Vietnam J Comput Sci (2015) 2:149–167
DOI 10.1007/s40595-015-0040-2

REGULAR PAPER

Bisimulation-based concept learning for information systems in description logics

Thanh-Luong Tran · Linh Anh Nguyen · Thi-Lan-Giao Hoang

Received: 11 December 2014 / Accepted: 2 February 2015 / Published online: 1 March 2015

Abstract In description logic-based information systems, objects are described not only by attributes but also by binary relations between them. This work studies concept learning in such information systems. It extends the bisimulation-based concept learning method of Nguyen and Szałas (Rough sets and intelligent systems. Springer, Berlin, pp 517–543, 2013). We take attributes as basic elements of the language. Each attribute may be discrete or numeric. A Boolean attribute is treated as a concept name. This approach is more general and suitable for practical information systems based on description logic than the one of Nguyen and Szałas (Rough sets and intelligent systems. Springer, Berlin, pp 517–543, 2013). As further extensions, we also allow data roles and the concept constructors "functionality" and "unqualified number restrictions". We formulate and prove an important theorem on basic selectors. We also present a domain partitioning method based on information gain that has been used for our implementation of the method. Apart from basic selectors and simple selectors, we introduce a new kind of selectors, called extended selectors. The evaluation results show that our concept learning method is valuable and extended selectors support it significantly.

Keywords Concept learning · Description logic · Bisimulation

T.-L. Tran · T.-L.-G. Hoang
College of Sciences, Hue University, 77 Nguyen Hue, Hue city, Vietnam
e-mail: ttluong@hueuni.edu.vn

T.-L.-G. Hoang
e-mail: hlgiao@hueuni.edu.vn

L. A. Nguyen (✉)
Division of Knowledge and System Engineering for ICT, Ton Duc Thang University, No. 19, Nguyen Huu Tho Street, Tan Phong Ward, District 7, Ho Chi Minh City, Vietnam
e-mail: nguyenanhlinh@tdt.edu.vn; nguyen@mimuw.edu.pl

L. A. Nguyen
Institute of Informatics, University of Warsaw, Banacha 2, 02-097 Warsaw, Poland

1 Introduction and motivation

Semantic Technologies have intensively been investigated and applied in many areas such as Bioinformatics, Semantic Web Browser, Knowledge Managements, Software Engineering, etc. One of the pillars of the Semantic Web is ontologies. They are an important aspect in knowledge representation and reasoning for the Semantic Web.

Nowadays, ontologies are usually modeled by using the Web Ontology Language (OWL), which is a standard recommended by W3C for the Semantic Web. In essence, OWL is a language based on description logics (DLs) [14], which are a family of formal languages suitable for representing and reasoning about terminological knowledge [3]. Constructing useful ontologies is desirable, and in ontology engineering, concept learning is helpful for suggesting important concepts and their definitions.

Concept learning in DLs is similar to binary classification in traditional machine learning. However, the problem in the context of DLs differs from the traditional setting in that objects are described not only by attributes but also by binary relations between objects.

The major settings of concept learning in DLs are as follows:

Setting 1 Given a knowledge base $\mathcal{KB}$ in a DL L and sets E^+, E^- of individuals, learn a concept C in L such that:

1. $\mathcal{KB} \models C(a)$ for all $a \in E^+$, and
2. $\mathcal{KB} \models \neg C(a)$ for all $a \in E^-$.

The sets E^+ and E^- contain positive examples and negative ones of C, respectively.

Setting 2 This setting differs from the previous one only in that the condition (2) is replaced by the weaker one: $\mathcal{KB} \not\models C(a)$ for all $a \in E^-$.

Setting 3 Given an interpretation $\mathcal{I}$ and sets E^+, E^- of individuals, learn a concept C in a DL L such that:

1. $\mathcal{I} \models C(a)$ for all $a \in E^+$, and
2. $\mathcal{I} \models \neg C(a)$ for all $a \in E^-$.

Note that $\mathcal{I} \not\models C(a)$ is the same as $\mathcal{I} \models \neg C(a)$.

In DLs, the domain of interest is described in terms of individuals (objects), concepts, object roles and data roles. A concept stands for a set of objects, an object role stands for a binary relation between objects, and a data role stands for a binary predicate relating objects to data values. Thus, a concept name is a Boolean attribute, and a "functional" data role name is a partial attribute. The interesting features are object roles, which specify the relationships between objects, as well as concept constructors and object role constructors, which allow us to form complex concepts and complex roles from concept names, role names and individual names.

Nguyen and Szałas [22] proposed to use bisimulation for concept learning in DLs. They also studied concept approximation using bisimulation and Pawlak's rough set theory [23,24]. They defined *terminological roughification* to be any technique that uses the largest auto-bisimulation relations as the equivalence relation for defining approximations. The learning algorithm proposed in [22] works for information systems that are either explicitly given as an interpretation or specified by an acyclic knowledge base with closed world assumption. This is similar the traditional setting of machine learning, and a bit different from those of [4,11,15,18,19], where learning algorithms work for (cyclic) knowledge base with open world assumption.

This work extends the concept learning method of [22] for richer DLs and provides experimental results. It combines and revises the results reported in our conference papers [30,31] to give a full picture on the current state of bisimulation-based concept learning for information systems in DLs.[1] In comparison with [22], we take attributes as basic elements of the language. Each attribute may be discrete or numeric. A Boolean attribute is treated as a concept name. One can simulate a discrete attribute by concepts as shown in [22], but this is inconvenient. A more serious problem concerns numeric attributes. To deal with this, one can directly allow numeric attributes as in the current work or use data roles of description logic together with comparison operators. Direct use of numeric attributes makes the concept learning problem closer to binary classification in traditional machine learning. Neither attributes nor data roles are considered in [22]. Thus, our approach is more general and suitable for practical information systems than the one of [22]. As further extensions, we also allow data roles and the concept constructors "functionality" and "unqualified number restrictions". Also note that the work [22] does not provide details on strategies for domain partitioning nor experimental results.

The class of DLs studied in this paper is very rich. It allows all role constructors of $\mathcal{ALC}_{reg}$ (a variant of propositional dynamic logic) plus I (inverse) and U (universal role), and all concept constructors of $\mathcal{ALC}_{reg}$ plus O (nominal), F (functionality), N (unqualified number restriction), Q (qualified number restriction) and Self (local reflexivity of a role).

We formulate and prove an important theorem on basic selectors, which states that: to reach the partition corresponding to the equivalence relation characterizing indiscernibility of objects w.r.t. a given sublanguage of the considered logic, it suffices to start from the full block consisting of all objects and repeatedly granulate it using the basic selectors of the sublanguage.

Regarding the granulation process, a natural question is: which block from the current partition should be split first and which selector should be used to split it? These affect both the "quality" of the final partition and the complexity of the process. One can use basic selectors and simple selectors together with information gain to guide the granulation process, where simple selectors cover basic selectors and are created from the blocks of the partitions appeared in the granulation process. Such selectors split blocks in the partitions and bring good results in favorable cases. However, they are not advanced enough for complex cases. To obtain a final partition, the main loop of the granulation process may need to repeat many times. This usually results in too complex concepts, which poorly classify new objects.

As an extension of [22,30], apart from simple selectors, we propose to use also a new kind of selectors, called *extended selectors*, which are created from available selectors (the current simple selectors and extended selectors) for splitting blocks.

We have implemented the bisimulation-based method using the mentioned selectors and information gain measures for concept learning in DLs w.r.t. Setting 3. The aim is to study effectiveness of simple selectors and extended selectors as well as to provide experimental results for the concept learning method. The evaluation results show that the concept learning method is valuable and extended selectors support it significantly.

The rest of this paper is organized as follows. In Sect. 2 we discuss related work. In Sect. 3 we give a brief introduction to DLs and DL-based information systems. Section 4 contains the theory of bisimulation and indiscernibility. In Sect. 5 we describe the concept learning method for DL-based information systems, formulate and prove a theorem on

[1] The paper [30] is co-authored by our colleagues.

basic selectors. In addition, we introduce extended selectors, the simplicity of concepts and information gain measure for granulating partitions. Techniques for processing the resulting concepts are also presented in this section. In Sect. 6 we provide examples to illustrate the bisimulation-based concept learning algorithm. The experimental results are presented in Sect. 7. We summarise our work and draw conclusions in Sect. 8.

2 Related work

Related work on methods of concept learning in DLs can be divided into three groups. The first group focuses on learnability in DLs [6,12] and presents some relatively simple algorithms [6,12,18]. The second group studies concept learning in DLs using refinement operators as in inductive logic programming [4,11,15,19]. The third group exploits bisimulation for concept learning in DLs [13,22,29,30].

