINT]_{LT\,RT-all}(Forest, Airport); \\ CONTAINS_{LD\,LT\,RT}(Airport, Airport\,Tower) \rightarrow \\ DISJOINT_{LT\,RT-all}(Forest, Airport\,Tower)$$

Since the compositions of both levels have a unique result the combined composition is also unique. For a more detailed explanation and further examples of the composition see Mäs (2007 and 2008).

	1. Relation \ 2. Relation	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	LD.RD	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	1	$\mathcal{U}$	$\mathcal{U}$	1,3	1,2	$\mathcal{U}$	1,2, 3,4	$\mathcal{U}$	1	1,2	1,3, 6,10	1,2,3, 4,6,7, 10,12	6, 7, 10,12
2	LD	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	2	$\mathcal{U}$	$\mathcal{U}$	1,2, 3,4	2	$\mathcal{U}$	1,2, 3,4	$\mathcal{U}$	2	2	1,2,3,4, 6,7,10, 12	1,2,3, 4,6,7, 10,12	6, 7, 10,12
3	RD	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	3	$\mathcal{U}$	$\mathcal{U}$	3	3,4	$\mathcal{U}$	3,4	$\mathcal{U}$	3	3,4	3,10	3,4, 10,12	10,12
4	some	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	4	$\mathcal{U}$	$\mathcal{U}$	3,4	4	$\mathcal{U}$	3,4	$\mathcal{U}$	4	4	3,4, 10,12	3,4, 10,12	10,12
5	LD.RD.LT	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	5	$\mathcal{U}$	$\mathcal{U}$	5,8	5,9	$\mathcal{U}$	5,8, 9,11	$\mathcal{U}$	5	5,9	5,8, 13,15	5,8,9, 11,13,14, 15,16,17	17
6	LD.RD.RT	1	2	3	4	1	6	7	3	2	10	4	12	6	7	10	12	6, 7, 10,12
7	LD.RT	1,2	2	1,2, 3,4	2,4	2	6,7	7	1,2, 3,4	2	6,7, 10,12	2,4	7,12	7	7	6,7, 10,12	7,12	6, 7, 10,12
8	RD.LT	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	8	$\mathcal{U}$	$\mathcal{U}$	8	8,11	$\mathcal{U}$	8,11	$\mathcal{U}$	8	8,11	8,15	8,11,15, 16,17	17
9	LD.LT	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	9	$\mathcal{U}$	$\mathcal{U}$	5,8, 9,11	9	$\mathcal{U}$	5,8, 9,11	$\mathcal{U}$	9	9	5,8,9,11, 13,14,15 16,17	5,8,9,11, 13,14,15, 16,17	17
10	RD.RT	1,3	2,4	3	4	3	6,10	7,12	3	4	10	4	12	10	12	10	12	10,12
11	LT	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	$\mathcal{U}$	11	$\mathcal{U}$	$\mathcal{U}$	8,11	11	$\mathcal{U}$	8,11	$\mathcal{U}$	11	11	8,11, 15,16, 17	8,11, 15,16, 17	17
12	RT	1,2 3,4	2,4	1,2, 3,4	2,4	4	6,7, 10,12	7,12	3,4	4	6,7, 10,12	4	7,12	12	12	10,12	12	10,12
13	LD.RD.LT.RT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
14	LD.LT.RT	1,2, 5,9	2,9	1,2,3, 4,5,8, 9,11	2,4, 9,11	9	6,7, 13,14	7,14	5,8, 9,11	9	6,7,10, 12,13,14, 15,16,17	9,11	7,12, 14,16, 17	14	14	13,14, 15,16, 17	14,16, 17	17
15	RD.LT.RT	1,3	2,4	3	4	8	6,10	7,12	8	11	10	11	12	15	16	15	16	17
16	LT.RT	1,2,3, 4,5,8, 9,11	2,4, 9,11	1,2,3, 4,5,8, 9,11	2,4, 9,11	11	6,7,10, 12,13,14, 15,16,17	7,12, 14,16, 17	8,11	11	6,7,10, 12,13,14, 15,16,17	11	7,12, 14,16, 17	16	16	15,16, 17	16,17	17
17	LT.RT-all	5,8, 9,11	9,11	5,8, 9,11	9,11	5,8, 9,11	17	17	5,8, 9,11	9,11	17	9,11	17	17	17	17	17	17