As an early work on concept learning in DLs, Cohen and Hirsh [6] studied PAC-learnability of the CLASSIC description logic (an early DL formalism) and its sublogic called C-CLASSIC. They proposed a concept learning algorithm called LCSLearn, which is based on "least common subsumers". In [12] Franzier and Pitt provided some results on learnability in the CLASSIC description logic using some kinds of queries and either the exact learning model or the PAC model. In [18] Lambrix and Larocchia proposed a simple concept learning algorithm based on concept normalization.

Badea and Nienhuys-Cheng [4] studied concept learning in the DL $\mathcal{ALER}$ for Setting 1 using refinement operators as in inductive logic programming. They introduced some theoretical properties of refinement operators and used a downward refinement operator to enable a top-down search. Iannone et al. [15] also investigated learning algorithms using refinement operators for Setting 1, but for the richer DL $\mathcal{ALC}$. The main idea of those algorithms is to find and remove some parts of a concept responsible for classification errors.

The works [11,19] study concept learning in DLs for Setting 2 using refinement operators. Fanizzi et al. [11] introduced the DL-FOIL system that is adapted to concept learning for DL representations supporting the OWL–DL language. They also considered unlabeled data as in semi-supervised learning. Lehmann and Hitzler [19] introduced methods from inductive logic programing for concept learning in DL knowledge bases. Their algorithm, DL-Learner, exploits genetic programming techniques.

Apart from refinement operators, scoring functions and search strategies also play an important role in the algorithms proposed in the works [4,11,15,19].

As DLs are variants of modal logics, bisimulation can be used to characterize indiscernibility of objects [9,10] and hence for concept learning in DLs. The earlier discussed work by Nguyen and Szałas [22] is devoted to Setting 3. Ha et al. [13] developed the first bisimulation-based methods, called BBCL and dual-BBCL, for concept learning in DLs using Setting 1. The idea is to use models of the given knowledge base and bisimulation in those models to guide the search for the concept to be learnt. Based on the BBCL method, Divroodi et al. [8] studied C-learnability in DLs for Setting 3. Tran et al. [29] developed the BBCL2 method for concept learning in DLs using Setting 2. It is based on dual-BBCL. The methods of [13,22,29] are formulated for a large class of useful DLs, with well-known DLs such as $\mathcal{ALC}$, $\mathcal{SHIQ}$, $\mathcal{SHOIQ}$, $\mathcal{SROIQ}$. There are also other related works on learning terminological axioms or ontologies. The works [2,7,20] study theory learning or concept inclusion axioms learning in DLs. The works [2,20] involve probabilistic DLs. Some other researchers combined concept learning in DLs with inductive logic programming (e.g., [16]) or studied concept learning in DLs via inductive logic programming (e.g., [17]).

3 Description logic-based information systems

3.1 Description logics and semantics

The basic DL $\mathcal{ALC}$ was first introduced by Schmidt–Schaubß and Smolka in [27]. The name $\mathcal{ALC}$ stands for "Attribute concept Language with Complements". It is obtained from the DL $\mathcal{AL}$ by adding the complement constructor ($\neg$). Complex concepts of $\mathcal{ALC}$ are built from simpler ones using various constructors. In the following, we recall a formal definition of the syntax of $\mathcal{ALC}$ [19].

Definition 1 ($\mathcal{ALC}$ Syntax) Let Σ_C be a set of *concept names* and Σ_R be a set of *role names* ($\Sigma_C \cap \Sigma_R = \emptyset$). The elements in Σ_C are called *atomic concepts*. The description logic $\mathcal{ALC}$ allows concepts defined recursively as follows:

- if $A \in \Sigma_C$ then A is a concept,
- if C and D are concepts and $r \in \Sigma_R$ is a role then $\top$, $\bot$, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists r.C$ and $\forall r.C$ are also concepts.

The intended meaning of the symbols and concept constructors is described as follows:

- $\top$ stands for the top concept,
- $\bot$ stands for the bottom concept,
- $\neg C$ stands for the complement of C,
- $C \sqcap D$ stands for the intersection of C and D,
- $C \sqcup D$ stands for the union of C and D,
- $\exists r.C$ stands for the existential restriction of C by r,
- $\forall r.C$ stands for the value/universal restriction of C by r.

The following grammar rule is a brief description of syntax of concepts in $\mathcal{ALC}$:

$$C, D \longrightarrow A \mid \top \mid \bot \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists r.C \mid \forall r.C$$

Propositional dynamic logic (PDL) is a modal logic specifically designed for reasoning about program schemes. Schild [26] pointed out an interesting correspondence between DLs and PDL. In the following, we give the definition of the description logic $\mathcal{ALC}_{reg}$ that corresponds to PDL.

Definition 2 ($\mathcal{ALC}_{reg}$ Syntax) Let Σ_C be a set of *concept names* and Σ_R be a set of *role names* ($\Sigma_C \cap \Sigma_R = \emptyset$). The elements in Σ_C are called *atomic concepts*, while the elements in Σ_R are called *atomic roles*. The DL $\mathcal{ALC}_{reg}$ allows concepts and roles defined recursively as follows:

- if $A \in \Sigma_C$ then A is a concept,
- if $r \in \Sigma_R$ then r is a role,
- if C and D are concepts, R and S are roles then
  - ε, $R \circ S$, $R \sqcup S$, R^*, $C?$ are roles,
  - $\top, \bot, \neg C, C \sqcap D, C \sqcup D, \exists R.C$ and $\forall R.C$ are concepts.

The following rules are a brief description of syntax of concepts and roles in $\mathcal{ALC}_{reg}$, where $r \in \Sigma_R$ and $A \in \Sigma_C$:

$$R, S \longrightarrow \varepsilon \mid r \mid R \circ S \mid R \sqcup S \mid R^* \mid C?$$
$$C, D \longrightarrow A \mid \top \mid \bot \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C$$

The meaning of the role constructors is as follows:

- ε stands for the identity relation,
- $R \circ S$ stands for the sequential composition of R and S,
- $R \sqcup S$ stands for the set-theoretical union of R and S,
- R^* stands for the reflexive and transitive closure of R,
- $C?$ stands for the test operator.

The concept constructors $\forall R.C$ and $\exists R.C$ correspond to the modal operators $[R]C$ and $\langle R \rangle\ C$ of PDL, respectively. We now consider description languages with attributes and additional concept/role constructors.

A *DL-signature* is a finite set $\Sigma = \Sigma_I \cup \Sigma_{dA} \cup \Sigma_{nA} \cup \Sigma_{oR} \cup \Sigma_{dR}$, where Σ_I is a set of *individuals*, Σ_{dA} is a set of *discrete attributes*, Σ_{nA} is a set of *numeric attributes*, Σ_{oR} is a set of *object role names*, and Σ_{dR} is a set of *data roles*.[2] All the sets Σ_I, Σ_{dA}, Σ_{nA}, Σ_{oR}, Σ_{dR} are pairwise disjoint.

Let $\Sigma_A = \Sigma_{dA} \cup \Sigma_{nA}$. For each attribute $A \in \Sigma_A$, $range(A)$ is a non-empty set that is countable if A is discrete, and partially ordered by $\leq$ otherwise.[3] (For simplicity we do not subscript $\leq$by A.) A discrete attribute is called a *Boolean attribute* if $range(A) = \{\mathsf{true}, \mathsf{false}\}$. We refer to Boolean attributes also as *concept names*. Let $\Sigma_C \subseteq \Sigma_{dA}$ be the set of all concept names of Σ.

An object role name stands for a binary predicate between individuals. A data role σ stands for a binary predicate relating individuals to elements of a set $range(\sigma)$.

We will denote individuals by letters such as a and b, attributes by letters such as A and B, object role names by letters such as r and s, data roles by letters such as σ and ϱ, and elements of sets of the form $range(A)$ or $range(\sigma)$ by letters such as c and d.

We consider some *DL-features* denoted by I (*inverse*), O (*nominal*), F (*functionality*), N (*unqualified number restriction*), Q (*qualified number restriction*), U (*universal role*), Self (*local reflexivity of a role*). A *set of DL-features* is a set consisting of zero or some of these names.