Fig. 2 Composition table of the 17 abstract class relations (Mäs 2008), the *numbers* are defined in the *left column*

To allow for a more convenient use of the compositions of the abstract class relation some of them can be summarized by general rules, which deduce the composition results directly from cardinality properties. In Mäs (2008) some obvious rules were defined. The corresponding compositions are highlighted in grey in Fig. 2. Examples of these rules are:

- If the first abstract class relation is not right-total and the second relation is not left-total the composition is always a universal disjunction $\mathcal{U}$.
- If the first relation holds '$R1_{LT\ RT\text{-}all}(A, B)$' and the second is right-total the composition is always '$R3_{LT\ RT\text{-}all}(A, C)$'.
- If the first relation is left-total and the second holds '$R2_{LT\ RT\text{-}all}(B, C)$' the composition is always '$R3_{LT\ RT\text{-}all}(A, C)$'.

A set of rules that completely represents the composition table is subject to further research. Such rule set could enhance the understanding of the class relation compositions and would prove the correctness and completeness of the contents of the composition table. Furthermore, it would make the compositions transferable to other sets of abstract class relations and the composition tables of different sets comparable, respectively.

5 Conceptual Neighborhood of Class Relations

The notion of conceptual neighborhood has been introduced by Freksa (1992b). It represents continuous transformations between relations by linking relations that are connected by an atomic change. Examples of conceptual neighborhood networks of spatio-temporal relations can be found for temporal interval relations (Freksa 1992b; Hornsby et al. 1999), for topological relations between regions (Egenhofer and Al-Taha 1992), between regions and lines (Egenhofer and Mark 1995), and between directed lines (Kurata and Egenhofer 2006). The conceptual neighborhood of class relations has been introduced by Mäs (2008). In this approach, two class relations are considered as conceptually neighbored if they are linked to the same instance relation and they differ only in a single instance relation between two entities. The number of instances of the classes is considered fixed.

In Fig. 3 the conceptual neighborhood of '$R_{LD\ RD}(A, B)$' and '$R_{RD}(A, B)$' is exemplarily illustrated. All arrows symbolize one instance relation of the same kind '*r*' (again: '*InstR(r, R)*'). In the example, the addition of a further instance relation between the instances '*a2*' and '*b*1' in the right box leads to a transition from '$R_{LD\ RD}$' to 'R_{RD}' (row 1, column 3 in Table 2). An addition of an instance relation between other instances can lead to other transitions. The 17 class relations have 45 conceptual neighborhoods. Additionally nine class relations are conceptual neighbors of themselves (Table 2).

Since the conceptual neighborhood is defined through the addition or removal of a single instance relation, all neighborhoods are directed. Table 2 represents the

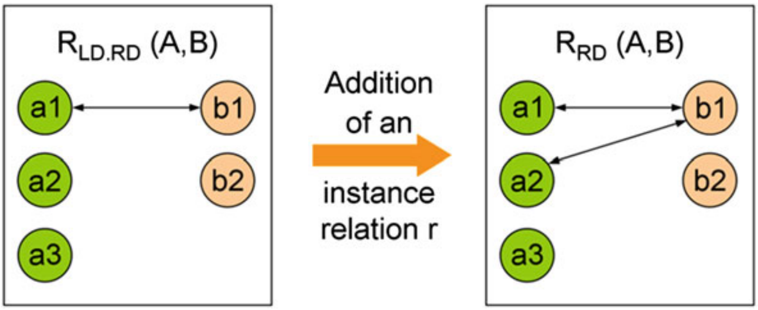

Fig. 3 Conceptual neighborhood between '$R_{LD\ RD}(A, B)$' and '$R_{RD}(A, B)$' (Mäs 2009a)

Table 2 Conceptual neighborhood between the class relations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1 LD.RD	±	+	+	+	+	+	+	+					+				
2 LD	−	±		+			+		+		+			+			
3 RD	−		±	+				+		+		+			+		
4 some	−	−	−	±							+	+				+	
5 LD.RD.LT	−								+		+			+			+
6 LD.RD.RT	−									+		+			+		+
7 LD.RT	−	−										+				+	
8 RD.LT	−		−								+					+	
9 LD.RT		−			−				±		+			+			+
10 RD.RT			−			−				±		+			+		+
11 LT		−		−	−			−	−		±					+	
12 RT			−	−		−	−			−		±				+	
13 LD RD LT RT	−															+	
14 LD.LT.RT		−			−				−							+	
15 RD.LT.RT			−			−				−						+	
16 LT.RT				−			−	−			−	−	−	−	−	±	+
17 LT.RT-all					−	−			−	−						−	

± corresponds to neighborhood through addition/removal of an instance (Mäs 2009a), numbers are defined in the left column

neighborhoods, which result from an addition ('+') and those which result from a removal ('−'). The symbol '±' marks class relations that are conceptual neighbors of themselves. If an addition or removal of an instance relation has changed a class relation it is impossible to get the same class relation again by further adding/removing of instance relations. The addition of instance relations ultimately leads to a '$R_{LT\ RT\text{-}all}$' class relation. A removal leads to '$R_{LD\ RD}$'.

In Mäs (2008 and 2009a) some practical examples of the conceptual neighborhood of class relations and its relevance for the reasoning on class relation compositions have been discussed. For example, if three class relations hold for the classes '*A*', '*B*' and '*C*': '$MEET_{some}(A, B)$', '$CONTAINS_{LD\ RD\ LT\ RT}(B, C)$' and '$DISJOINT_{LT\ RT}(A, C)$'. These relations are analyzed for conflicts through the comparison of the class relation composition '*R(A,B);R(B,C)*' with the given '*R(A, C)*'. The compositions of the corresponding instance and abstract class relations are:

$$meet(a1, b1);\ contains(b1, c1) \rightarrow disjoint(a1, c1).$$

$$R1_{some}(A, B);\ R2_{LD\,RD\,LT\,RT}(B, C) \rightarrow R3_{some}(A, C).$$

The combination of the compositions of the two levels results in: '$DISJOINT_{some}(A, C)$' (row 4, column 13 in Fig. 2), which seems to be in conflict with the given third relation '$DISJOINT_{LT\ RT}(A, C)$'. Figure 4 exemplarily illustrates this situation. Figure 4a shows a possible setting of instance relations between the classes '*A*' to '*B*' and '*B*' to '*C*', and Fig. 4b the inferred relations between '*A*' and '*C*'. Thereby

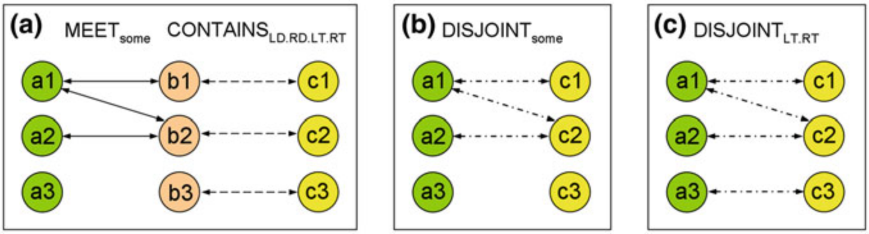

Fig. 4 Use of the conceptual neighborhood for the composition of class relations (Mäs 2009a)

only those instances of '*A*' and '*C*' are related in Fig. 4b, which are connected via an instance of '*B*' in Fig. 4a. In comparison to this, the Fig. 4c shows that the given relation '$DISJOINT_{LT\ RT}(A, C)$' possibly differs from '$DISJOINT_{some}(A, C)$' by only one '*disjoint*' instance relation (in this case '*a3*' to '*c3*'). Thus '$DISJOINT_{some}$' and '$DISJOINT_{LT\ RT}$' are conceptual neighbors (row 4, column 16 in Table 2).

The three instance relations in Fig. 4b are implied by the relations shown in Fig. 4a. The composition does not allow for any conclusion about further relations between the instances of '*A*' and '*C*'. Also, it cannot be excluded that further pairs of '*A*' and '*C*' instances are '*disjoint*'. Hence the composition of '$MEET_{some}(A,B)$' and '$CONTAINS_{LD\ RD\ LT\ RT}(B, C)$' does not contradict '$DISJOINT_{LT\ RT}(A,C)$' and the given triple of class relations is consistent. Beside '$DISJOINT_{LT\ RT}$', also the other direct conceptual neighbors of '$DISJOINT_{some}$' '$DISJOINT_{LT}$' and '$DISJOINT_{RT}$' (row 4, columns 11 and 12 in Table 2), as well as '$DISJOINT_{LT\ RT\text{-}all}$' as a conceptual neighbor of '$DISJOINT_{LT\ RT}(A,C)$' (row 16, columns 17 in Table 2) have no conflict.