Definition 3 (The $\mathcal{L}_{\Sigma,\Phi}$ Language) Let Σ be a DL-signature, Φ be a set of DL-features and $\mathcal{L}$ stand for $\mathcal{ALC}_{reg}$. The DL language $\mathcal{L}_{\Sigma,\Phi}$ allows *object roles* and *concepts* defined recursively as follows:

- if $r \in \Sigma_{oR}$ then r is an object role of $\mathcal{L}_{\Sigma,\Phi}$,
- if $A \in \Sigma_C$ then A is concept of $\mathcal{L}_{\Sigma,\Phi}$,
- if $A \in \Sigma_A \setminus \Sigma_C$ and $d \in range(A)$ then $A = d$ and $A \neq d$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
- if $A \in \Sigma_{nA}$ and $d \in range(A)$ then $A \leq d$, $A < d$, $A \geq d$ and $A > d$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
- if R and S are object roles of $\mathcal{L}_{\Sigma,\Phi}$, C and D are concepts of $\mathcal{L}_{\Sigma,\Phi}$, $r \in \Sigma_{oR}$, $\sigma \in \Sigma_{dR}$, $a \in \Sigma_I$, and n is a natural number then
  - ε, $R \circ S$, $R \sqcup S$, R^* and $C?$ are object roles of $\mathcal{L}_{\Sigma,\Phi}$,
  - $\top, \bot, \neg C, C \sqcap D, C \sqcup D, \forall R.C$ and $\exists R.C$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $d \in range(\sigma)$ then $\exists \sigma.\{d\}$ is a concept of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $I \in \Phi$ then R^- is an object role of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $O \in \Phi$ then $\{a\}$ is a concept of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $F \in \Phi$ then $\leq 1\, r$ is a concept of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $\{F, I\} \subseteq \Phi$ then $\leq 1\, r^-$ is a concept of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $N \in \Phi$ then $\geq n\, r$ and $\leq n\, r$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $\{N, I\} \subseteq \Phi$ then $\geq n\, r^-$ and $\leq n\, r^-$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $Q \in \Phi$ then $\geq n\, r.C$ and $\leq n\, r.C$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $\{Q, I\} \subseteq \Phi$ then $\geq n\, r^-.C$ and $\leq n\, r^-.C$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,

[2] Object role names are atomic object roles.

[3] One can assume that, if A is a numeric attribute, then $range(A)$ is the set of real numbers and $\leq$ is the usual linear order between real numbers.

$$(R \circ S)^{\mathcal{I}} = R^{\mathcal{I}} \circ S^{\mathcal{I}} \qquad (R^-)^{\mathcal{I}} = (R^{\mathcal{I}})^{-1} \qquad (R^*)^{\mathcal{I}} = (R^{\mathcal{I}})^*$$

$$(R \sqcup S)^{\mathcal{I}} = R^{\mathcal{I}} \cup S^{\mathcal{I}} \qquad (C?)^{\mathcal{I}} = \{\langle x, x\rangle \mid C^{\mathcal{I}}(x)\}$$

$$\varepsilon^{\mathcal{I}} = \{\langle x, x\rangle \mid x \in \Delta^{\mathcal{I}}\} \qquad U^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \qquad \top^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \qquad \bot^{\mathcal{I}} = \emptyset$$

$$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \qquad \{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$$

$$(A \leq d)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid A^{\mathcal{I}}(x) \text{ is defined}, A^{\mathcal{I}}(x) \leq d\}$$

$$(A \geq d)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid A^{\mathcal{I}}(x) \text{ is defined}, A^{\mathcal{I}}(x) \geq d\}$$

$$(A = d)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid A^{\mathcal{I}}(x) = d\}$$

$$(A \neq d)^{\mathcal{I}} = (\neg(A = d))^{\mathcal{I}}$$

$$(A < d)^{\mathcal{I}} = ((A \leq d) \sqcap (A \neq d))^{\mathcal{I}}$$

$$(A > d)^{\mathcal{I}} = ((A \geq d) \sqcap (A \neq d))^{\mathcal{I}}$$

$$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\,[R^{\mathcal{I}}(x, y) \Rightarrow C^{\mathcal{I}}(y)]\}$$

$$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\,[R^{\mathcal{I}}(x, y) \wedge C^{\mathcal{I}}(y)]\}$$

$$(\exists r.\mathsf{Self})^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x, x)\}$$

$$(\exists \sigma.\{d\})^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sigma^{\mathcal{I}}(x, d)\}$$

$$(\geq n\, R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \mid R^{\mathcal{I}}(x, y) \wedge C^{\mathcal{I}}(y)\} \geq n\}$$

$$(\leq n\, R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \mid R^{\mathcal{I}}(x, y) \wedge C^{\mathcal{I}}(y)\} \leq n\}$$

$$(\geq n\, R)^{\mathcal{I}} = (\geq n\, R.\top)^{\mathcal{I}} \qquad (\leq n\, R)^{\mathcal{I}} = (\leq n\, R.\top)^{\mathcal{I}}$$

Fig. 1 Interpretation of complex object roles and complex concepts

- if $U \in \Phi$ then U is an object role of $\mathcal{L}_{\Sigma,\Phi}$,
- if $\mathsf{Self} \in \Phi$ then $\exists r.\mathsf{Self}$ is a concept of $\mathcal{L}_{\Sigma,\Phi}$.

The concept constructors $\geq n\, R.C$ and $\leq n\, R.C$ are called qualified number restrictions. They correspond to graded modal operators. The concept constructors $\geq n\, R$ and $\leq n\, R$ are called unqualified number restrictions.

Definition 4 (Semantics) An *interpretation* in $\mathcal{L}_{\Sigma,\Phi}$ is a pair $\mathcal{I} = \left(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\right)$, where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain* of $\mathcal{I}$ and $\cdot^{\mathcal{I}}$ is a mapping called the *interpretation function* of $\mathcal{I}$ that associates each individual $a \in \Sigma_I$ with an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each concept name $A \in \Sigma_C$ with a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, each attribute $A \in \Sigma_A \setminus \Sigma_C$ with a partial function $A^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \mathit{range}(A)$, each object role name $r \in \Sigma_{oR}$ with a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and each data role $\sigma \in \Sigma_{dR}$ with a binary relation $\sigma^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \mathit{range}(\sigma)$. The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex object roles and complex concepts as shown in Fig. 1, where $\#\Gamma$ stands for the cardinality of the set Γ.

As showed in Definition 4, each individual is interpreted as an object, each concept name is interpreted as a set of objects, each attribute is interpreted as a partial function from the domain to the set of values of the attribute, each object role name is interpreted as a binary relation between objects and each data role is interpreted as a binary relation between objects and elements in the range of the data role.

We say that $C^{\mathcal{I}}$ (resp. $R^{\mathcal{I}}$) is the *denotation* of the concept C (resp. object role R) in the interpretation $\mathcal{I}$. If $a^{\mathcal{I}} \in C^{\mathcal{I}}$, then we say that a is an *instance* of C in the interpretation $\mathcal{I}$. For abbreviation, we write $C^{\mathcal{I}}(x)$ (resp. $R^{\mathcal{I}}(x, y)$ and $\sigma^{\mathcal{I}}(x, d)$) instead of $x \in C^{\mathcal{I}}$ (resp. $\langle x, y\rangle \in R^{\mathcal{I}}$ and $\langle x, d\rangle \in \sigma^{\mathcal{I}}$).

3.2 Description logic-based information systems

In DL systems, information is stored in a knowledge base. It is usually made up of two parts. The first part contains individual assertions, called the *ABox*, while the second part contains terminological axioms, called the *TBox*.

An *individual assertion* in $\mathcal{L}_{\Sigma,\Phi}$ is of the form $A(a)$, $(B = c)(a)$, $r(a, b)$ or $\sigma(a, d)$, where $a, b \in \Sigma_I$, $A \in \Sigma_C$, $B \in \Sigma_A \setminus \Sigma_C$, $c \in \mathit{range}(B)$, $r \in \Sigma_{oR}$, $\sigma \in \Sigma_{dR}$ and $d \in \mathit{range}(\sigma)$. For simplicity of notation, we write $B(a) = c$ instead of $(B = c)(a)$.

In the most general case, a *terminological axiom* in $\mathcal{L}_{\Sigma,\Phi}$ is of the form $C \sqsubseteq D$ (resp. $R \sqsubseteq S$), $C \equiv D$ (resp. $R \equiv S$), where C and D are concepts of $\mathcal{L}_{\Sigma,\Phi}$, R and S are object roles of $\mathcal{L}_{\Sigma,\Phi}$. The former is called a *general inclusion axiom*, while the latter is called an *equivalence axiom*. For an equivalence axiom $C \equiv D$ (resp. $R \equiv S$), if C (resp. R) is a concept (resp. an object role) name of the form $A \in \Sigma_C$ (resp. $r \in \Sigma_{oR}$) then $A \equiv D$ (resp. $r \equiv S$) is called a concept (resp. an object role) definition.

Definition 5 (Knowledge Base) An *acyclic knowledge base* in $\mathcal{L}_{\Sigma,\Phi}$ is a pair $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A}\rangle$, where:

- $\mathcal{A}$ is a finite set, called the *ABox* (assertion box) of $\mathcal{KB}$, consisting of individual assertions.
- $\mathcal{T}$ is a finite list $(\varphi_1, \varphi_2, \ldots, \varphi_n)$, called the *TBox* (terminological box) of $\mathcal{KB}$, where each φ_i is a terminological axiom of one of the the following forms:
  - $A \equiv C$, where $A \in \Sigma_C$ is a concept name not occurring in C, $\mathcal{A}$ and $\varphi_1, \varphi_2, \ldots, \varphi_{i-1}$,
  - $r \equiv R$, where R is an object role of $\mathcal{L}_{\Sigma,\Phi}$ and $r \in \Sigma_{oR}$ is an object role name not occurring in R, $\mathcal{A}$ and $\varphi_1, \varphi_2, \ldots, \varphi_{i-1}$.