Mäs (2008) correspondingly concluded with the generic rule: a class relation '*R3*' is not in conflict with a composition '*R1; R2* → *R3**' if '*R3**' and '*R3*' base on the same instance relation '*r3*' ('*InstR(r3, R3)*' and '*InstR(r3, R3*)*') and the addition of further '*r3*' instance relations to '*R3**' can lead to a transition to class relation '*R3*'. For this the result of the composition '*R3**' and '*R3*' do not need to be direct conceptual neighbors. There can also be further class relation transitions between the two class relations. Nevertheless the conceptual neighborhood points out which '*R3*' class relations are valid, since it shows which transitions are possible for a certain class relation '*R3**'.

6 Constraint Satisfaction Problems in Class Relation Networks

The previous sections discussed the reasoning properties of class relations. The application of these reasoning techniques for checking consistency in networks of class relations is a constraint satisfaction problem (CSP). Such detection of conflicts and redundancies in sets of class relations requires a network graph, in which the nodes represent the entity classes and the edges represent the class relations. In a consistent network of jointly exhaustive and pair wise disjoint (JEPD) relations, the following three requirements are fulfilled (Rodriguez et al. 2004). Proofs of these requirements for class relation networks have been discussed in Mäs (2007 and 2009a):

- Node consistency is ensured if every node has a self-loop arc, which corresponds to the identity relation (i.e., relation of an entity/entity class to itself). If a corresponding identity instance relation is available the identity class relation is in general '$R_{LD\ RD\ LT\ RT}(A, A)$'; for example '$EQUAL_{LD\ RD\ LT\ RT}(A, A)$' when using the topological relations areal entities.
- Arc consistency is ensured if every edge of the network has an edge in the reverse direction, that is, every relation has a converse relation that is consistent with the network. As shown in Sect. 3, the converse of a class relation can be defined, if the converse relation of the corresponding instance relation is known.
- Path consistency is ensured if all relations are consistent with their induced relations, determined by the corresponding intersection of all possible composition paths of length two (n = number of nodes):

$$\forall_{i,j} r_{i,j} = \bigcap_{k=1}^{n} r_{i,k};\ r_{k,j}.$$

In general, the algorithm for checking path consistency of a class relation network is similar to algorithms used for the instance relations (e.g. Allen 1983; Hernández 1994). However, due to the higher complexity of the relations, the detection whether two relations between the same classes are in conflict is more extensive (Mäs 2009a).

7 Reasoning on Class Relations in Class Hierarchies

So far, the existing reasoning approaches only consider classes without a hierarchical structure. For a class relation network this means that all nodes in the graph are considered at the same level and all edges are treated equally. For a practical application this is insufficient, since most data models and ontologies are hierarchically structured. The hierarchical organization makes it easy to distribute properties, since the properties and methods of a class depend on the properties of its superclass(es). The properties that are shared by a superclass and all its subclasses are defined only once with the superclass. The subclasses inherit all properties of their parent-/superclasses in the hierarchy (Brachman 1983; Egenhofer and Frank 1992). Such properties can be spatio-temporal or thematic attributes or explicitly defined relations between classes. If a class relation is defined in a class hierarchy it has an influence on the classes at the lower levels. It also results in additional logical rules and consistency requirements between the class relations of the different hierarchy levels. For example from: "*Watercourse* is a subclass of *Waterbody*" and

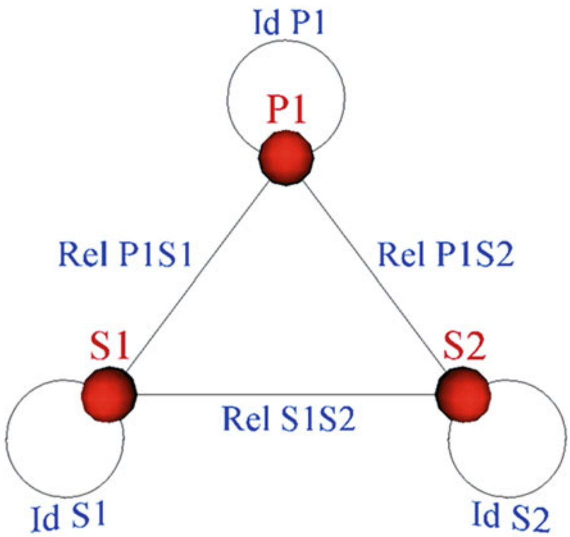

Fig. 5 Class relation network of a simple hierarchy with one superclass and two subclasses