An acyclic knowledge base as defined above is similar to stratified logic programs [1]. The concept (resp. object role) names occurring in $\mathcal{A}$ are said to be *primitive* concepts (resp. object roles), while the concept (resp. object role) names occurring in the left hand-side of '$\equiv$' in the definitions from $\mathcal{T}$ are called *defined* concepts (resp. object roles).

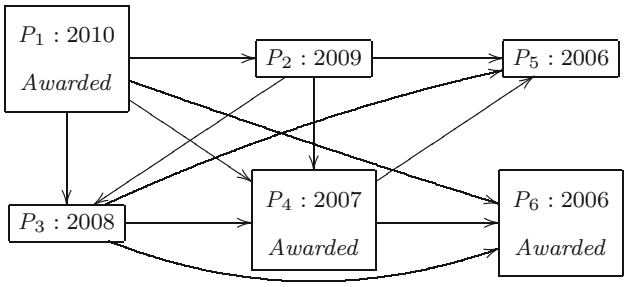

Fig. 2 An illustration for the knowledge base given in Example 1

Example 1 This example is about publications. Let

$\Phi = \{I, O, N, Q\}$, $\Sigma_I = \{P_1, P_2, P_3, P_4, P_5, P_6\}$,
$\Sigma_C = \{Pub, Book, Article, Awarded, ExcellentPub,$
$UsefulPub, GoodPub, RecentPub, CitingP5\}$,
$\Sigma_{dA} = \Sigma_C \cup \{Title, Kind\}$, $\Sigma_{nA} = \{Year\}$,
$\Sigma_{oR} = \{cites, cited\_by\}$, $\Sigma_{dR} = \emptyset$,
$\mathcal{A} = \{Title(P_1) =$ "Introduction to Logic",
$Title(P_2) =$ "The Essence of Logic",
$Awarded(P_1), Awarded(P_4), Awarded(P_6)$,
$Kind(P_1) =$ "book", $Kind(P_2) =$ "book",
$Kind(P_3) =$ "book", $Kind(P_4) =$ "article",
$Kind(P_5) =$ "article", $Kind(P_6) =$ "conf. paper",
$Year(P_1) = 2010$, $Year(P_2) = 2009$,
$Year(P_3) = 2008$, $Year(P_4) = 2007$,
$Year(P_5) = 2006$, $Year(P_6) = 2006$,
$cites(P_1, P_2), cites(P_1, P_3), cites(P_1, P_4)$,
$cites(P_1, P_6), cites(P_2, P_3), cites(P_2, P_4)$,
$cites(P_2, P_5), cites(P_3, P_4), cites(P_3, P_5)$,
$cites(P_3, P_6), cites(P_4, P_5), cites(P_4, P_6)\}$,
$\mathcal{T} = (P \equiv \top, Book \equiv (Kind =$ "book"),
$Article \equiv (Kind =$ "article"),
$cited\_by \equiv cites^{-}$, $UsefulPub \equiv \exists cited\_by.\top$,
$GoodPub \equiv (\geq 2\, cited\_by)$,
$ExcellentPub \equiv GoodPub \sqcap Awarded$,
$RecentPub \equiv (Year \geq 2008)$,
$CitingP5 \equiv \exists cites.\{P_5\})$.

Then $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$ is an acyclic knowledge base in $\mathcal{L}_{\Sigma,\Phi}$. The definition $Pub \equiv \top$ states that the domain of any model of $\mathcal{KB}$ consists of only publications. This knowledge base is illustrated in Fig. 2. In this figure, nodes denote publications and edges denote citations (i.e., assertions of the role *cites*), and we display only information concerning assertions about *Year*, *Awarded* and *cites*.

The relation whether an interpretation $\mathcal{I}$ *satisfies* an individual assertion or a terminological axiom φ, denoted by $\mathcal{I} \models \varphi$, is defined as follows:

$\mathcal{I} \models A(a)$ if $A^{\mathcal{I}}(a^{\mathcal{I}}) = \mathsf{true}$,
$\mathcal{I} \models (B(a) = c)$ if $B^{\mathcal{I}}(a^{\mathcal{I}}) = c$,
$\mathcal{I} \models r(a, b)$ if $r^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}})$,
$\mathcal{I} \models \sigma(a, d)$ if $\sigma^{\mathcal{I}}(a^{\mathcal{I}}, d)$,
$\mathcal{I} \models C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$,
$\mathcal{I} \models C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$,
$\mathcal{I} \models R \equiv S$ if $R^{\mathcal{I}} = S^{\mathcal{I}}$,
$\mathcal{I} \models R \sqsubseteq S$ if $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$.

Definition 6 (Model) An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$, denoted by $\mathcal{I} \models \mathcal{T}$, if it satisfies all the terminological axioms of $\mathcal{T}$. It is a *model* of an ABox $\mathcal{A}$, denoted by $\mathcal{I} \models \mathcal{A}$, if it satisfies all the individual assertions of $\mathcal{A}$. It is a *model* of a knowledge base $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$, denoted by $\mathcal{I} \models \mathcal{KB}$, if it is a model of both $\mathcal{T}$ and $\mathcal{A}$.

The *standard model* of an acyclic knowledge base $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$ in $\mathcal{L}_{\Sigma,\Phi}$ is an interpretation $\mathcal{I}$ such that:

- $\Delta^{\mathcal{I}} = \Sigma_I$ (i.e., the domain of $\mathcal{I}$ consists of all the individual names of Σ),
- if $A \in \Sigma_C$ is a primitive concept of $\mathcal{KB}$ then $A^{\mathcal{I}} = \{a \mid A(a) \in \mathcal{A}\}$,
- if $B \in \Sigma_A \backslash \Sigma_C$ then $B^{\mathcal{I}} : \Delta^{\mathcal{I}} \to range(B)$ is the partial function such that $B^{\mathcal{I}}(a) = c$ if $(B(a) = c) \in \mathcal{A}$,
- if $r \in \Sigma_{oR}$ is a primitive object role of $\mathcal{KB}$ then $r^{\mathcal{I}} = \{\langle a, b \rangle \mid r(a, b) \in \mathcal{A}\}$,
- if $\sigma \in \Sigma_{dR}$ then $\sigma^{\mathcal{I}} = \{\langle a, d \rangle \mid \sigma(a, d) \in \mathcal{A}\}$,
- if $A \equiv C$ is a definition from $\mathcal{T}$ then $A^{\mathcal{I}} = C^{\mathcal{I}}$,
- if $r \equiv R$ is a definition from $\mathcal{T}$ then $r^{\mathcal{I}} = R^{\mathcal{I}}$,
- if $A \in \Sigma_C$ but A does not occur in $\mathcal{KB}$ then $A^{\mathcal{I}} = \emptyset$,
- if $r \in \Sigma_{oR}$ but r does not occur in $\mathcal{KB}$ then $r^{\mathcal{I}} = \emptyset$.

Note that the standard model of $\mathcal{KB}$ is a finite interpretation. The definition adopts the *unique name* and *closed world assumptions*.

Remark 1 The unique name assumption is used just for increasing simplicity. One can allow ABoxes to contain individual assertions of the form $a = b$, where $a, b \in \Sigma_I$, with the semantics that an interpretation $\mathcal{I}$ satisfies $a = b$ if $a^{\mathcal{I}} = b^{\mathcal{I}}$. In that case a given acyclic knowledge base in $\mathcal{L}_{\Sigma,\Phi}$ can be converted to an equivalent one by merging individuals that are equal to each other.

Definition 7 (Information System) An *information system specified by an acyclic knowledge base* in $\mathcal{L}_{\Sigma,\Phi}$ is the standard model of that knowledge base in $\mathcal{L}_{\Sigma,\Phi}$.