"every *Waterfall* is an individual part of some *Watercourse*" it can be derived that "every *Waterfall* is an individual part of some *Waterbody*" (analogous to Bittner et al. 2009). Similar examples from the biomedical domain can be found in Donnelly et al. (2006). Although these papers provide some inference rules they only consider the inference of specific class relations. A generic solution is subject to future research. Therefore, the remainder of this section shall illustrate some obvious dependencies of the class relations between super- and subclasses in a class hierarchy.

For the following propositions it is assumed that all subclasses of a superclass are known and that all entities of the superclass belong to (exactly) one of the subclasses and vice versa. These assumptions are commonly made in ontologies (e.g. Bittner et al. 2009). Further, all subclasses have only one superclass, i.e., multiple inheritance is not considered here.

In a simple hierarchy with one superclass and two subclasses there are three edges between the classes and three edges for the identity class relations (Fig. 5). The superclass (P1) subsumes the subclasses (S1 and S2).[1] The class relations define cardinality restrictions of instance relations between the entities of the classes. Therefore class relations at a higher level or between the two successive levels subsume the class relations of the lower level. This leads to the following dependencies:

D1: The identity relation of the superclass (IdP1) subsumes the identity class relations of the subclasses (IdS1 and IdS2) and the class relations between the subclasses in both directions (RelS1S2 and RelS2S1) (see Fig. 6).

[1] Please be aware that all edges in Fig. 5 represent class relations that define cardinality restrictions and not the inheritance/generalisation relation between the classes.

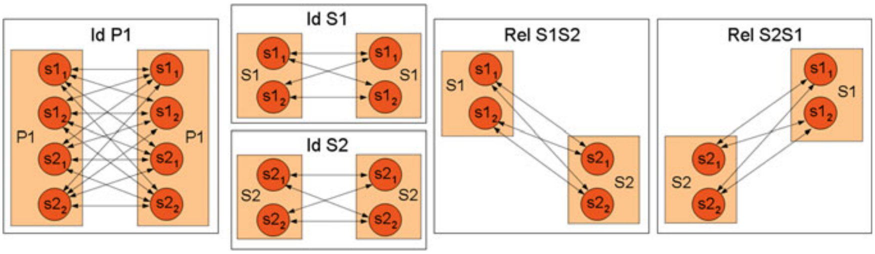

Fig. 6 Example for the subsumption of class relations of a lower hierarchy level in the identity relation of the corresponding superclass

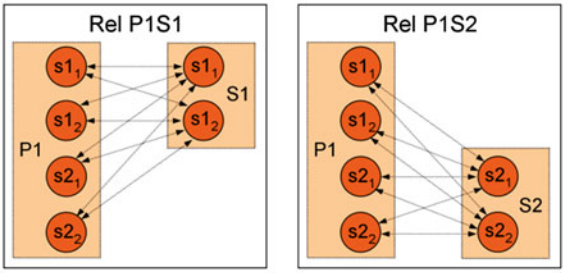

Fig. 7 Example class relations that connect the two hierarchy levels

D2: The class relations that connect the two hierarchy levels subsume the identity relation of the related subclass and the class relations of all other subclasses to this subclass. For the scene shown in Fig. 5 this means that RelP1S1 subsumes IdS1 and RelS2S1, and RelP1S2 subsumes IdS2 and RelS1S2 (see Figs. 6 and 7).

D3: Therewith the identity relation of the superclass (IdP1) also subsumes all class relations that connect the two hierarchy levels (RelP1S1 and RelP1S2).

Figures 6 and 7 schematically illustrate the class relations of the hierarchy of Fig. 5. In the figures both subclasses have two entities. Again, the arrows represent the relations between the entities, which could be for example one of the topological relations '*intersect*' or '*disjoint*'. Through the comparison of the arrows it is easy to comprehend the three interdependencies.