Example 2 Consider the knowledge base $\mathcal{KB}$ given in Example 1. The information system specified by $\mathcal{KB}$ is the interpretation $\mathcal{I}$ with:

$$\Delta^{\mathcal{I}} = \{P_1, P_2, P_3, P_4, P_5, P_6\}, Pub^{\mathcal{I}} = \Delta^{\mathcal{I}},$$
$$P_1^{\mathcal{I}} = P_1, P_2^{\mathcal{I}} = P_2, \ldots, P_6^{\mathcal{I}} = P_6, \ Book^{\mathcal{I}} = \{P_1, P_2, P_3\},$$
$$Article^{\mathcal{I}} = \{P_4, P_5\}, \ Awarded^{\mathcal{I}} = \{P_1, P_4, P_6\},$$
$$cites^{\mathcal{I}} = \{\langle P_1, P_2\rangle, \langle P_1, P_3\rangle, \langle P_1, P_4\rangle, \langle P_1, P_6\rangle, \langle P_2, P_3\rangle, \langle P_2, P_4\rangle, \langle P_2, P_5\rangle, \langle P_3, P_4\rangle, \langle P_3, P_5\rangle, \langle P_3, P_6\rangle, \langle P_4, P_5\rangle, \langle P_4, P_6\rangle\},$$
$$cited\_by^{\mathcal{I}} = (cites^{\mathcal{I}})^{-1}, UsefulPub^{\mathcal{I}} = \{P_2, P_3, P_4, P_5, P_6\},$$
$$GoodPub^{\mathcal{I}} = \{P_3, P_4, P_5, P_6\}, ExcellentPub^{\mathcal{I}} = \{P_4, P_6\},$$
$$RecentPub^{\mathcal{I}} = \{P_1, P_2, P_3\}, CitingP5^{\mathcal{I}} = \{P_2, P_3, P_4\},$$

the (partial) functions $Title^{\mathcal{I}}$, $Year^{\mathcal{I}}$ and $Kind^{\mathcal{I}}$ are specified as usual.

4 Bisimulation and indiscernibility

Bisimulations were first introduced under the name *p-relation* [32] and the name *zigzag relation* [33] by J. van Benthem. They have been used to analyze the expressiveness of a wide range of extended modal logics. Divroodi and Nguyen [9,10] studied bisimulations for a number of DLs. Nguyen and Szałas [22] generalized that notion to model indiscernibility of objects and study concept learning. In this section, we generalize the notion of bisimulation further for the class of DLs studied in this paper, which is larger and more general than the one studied in [9,10,22].

Definition 8 (Bisimulation) Let Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, Φ and $\Phi^\dagger$ be sets of DL-features such that $\Phi^\dagger \subseteq \Phi$, $\mathcal{I}$ and $\mathcal{I}'$ be interpretations in $\mathcal{L}_{\Sigma,\Phi}$.

A binary relation $Z \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}'}$ is called an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*bisimulation* between $\mathcal{I}$ and $\mathcal{I}'$ if the following conditions hold for every $a \in \Sigma_I^\dagger$, $A \in \Sigma_C^\dagger$, $B \in \Sigma_A^\dagger \setminus \Sigma_C^\dagger$, $r \in \Sigma_{oR}^\dagger$, $\sigma \in \Sigma_{dR}^\dagger$, $d \in range(\sigma)$, $x, y \in \Delta^{\mathcal{I}}$, $x', y' \in \Delta^{\mathcal{I}'}$:

$$Z(a^{\mathcal{I}}, a^{\mathcal{I}'}) \tag{1}$$
$$Z(x, x') \Rightarrow [A^{\mathcal{I}}(x) \Leftrightarrow A^{\mathcal{I}'}(x')] \tag{2}$$
$$Z(x, x') \Rightarrow [B^{\mathcal{I}}(x) = B^{\mathcal{I}'}(x') \text{ or both are undefined}] \tag{3}$$
$$[Z(x, x') \wedge r^{\mathcal{I}}(x, y)] \Rightarrow \exists y' \in \Delta^{\mathcal{I}'}[Z(y, y') \wedge r^{\mathcal{I}'}(x', y')] \tag{4}$$
$$[Z(x, x') \wedge r^{\mathcal{I}'}(x', y')] \Rightarrow \exists y \in \Delta^{\mathcal{I}}[Z(y, y') \wedge r^{\mathcal{I}}(x, y)] \tag{5}$$
$$Z(x, x') \Rightarrow [\sigma^{\mathcal{I}}(x, d) \Leftrightarrow \sigma^{\mathcal{I}'}(x', d)], \tag{6}$$

if $I \in \Phi^\dagger$ then

$$[Z(x, x') \wedge r^{\mathcal{I}}(y, x)] \Rightarrow \exists y' \in \Delta^{\mathcal{I}'}[Z(y, y') \wedge r^{\mathcal{I}'}(y', x')] \tag{7}$$
$$[Z(x, x') \wedge r^{\mathcal{I}'}(y', x')] \Rightarrow \exists y \in \Delta^{\mathcal{I}}[Z(y, y') \wedge r^{\mathcal{I}}(y, x)], \tag{8}$$

if $O \in \Phi^\dagger$ then

$$Z(x, x') \Rightarrow [x = a^{\mathcal{I}} \Leftrightarrow x' = a^{\mathcal{I}'}], \tag{9}$$

if $N \in \Phi^\dagger$ then

$$Z(x, x') \Rightarrow \#\{y \mid r^{\mathcal{I}}(x, y)\} = \#\{y' \mid r^{\mathcal{I}'}(x', y')\}, \tag{10}$$

if $\{N, I\} \subseteq \Phi^\dagger$ then

$$Z(x, x') \Rightarrow \#\{y \mid r^{\mathcal{I}}(y, x)\} = \#\{y' \mid r^{\mathcal{I}'}(y', x')\}, \tag{11}$$

if $F \in \Phi^\dagger$ then

$$Z(x, x') \Rightarrow [\#\{y \mid r^{\mathcal{I}}(x, y)\} \leq 1 \Leftrightarrow \#\{y' \mid r^{\mathcal{I}'}(x', y')\} \leq 1], \tag{12}$$

if $\{F, I\} \subseteq \Phi^\dagger$ then

$$Z(x, x') \Rightarrow [\#\{y \mid r^{\mathcal{I}}(y, x)\} \leq 1 \Leftrightarrow \#\{y' \mid r^{\mathcal{I}'}(y', x')\} \leq 1], \tag{13}$$

if $Q \in \Phi^\dagger$ then

$$\text{if } Z(x, x') \text{ holds then, for every } r \in \Sigma_{oR}^\dagger, \text{ there exists a bijection } h : \{y \mid r^{\mathcal{I}}(x, y)\} \to \{y' \mid r^{\mathcal{I}'}(x', y')\} \text{ such that } h \subseteq Z, \tag{14}$$

if $\{Q, I\} \subseteq \Phi^\dagger$ then

$$\text{if } Z(x, x') \text{ holds then, for every } r \in \Sigma_{oR}^\dagger, \text{ there exists a bijection } h : \{y \mid r^{\mathcal{I}}(y, x)\} \to \{y' \mid r^{\mathcal{I}'}(y', x')\} \text{ such that } h \subseteq Z, \tag{15}$$

if $U \in \Phi^\dagger$ then

$$\forall x \in \Delta^{\mathcal{I}} \exists x' \in \Delta^{\mathcal{I}'} Z(x, x') \tag{16}$$
$$\forall x' \in \Delta^{\mathcal{I}'} \exists x \in \Delta^{\mathcal{I}} Z(x, x'), \tag{17}$$

if $\mathsf{Self} \in \Phi^\dagger$ then

$$Z(x, x') \Rightarrow [r^{\mathcal{I}}(x, x) \Leftrightarrow r^{\mathcal{I}'}(x', x')]. \tag{18}$$

We say that $\mathcal{I}$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*bisimilar* to $\mathcal{I}'$ if there exists an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation between $\mathcal{I}$ and $\mathcal{I}'$. We say that $x \in \Delta^{\mathcal{I}}$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*bisimilar* to $x' \in \Delta^{\mathcal{I}'}$ if there exists an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation between $\mathcal{I}$ and $\mathcal{I}'$ such that $Z(x, x')$ holds.

A concept C of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ is said to be *invariant for* $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*bisimulation* if, for every interpretations $\mathcal{I}$ and $\mathcal{I}'$ in $\mathcal{L}_{\Sigma,\Phi}$ with $\Sigma \supseteq \Sigma^\dagger$ and $\Phi \supseteq \Phi^\dagger$, and every $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation Z between $\mathcal{I}$ and $\mathcal{I}'$, if $Z(x, x')$ holds then $x \in C^{\mathcal{I}}$ iff $x' \in C^{\mathcal{I}'}$.

Theorem 1 *All concepts of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ are invariant for $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation.*

This theorem can be proved in a similar way as [9, Theorem 3.4]. By this theorem, $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimilarity formalizes indiscernibility by the sublanguage $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$.

An interpretation $\mathcal{I}$ is *finitely branching* (or *image-finite*) w.r.t. $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ if, for every $x \in \Delta^\mathcal{I}$ and every $r \in \Sigma^\dagger_{oR}$:

- the set $\{y \in \Delta^\mathcal{I} \mid r^\mathcal{I}(x, y)\}$ is finite,
- if $I \in \Phi^\dagger$ then the set $\{y \in \Delta^\mathcal{I} \mid r^\mathcal{I}(y, x)\}$ is finite.

Let $x \in \Delta^\mathcal{I}$ and $x' \in \Delta^{\mathcal{I}'}$. We say that x is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*equivalent* to x' if, for every concept C of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, $x \in C^\mathcal{I}$ iff $x' \in C^{\mathcal{I}'}$.

Theorem 2 (The Hennessy–Milner Property) *Let Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, Φ and $\Phi^\dagger$ be sets of DL-features such that $\Phi^\dagger \subseteq \Phi$. Let $\mathcal{I}$ and $\mathcal{I}'$ be interpretations in $\mathcal{L}_{\Sigma,\Phi}$, finitely branching w.r.t. $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ and such that for every $a \in \Sigma^\dagger_I$, $a^\mathcal{I}$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-equivalent to $a^{\mathcal{I}'}$. Assume $U \notin \Phi^\dagger$ or $\Sigma^\dagger_I \neq \emptyset$. Then $x \in \Delta^\mathcal{I}$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-equivalent to $x' \in \Delta^{\mathcal{I}'}$ iff there exists an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation Z between $\mathcal{I}$ and $\mathcal{I}'$ such that $Z(x, x')$ holds.*

This theorem presents *necessary* and *sufficient* for invariant w.r.t finitely branching interpretations. It can be proved in a similar way as [9, Theorem 4.1]. We now have the following corollary.

Corollary 1 *Let Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, let Φ and $\Phi^\dagger$ be sets of DL-features such that $\Phi^\dagger \subseteq \Phi$, and let $\mathcal{I}$ and $\mathcal{I}'$ be finite interpretations in $\mathcal{L}_{\Sigma,\Phi}$. Assume that $\Sigma^\dagger_I \neq \emptyset$ and, for every $a \in \Sigma^\dagger_I$, $a^\mathcal{I}$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-equivalent to $a^{\mathcal{I}'}$. Then, the relation $\{\langle x, x'\rangle \in \Delta^\mathcal{I} \times \Delta^{\mathcal{I}'} \mid x$ is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-equivalent to $x'\}$ is an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation between $\mathcal{I}$ and $\mathcal{I}'$.*

Definition 9 (Auto-bisimulation) An $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*auto-bisimulation of $\mathcal{I}$* is an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation between $\mathcal{I}$ and itself. An $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$ is said to be the *largest* if it is larger than or equal to ($\supseteq$) any other $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$.

Given an interpretation $\mathcal{I}$ in $\mathcal{L}_{\Sigma,\Phi}$, by $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ we denote the largest $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$, and by $\equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ we denote the binary relation on $\Delta^\mathcal{I}$ with the property that $x \equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}} x'$ iff x is $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-equivalent to x'.

Theorem 3 *Let Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, Φ and $\Phi^\dagger$ be sets of DL-features such that $\Phi^\dagger \subseteq \Phi$, and $\mathcal{I}$ be an interpretation in $\mathcal{L}_{\Sigma,\Phi}$. Then:*

1. *the largest $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$ exists and is an equivalence relation,*
2. *if $\mathcal{I}$ is finitely branching w.r.t. $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ then the relation $\equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ is the largest $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$ (i.e., the relations $\equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ and $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ coincide).*

This theorem can be proved as [9, Proposition 5.1 and Theorem 5.2].

We say that a set Y is *split* by a set X if $Y \setminus X \neq \emptyset$ and $Y \cap X \neq \emptyset$. Thus, Y is not split by X if either $Y \subseteq X$ or $Y \cap X = \emptyset$. A partition $\mathbb{Y} = \{Y_1, Y_2, \ldots, Y_n\}$ is *consistent* with a set X if, for every $1 \leq i \leq n$, Y_i is not split by X.

Theorem 4 *Let $\mathcal{I}$ be an interpretation in $\mathcal{L}_{\Sigma,\Phi}$, and let $X \subseteq \Delta^\mathcal{I}$, $\Sigma^\dagger \subseteq \Sigma$ and $\Phi^\dagger \subseteq \Phi$. Then:*

1. *if there exists a concept C of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ such that $X = C^\mathcal{I}$ then the partition of $\Delta^\mathcal{I}$ by $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ is consistent with X,*
2. *if the partition of $\Delta^\mathcal{I}$ by $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ is consistent with X then there exists a concept C of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ such that $C^\mathcal{I} = X$.*

This theorem differs from the ones of [13,22] only in the studied class of DLs. It can be proved analogously as in [22, Theorem 4].

5 Concept learning

Let $\mathcal{I}$ be an information system in $\mathcal{L}_{\Sigma,\Phi}$ treated as a training system. Let $A_d \in \Sigma_C$ be a concept name standing for the "decision attribute". Let $E^+ = \{a \mid a^\mathcal{I} \in A_d^\mathcal{I}\}$ and $E^- = \{a \mid a^\mathcal{I} \in (\neg A_d)^\mathcal{I}\}$ be sets of *positive examples* and *negative examples* of A_d in $\mathcal{I}$, respectively. Suppose that A_d can be expressed by a concept C in a sublanguage $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, where $\Sigma^\dagger \subseteq \Sigma \setminus \{A_d\}$ and $\Phi^\dagger \subseteq \Phi$. How can we learn that concept C on the basis of $\mathcal{I}$, E^+ and E^-? The concept C must satisfy the following conditions:

- $\mathcal{I} \models C(a)$ for all $a \in E^+$, and
- $\mathcal{I} \models \neg C(a)$ for all $a \in E^-$.

We say that a set $Y \subseteq \Delta^\mathcal{I}$ is *split* by E if there exist $a \in E^+$ and $b \in E^-$ such that $\{a^\mathcal{I}, b^\mathcal{I}\} \subseteq Y$. A partition $\mathbb{Y} = \{Y_1, Y_2, \ldots, Y_n\}$ of $\Delta^\mathcal{I}$ is said to be *consistent* with E if, for every $1 \leq i \leq n$, Y_i is not split by E.

Observe that if A_d is definable in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ by a concept C then:

- by the first assertion of Theorem 4, $C^\mathcal{I}$ should be the union of a number of equivalence classes of $\Delta^\mathcal{I}$ w.r.t. $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$,
- we should have that $a^\mathcal{I} \in C^\mathcal{I}$ for all $a \in E^+$, and $a^\mathcal{I} \notin C^\mathcal{I}$ for all $a \in E^-$.

The general method of [22] for learning C is as follows:

1. Starting from the partition $\{\Delta^{\mathcal{I}}\}$, make subsequent granulations to reach the partition corresponding to $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$. The granulation process can be terminated as soon as the current partition is consistent with $A_d^{\mathcal{I}}$ (or when some criteria are met).
2. In the granulation process, we denote the blocks created so far in all steps by $Y_1, Y_2, \ldots, Y_n$, where the current partition $\{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\}$ consists of only some of them. We do not use the same subscript to denote blocks of different contents (i.e., we always use new subscripts obtained by increasing n for new blocks). We take care that, for each $1 \le i \le n$:
   - Y_i is characterized by an appropriate concept C_i (such that $Y_i = C_i^{\mathcal{I}}$),
   - we keep information about whether Y_i is split by $A_d^{\mathcal{I}}$,
   - if $Y_i \subseteq A_d^{\mathcal{I}}$ then $LargestContainer[i] := j$, where $1 \le j \le n$ is the subscript of the largest block Y_j such that $Y_i \subseteq Y_j \subseteq A_d^{\mathcal{I}}$.
3. At the end, let $j_1, j_2, \ldots, j_h$ be all the indices from $\{i_1, i_2, \ldots, i_k\}$ such that $Y_{j_t} \subseteq A_d^{\mathcal{I}}$ for $1 \le t \le h$, and let $\{l_1, l_2, \ldots, l_p\} = \{LargestContainer[j_t] \mid 1 \le t \le h\}$. Let C be a simplified form of $C_{l_1} \sqcup C_{l_2} \sqcup \cdots \sqcup C_{l_p}$. Return C as the result.

In the granulation process, we use selectors to split blocks of the current partition. In the next definitions, we introduce basic selectors, simple selectors and extended selectors in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ for splitting a block. We also prove that using the basic selectors for the granulation process is sufficient to reach the partition corresponding to the equivalence relation $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$.