The following Eqs. (9)–(11) define the interdependencies D1–D3 for hierarchies with arbitrary many subclasses. The variables m and n are indices of the subclasses. Equations (12) and (13) define the interdependency for the class relations from the subclasses to the superclass (converse to the relations considered in Eqs. (10) and (11)).

$$\mathrm{Id}_{\mathrm{P}} = \{Id_{S_1}, \ldots, Id_{S_m}, Rel_{S_1S_2}, Rel_{S_2S_1}, \ldots, Rel_{S_mS_n}, Rel_{S_nS_m}\}_{m \neq n} \tag{9}$$

$$Rel_{PS_m} = \{Id_{S_m}, Rel_{S_1S_m}, \ldots, Rel_{S_nS_m}\}_{m \neq n} \tag{10}$$

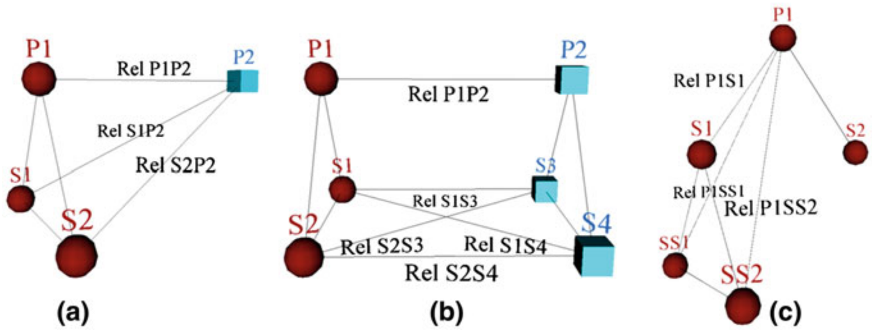

Fig. 8 Hierarchical settings with interdependent class relations

$$Id_P = \{Rel_{PS_1}, \ldots, Rel_{PS_m}\} \tag{11}$$

$$Rel_{S_mP} = \{Id_{S_m}, Rel_{S_mS_1}, \ldots, Rel_{S_mS_n}\}_{m \neq n} \tag{12}$$

$$Id_P = \{Rel_{S_1P}, \ldots, Rel_{S_mP}\} \tag{13}$$

Similar interdependencies can be found for class relations between a class hierarchy and a class outside of the hierarchy (Fig. 8a), class relations that connect two independent class hierarchies (Fig. 8b) and class relations between three hierarchy levels (Fig. 8c).

These observations show that there are dependencies between the class relations of different hierarchy levels. The interdependencies of relations in class hierarchies result from the fact that the class relations at a higher hierarchy level subsume those of the lower levels. The entities of the classes of the subsuming relation are divided between the subclasses. Correspondingly, **class relation subsumption** means that the subsuming relation includes all instance relations of the subsumed class relations.

At this stage the constraints that the subsumption exposes on concrete class relations are not defined. A detailed set of rules to describe the dependency for a set of abstract class relations or directly for cardinality properties is subject to future research. These rules will extend the consistency requirements in class relations networks, which are implied by their compositions and converses (Sect. 6).

The four elementary hierarchical settings (Figs. 5 and 8) and their interdependencies can serve as building blocks for an analysis of more complex class hierarchies. Therefore the constraints that are exposed by the interdependencies must be defined independently of the number of involved subclasses and also independent of the number of involved hierarchy levels.

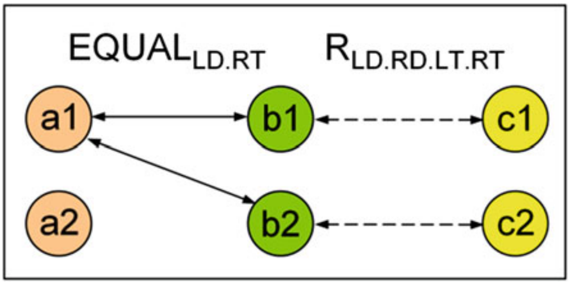

Fig. 9 Inconsistent combination of class relations

8 Further Open Issues

The described inference approaches separately analyze the reasoning properties of the abstract class relations and those of the instance relations. However, some combinations of instance and abstract class relations lead to conflicts that cannot be found in this way. For example, the combination of *'$EQUAL_{LD\ RT}$(A, B)'* and the abstract class relation *'$R_{LD\ RD\ LT\ RT}$(B, C)'* is impossible (Fig. 9). This is due to the specific identity properties of the '*equal*' instance relation and the cardinality properties of the two abstract class relations.