5.1 Selectors

Definition 10 (Basic Selectors) Let the current partition of $\Delta^{\mathcal{I}}$ be $\{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\}$, where each block Y_{i_t} is characterized by a concept C_{i_t} such that $Y_{i_t} = C_{i_t}^{\mathcal{I}}$. A *basic selector* in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ for splitting a block Y_{i_j} is a concept of one of the following forms:

- A, where $A \in \Sigma_C^\dagger$,
- $A = d$, where $A \in \Sigma_A^\dagger \setminus \Sigma_C^\dagger$ and $d \in range(A)$,
- $\exists\sigma.\{d\}$, where $\sigma \in \Sigma_{dR}^\dagger$ and $d \in range(\sigma)$,
- $\exists r.C_{i_t}$, where $r \in \Sigma_{oR}^\dagger$ and $1 \le t \le k$,
- $\exists r^-.C_{i_t}$, if $I \in \Phi^\dagger$, $r \in \Sigma_{oR}^\dagger$ and $1 \le t \le k$,
- $\{a\}$, if $O \in \Phi^\dagger$ and $a \in \Sigma_I^\dagger$,
- $\le 1\, r$, if $F \in \Phi^\dagger$ and $r \in \Sigma_{oR}^\dagger$,
- $\le 1\, r^-$, if $\{F, I\} \subseteq \Phi^\dagger$ and $r \in \Sigma_{oR}^\dagger$,
- $\ge l\, r$ and $\le m\, r$, if $N \in \Phi^\dagger$, $r \in \Sigma_{oR}^\dagger$, $0 < l \le \#\Delta^{\mathcal{I}}$ and $0 \le m < \#\Delta^{\mathcal{I}}$,
- $\ge l\, r^-$ and $\le m\, r^-$, if $\{N, I\} \subseteq \Phi^\dagger$, $r \in \Sigma_{oR}^\dagger$, $0 < l \le \#\Delta^{\mathcal{I}}$ and $0 \le m < \#\Delta^{\mathcal{I}}$,
- $\ge l\, r.C_{i_t}$ and $\le m\, r.C_{i_t}$, if $Q \in \Phi^\dagger$, $r \in \Sigma_{oR}^\dagger$, $1 \le t \le k$, $0 < l \le \#C_{i_t}^{\mathcal{I}}$ and $0 \le m < \#C_{i_t}^{\mathcal{I}}$,
- $\ge l\, r^-.C_{i_t}$ and $\le m\, r^-.C_{i_t}$, if $\{Q, I\} \subseteq \Phi^\dagger$, $r \in \Sigma_{oR}^\dagger$, $1 \le t \le k$, $0 < l \le \#C_{i_t}^{\mathcal{I}}$ and $0 \le m < \#C_{i_t}^{\mathcal{I}}$,
- $\exists r.\mathsf{Self}$, if $\mathsf{Self} \in \Phi^\dagger$ and $r \in \Sigma_{oR}^\dagger$.

Theorem 5 (On Basic Selectors) *Let $\mathcal{I}$ be an information system in $\mathcal{L}_{\Sigma,\Phi}$, Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, Φ and $\Phi^\dagger$ be sets of DL-features such that $\Phi^\dagger \subseteq \Phi$. To reach the partition corresponding to the equivalence relation $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ it suffices to start from the partition $\{\Delta^{\mathcal{I}}\}$ and repeatedly granulate it using the basic selectors.*

Proof Let $\mathbb{Y} = \{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\}$ be a final partition obtained by starting from the partition $\{\Delta^{\mathcal{I}}\}$ and repeatedly granulating it using the basic selectors. Recall that C_{i_t} is the concept characterizing Y_{i_t}, i.e., $Y_{i_t} = C_{i_t}^{\mathcal{I}}$. Let Z be the equivalence relation corresponding to that partition, i.e., $Z = \{\langle x, x'\rangle \mid x, x' \in Y_{i_j} \text{ for some } 1 \le j \le k\}$.

We first show that Z is an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$.

- Clearly, $Z(a^{\mathcal{I}}, a^{\mathcal{I}})$ always holds for $a \in \Sigma_I^\dagger$.
- Consider the assertion (2) and suppose $Z(x, x')$ holds. Since the partition $\mathbb{Y}$ cannot be granulated anymore using the concept A, we have $Y_{i_j} \subseteq A^{\mathcal{I}}$ or $Y_{i_j} \cap A^{\mathcal{I}} = \emptyset$ for any $1 \le j \le k$. If $x, x' \in Y_{i_j}$ and $Y_{i_j} \subseteq A^{\mathcal{I}}$ then $x, x' \in A^{\mathcal{I}}$. If $x, x' \in Y_{i_j}$ and $Y_{i_j} \cap A^{\mathcal{I}} = \emptyset$ then $x, x' \notin A^{\mathcal{I}}$. Therefore $A^{\mathcal{I}}(x) \Leftrightarrow A^{\mathcal{I}}(x')$.
- Consider the assertion (3) and suppose $Z(x, x')$ holds. Similarly as above, since the partition cannot be granulated anymore using any selector $(B = d)$ with $d \in range(B)$, we have $Y_{i_j} \subseteq (B = d)^{\mathcal{I}}$ or $Y_{i_j} \cap (B = d)^{\mathcal{I}} = \emptyset$ for any $1 \le j \le k$. If $x, x' \in Y_{i_j}$ and $Y_{i_j} \subseteq (B = d)^{\mathcal{I}}$ for some $d \in range(B)$ then $x, x' \in (B = d)^{\mathcal{I}}$, and hence $B(x) = d = B(x')$. If $x, x' \in Y_{i_j}$ and $Y_{i_j} \cap (B = d)^{\mathcal{I}} = \emptyset$ for all $d \in range(B)$ then both $B(x)$ and $B(x')$ are undefined.
- Consider the assertion (4) and suppose that $Z(x, x')$ and $r^{\mathcal{I}}(x, y)$ hold. Let Y_{i_t} be the block of $\mathbb{Y}$ that contains y. We have that $x \in (\exists r.C_{i_t})^{\mathcal{I}}$. Since $\mathbb{Y}$ cannot be granulated anymore using the selector $(\exists r.C_{i_t})$, it follows that $x' \in (\exists r.C_{i_t})^{\mathcal{I}}$. Hence, there exists $y' \in \Delta^{\mathcal{I}}$ such that $r^{\mathcal{I}}(x', y')$ holds and $y' \in C_{i_t}^{\mathcal{I}} = Y_{i_t}$, which means $Z(y, y')$.
- The assertion (5) can be proved analogously to (4).
- For the assertion (6) we can use the argumentation given for (2), with A replaced by $\exists\sigma.\{d\}$.
- The assertion (7) holds when $I \in \Phi^\dagger$ because the argumentation used for proving (4) is still applicable when r is replaced by r^-.

- Similarly, the assertion (8) also holds when $I \in \Phi^\dagger$.
- The assertion (9) holds when $O \in \Phi^\dagger$ because we can use the argumentation given for (2), with A replaced by $\{a\}$.
- Consider the assertion (10) and the case $N \in \Phi^\dagger$. Suppose $Z(x, x')$ holds and let $l = \#\{y \mid r^{\mathcal{I}}(x, y)\}$. Since the partition $\mathbb{Y}$ cannot be granulated anymore using the selectors $\geq l\, r$ (when $l > 0$) and $\leq l\, r$ (when $l < \#\Delta^{\mathcal{I}}$), we have that $x' \in (\geq l\, r)^{\mathcal{I}}$ and $x' \in (\leq l\, r)^{\mathcal{I}}$. Therefore $\#\{y' \mid r^{\mathcal{I}}(x', y')\} = l$.
- The assertion (11) for the case $\{N, I\} \subseteq \Phi^\dagger$ can be proved analogously to (10), using r^- instead of r.
- Consider the assertion (12) and the case $F \in \Phi^\dagger$. Suppose $Z(x, x')$ holds. Using the argumentation given for (2) with A replaced by $(\leq 1\, r)$, we can derive that $x \in (\leq 1\, r)^{\mathcal{I}}$ iff $x' \in (\leq 1\, r)^{\mathcal{I}}$. Therefore, $\#\{y \mid r^{\mathcal{I}}(x, y)\} \leq 1 \Leftrightarrow \#\{y' \mid r^{\mathcal{I}}(x', y')\} \leq 1$.
- The assertion (13) for the case $\{F, I\} \subseteq \Phi^\dagger$ can be proved analogously to (12), using r^- instead of r.
- Consider the assertion (14) and the case $Q \in \Phi^\dagger$. Suppose $Z(x, x')$ holds. Let $S = \{y \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x, y)\}$ and $S' = \{y' \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x', y')\}$. Clearly, S and S' are finite. For the sake of contradiction, suppose there does not exist any bijection $h : S \to S'$ such that $h \subseteq Z$. Thus, there must exist $1 \leq t \leq k$ such that $\#(S \cap Y_{i_t}) = l \neq m = \#(S' \cap Y_{i_t})$. If $l > m$ then $x \in (\geq l\, r.C_{i_t})^{\mathcal{I}}$ but $x' \notin (\geq l\, r.C_{i_t})^{\mathcal{I}}$. If $m > l$ then $x \notin (\geq m\, r.C_{i_t})^{\mathcal{I}}$ but $x' \in (\geq m\, r.C_{i_t})^{\mathcal{I}}$. Therefore, x and x' are distinguishable by a basic selector in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, which contradicts the fact that the partition $\mathbb{Y}$ cannot be granulated anymore.
- The assertion (15) for the case $\{Q, I\} \subseteq \Phi^\dagger$ can be proved analogously to (14), by using r^- instead of r.
- The assertion (16) holds because for any $x \in \Delta^{\mathcal{I}}$ we can choose $x' = x$. Similarly, the assertion (17) holds because for any $x' \in \Delta^{\mathcal{I}}$ we can choose $x = x'$.
- The assertion (18) holds when $\mathsf{Self} \in \Phi^\dagger$ because we can use the argumentation given for (2), with A replaced by $\exists r.\mathsf{Self}$.