'$EQUAL_{LD\ RT}$(A, B)' requires at least one instance of '*A*' that is equal to at least two instances of '*B*', because it is defined as not *right-definite* (Eq. 4). Since equal is symmetric and transitive this implies that the corresponding '*B*' instances are also equal ('*b1*' equals '*b2*' in Fig. 9). Thus if one of these '*B*' instances has an instance relation to a '*C*' instance, the other '*B*' must have the same relation to this '*C*'. This means for the scene in Fig. 9 that both '*B*' should have the same instance relation to both '*C*'. This is in conflict with *'$R_{LD\ RD\ LT\ RT}$(B, C)'*, because this class relation is defined as *left-definite* and *right-definite* (Eq. 5). A generic description of such conflicts is the subject of further research.

9 Conclusion

The interoperable exchange of data of different domains and application areas requires semantic descriptions of the data. The explicit knowledge about logical properties and interrelations between relations is fundamental for automated reasoning based on such descriptions (Bittner et al. 2009). The proposed class relations and reasoning methods provide a basis for the formalization of such knowledge. Nevertheless, the formal definition of class relations and their logical properties are still hardly researched yet.

A relation among the classes is not subject to the same logical properties as the applied relation between instances because the cardinality restrictions must also be considered for reasoning. In general, the reasoning algorithms at the class level are similar to those of the instance relations. However, due to the higher complexity of the class relations, the detection of conflicts and redundancies is more extensive. This chapter summarizes current research results with regard to class relation reasoning based on properties such as symmetry, composition and conceptual neighborhood. Therefore a set of 17 abstract class relations has been exemplarily used. This shall provide a basic framework, which can be extended for other possibly more complex types of class relations. Furthermore, future research should particularly concern the dependencies of class relations in class hierarchies, since the class concepts described in data models or ontologies are usually hierarchically structured.

Class relations can be used in combination with any type of instance relation. To enable a flexible use of reasoning algorithms, the inference rules must be defined in a generic way. This means they must hold for abstract class relations or directly for cardinality properties and separately integrate the logical properties of the instance relations.

A major advantage of class relations is their logical soundness. Their logical properties allow for the detection of conflicts and redundancies in sets of class relations. This is of interest for many application areas, for example for the management of spatial semantic integrity constraints (Mäs 2007), geospatial ontologies (Bittner et al. 2009), conceptual data modeling and usability evaluation (Mäs 2009b).

Most of the discussed reasoning algorithms have been implemented in a research prototype that is available at http://www.stephanmaes.de/classrelations.html. The tool is implemented as a plug-in of the Protégé ontology editor.

Acknowledgments Parts of this chapter have been written while the author was working at the AGIS at the University of the Bundeswehr Munich, Germany. The support from Wolfgang Reinhardt during that time is gratefully acknowledged. Also, the author likes to acknowledge the fruitful discussions with Max J. Egenhofer. Thanks are due to the two anonymous reviewers for their suggestions and critical comments that helped to improve this chapter.

References

Allen JF (1983) Maintaining knowledge about temporal intervals. Commun ACM 26(11):832–843

Agarwal P (2005) Ontological considerations in GIScience. Int J Geogr Inf Sci 19(5):501–536

Baumeister J, Seipel D (2006) Verification and refactoring of ontologies with rules. In: Staab S, Svátek V (eds) Managing knowledge in a world of networks, 15th international conference, EKAW 2006, LNCS 4248. Springer, Podebrady, pp 82–95

Bittner T, Donnelly M, Smith B (2009) A spatio-temporal ontology for geographic information integration. Int J Geogr Inf Sci 23(6):765–798

Brachman RJ (1983) What IS-A is and isn't: an analysis of taxonomic links in semantic networks. Computer 16(10):30–36

Cohn AG, Hazarika SM (2001) Qualitative spatial representation and reasoning: an overview. Fundamenta Informaticae 46(1–2):1–29

Donnelly M, Bittner T (2005) Spatial relations between classes of individuals. In: Cohn AG, Mark DM (eds) COSIT 2005. LNCS, vol 3693. Springer, Heidelberg, pp 182–199

Donnelly M, Bittner T, Rosse C (2006) A formal theory for spatial representation and reasoning in biomedical ontologies. Artif Intell Med 36(1):1–27

Egenhofer MJ, Herring J (1991) Categorizing binary topological relationships between regions, lines, and points in geographic databases. Technical report, Department of Surveying Engineering, University of Maine, Orono, ME

Egenhofer MJ, Frank AU (1992) Object-oriented modeling for GIS. J Urban Reg Inf Syst Assoc 4(2):3–19

Egenhofer MJ, Al-Taha KK (1992) Reasoning about gradual changes of topological relationships. In: Theories and methods of spatio-temporal reasoning in geographic space, International conference GIS—from space to territory: theories and methods of spatio-temporal reasoning, LNCS, vol 639, pp 196–219, Springer

Egenhofer M (1994) Deriving the composition of binary topological relations. J Vis Lang Comput 5(2):133–149

Egenhofer MJ, Mark DM (1995) Modelling conceptual neighbourhoods of toplogical line-region relations. Int J Geogr Inf Syst 9(5):555–565

Egenhofer M (2011) Reasoning with complements. In: De Troyer O, Bauzer Medeiros C, Billen R, Hallot P, Simitsis A, Van Mingroot H (eds) Advances in conceptual modeling. Recent developments and new directions; ER 2011, vol 6999 of Lecture notes in computer science. Springer, San Diego, pp 261–270

Frank AU (1992) Qualitative spatial reasoning about distances and directions in geographic space. J Vis Lang Comput 3:343–371

Freksa C (1991) Qualitative spatial reasoning. In: Mark DM, Frank AU (eds) Cognitive and linguistic aspects of geographic space. Kluwer Academic Press, Dordrecht, pp 361–372

Freksa C (1992a) Using orientation information for qualitative spatial reasoning. In: Frank AU et al (eds) Theories and methods of spatio-temporal reasoning in geographic space. LNCS, vol 639. Springer, Pisa, pp 162–178

Freksa C (1992b) Temporal reasoning based on semi-intervals. Artif Intell 54(1):199–227

Grigni M, Papadias D, Papadimitriou C (1995) Topological inference. In: Proceedings of the International Joint Conference of Artificial Intelligence (IJCAI), pp 901–906

Hernández D (1994) Qualitative representation of spatial knowledge, LNCS, vol 804 of Lecture notes in artifcial intelligence. Springer, Heidelberg

Hornsby K, Egenhofer MJ, Hayes PJ (1999) Modeling cyclic change. In: Chen PP et al (eds) Advances in conceptual modeling: Proceedings of ER '99 workshops on evolution and change in data management, reverse engineering in information systems, and the world wide web and conceptual modeling, LNCS, vol 1727. Springer, Paris, pp 98–109

Kurata Y, Egenhofer MJ (2006) The head-body-tail intersection for spatial relations between directed line segments. In: Raubal M et al (eds) Geographic information science, 4th international conference, LNCS, vol 4197. Springer, Berlin, pp 269–286

Mäs S, Wang F, Reinhardt W (2005) Using ontologies for integrity constraint definition. In: Wu L et al (eds) Proceedings of the 4th international symposium on spatial data quality, ISSDQ 2005, pp 304–313

Mäs S (2007) Reasoning on spatial semantic integrity constraints. In: Winter S et al (eds) COSIT 2007. LNCS, vol 4736. Springer, Heidelberg, pp 285–302

Mäs S (2008) Reasoning on spatial relations between entity classes. In: Cova TJ et al (eds) 5th international conference geographic information science (GIScience), Park City, Utah, USA; LNCS, vol 5266, pp 234–248

Mäs S (2009a) On the consistency of spatial semantic integrity constraints. PhD thesis, University of the Bundeswehr, Munich (published in: Dissertations in Geographic Information Science, GISDISS 004; IOS Press)

Mäs S (2009b) Harmonisation of spatial semantic integrity constraints. In: AGILE 2009 pre-conference workshop "challenges in geospatial data harmonisation", Hannover, Germany

Mäs S, Reinhardt W (2009) Categories of geospatial and temporal integrity constraints. In: International conference on advanced geographic information systems and web services, GEOWS 2009, pp 146–151, IEEE Computer Society

Rodriguez A, de Weghe NV, Maeyer PD (2004) Simplifying sets of events by selecting temporal relations. In: Egenhofer MJ et al (eds) Geographic information science. Third international conference, GIScience 2004, LNCS, vol 3234. Springer, Heidelberg, pp 269–284

Tarquini F, Clementini E (2008) Spatial relations between classes as integrity constraints. Trans GIS 12(1):45–57