We now show that Z is the largest $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$. At each step of the granulation process, the equivalence relation corresponding to the current partition is a superset of $\equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ because each block of the current partition is characterized by a concept. Thus, at the end, we have $Z \supseteq \equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$. Since Z is an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$ and $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ is the largest $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$, we have that $Z \subseteq \sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$. Since $\equiv_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ and $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$ coincide (Theorem 3), we can conclude that $Z = \sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$.

Basic selectors are adequate to granulate $\{\Delta^{\mathcal{I}}\}$ to reach the partition corresponding to the equivalence relation $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$. However, to prevent overfitting to the training system $\mathcal{I}$ and obtain as simple as possible definitions for the learnt concept, it is worth to consider also *simple selectors* defined below although they are expressible by the basic selectors over $\mathcal{I}$. Most of them, except the ones related to attributes, were proposed in [22].

Definition 11 (Simple Selectors) Let $Y_1, \ldots, Y_n$ be the blocks that have been created so far in the process of granulating $\{\Delta^{\mathcal{I}}\}$, where each block Y_i is characterized by a concept C_i such that $Y_i = C_i^{\mathcal{I}}$.[4] A *simple selector* in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ for splitting a block is either a basic selector or a concept of one of the following forms:

- $A \leq d$ and $A < d$, where $A \in \Sigma^\dagger_{nA}$, $d \in range(A)$ and d is not a minimal element of $range(A)$,
- $A \geq d$ and $A > d$, where $A \in \Sigma^\dagger_{nA}$, $d \in range(A)$ and d is not a maximal element of $range(A)$,
- $\exists r.\top$, $\exists r.C_i$ and $\forall r.C_i$, where $r \in \Sigma^\dagger_{oR}$ and $1 \leq i \leq n$,
- $\exists r^-.\top$, $\exists r^-.C_i$ and $\forall r^-.C_i$, if $I \in \Phi^\dagger$, $r \in \Sigma^\dagger_{oR}$ and $1 \leq i \leq n$,
- $\geq l\, r.C_i$ and $\leq m\, r.C_i$, if $Q \in \Phi^\dagger$, $r \in \Sigma^\dagger_{oR}$, $1 \leq i \leq n$, $0 < l \leq \#C_i^{\mathcal{I}}$ and $0 \leq m < \#C_i^{\mathcal{I}}$,
- $\geq l\, r^-.C_i$ and $\leq m\, r^-.C_i$, if $\{Q, I\} \subseteq \Phi^\dagger$, $r \in \Sigma^\dagger_{oR}$, $1 \leq i \leq n$, $0 < l \leq \#C_i^{\mathcal{I}}$ and $0 \leq m < \#C_i^{\mathcal{I}}$.

Our experiments showed that the resulting concepts obtained by the learning method that use only simple selectors to split blocks have the following characteristics:

- the length of the resulting concept is usually long,
- the accuracy, precision, recall and F1 measures of the resulting classifier are not high for new objects.

The main reason is that simple selectors are not advanced enough for granulating partitions. We propose to use a new kind of selectors, called *extended selectors*.

Let $\mathbb{D}$ be a set of available selectors (at the beginning, $\mathbb{D} = \emptyset$). At each step in the granulation process, selectors are created and added into $\mathbb{D}$. Together with the current partition $\mathbb{Y}$, we have $\mathbb{D} = \{D_1, D_2, \ldots, D_h\}$, called the *current set of selectors*. Extended selectors are defined using the current set of selectors and object roles of the sublanguage as follows.

Definition 12 (Extended Selectors) Let the current set of selectors be $\mathbb{D} = \{D_1, D_2, \ldots, D_h\}$. An *extended selector* in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ for splitting a block is a concept of one of the following forms:

- $\exists r.D_u$ and $\forall r.D_u$, where $r \in \Sigma^\dagger_{oR}$ and $D_u \in \mathbb{D}$,
- $\exists r^-.D_u$ and $\forall r^-.D_u$, if $I \in \Phi^\dagger$, $r \in \Sigma^\dagger_{oR}$ and $D_u \in \mathbb{D}$,

[4] The current partition of $\Delta^{\mathcal{I}}$ may consist of only some of these blocks.

- $\geq\ l\, r.D_u$ and $\leq\ m\, r.D_u$, if $Q \in \Phi^{\dagger}$, $r \in \Sigma^{\dagger}_{oR}$, $D_u \in \mathbb{D}$, $0 < l \leq \#D_u^{\mathcal{I}}$ and $0 \leq m < \#D_u^{\mathcal{I}}$,
- $\geq\ l\, r^{-}.D_u$ and $\leq\ m\, r^{-}.D_u$, if $\{Q, I\} \subseteq \Phi^{\dagger}$, $r \in \Sigma^{\dagger}_{oR}$, $D_u \in \mathbb{D}$, $0 < l \leq \#D_u^{\mathcal{I}}$ and $0 \leq m < \#D_u^{\mathcal{I}}$.

Extended selectors are useful for the granulation process. Using them, we have more selectors that can be used to split blocks and we can obtain better partitions than using only simple selectors. Furthermore, using extended selectors, the number of iterations of the main loop of the learning method is usually reduced significantly. This leads to simpler resulting concepts with higher accuracy in classifying new objects.

5.2 Simplicity of concepts

There are different normal forms for concepts [3,21]. Using a normal form, one can represent concepts in a unified way. We propose below a new normal form, which extends the one of [21]. The normal form of a concept is obtained by applying the following normalization rules:

1. concepts are represented in the negation normal form,
2. a conjunction $C_1 \sqcap C_2 \sqcap \cdots \sqcap C_n$ is represented by an "and"-set, denoted by $\sqcap\{C_1, C_2, \ldots, C_n\}$,
3. $\sqcap\{C\}$ is replaced by C,
4. $\sqcap\{\sqcap\{C_1, C_2, \ldots, C_i\}, C_{i+1}, \ldots, C_n\}$ is replaced by $\sqcap\{C_1, C_2, \ldots, C_n\}$,
5. $\sqcap\{\top, C_1, C_2, \ldots, C_n\}$ is replaced by $\sqcap\{C_1, C_2, \ldots, C_n\}$,
6. $\sqcap\{\bot, C_1, C_2, \ldots, C_n\}$ is replaced by $\bot$,
7. if $C_i \sqsubseteq C_j$ and $1 \leq i \neq j \leq n$, then remove C_j from $\sqcap\{C_1, C_2, \ldots, C_n\}$,
8. if $C_i = \overline{C}_j$ and $1 \leq i \neq j \leq n$, then $\sqcap\{C_1, C_2, \ldots, C_n\}$ is replaced by $\bot$, where $\overline{C}$ is the normal form of $\neg C$,
9. $\forall R.(\sqcap\{C_1, C_2, \ldots, C_n\})$ is replaced by $\sqcap\{\forall R.C_1, \forall R.C_2, \ldots, \forall R.C_n\}$,
10. $\forall R.\top$ is replaced by $\top$,
11. $\leq\ n\, R.\bot$ is replaced by $\top$,
12. $\geq\ n\, R.\bot$ is replaced by $\bot$ when $n > 0$,
13. $\geq\ 1\, R.C$ is replaced by $\exists R.C$,
14. the rules "dual" to the rules 2–10 (for example, dually to the rule 5, $\sqcup\{\bot, C_1, C_2, \ldots, C_n\}$ is replaced by $\sqcup\{C_1, C_2, \ldots, C_n\}$).

Each step of the partition process may generate many concepts. To avoid repetition of the same concepts in the storage, we design the data structure appropriately. If two concepts have the same normal form then they are represented only once in the data structure by the normal form.

Example 3 Let A and B be concept names, r and s be object role names. Let $C = \neg(\exists r.\neg A \sqcap (B \sqcup \forall s.A)) \sqcap \neg(\geq\ 3\, r.A \sqcup \neg B)$. The negation normal form of C is $(\forall r.A \sqcup (\neg B \sqcap \exists s.\neg A)) \sqcap (\leq\ 2\, r.A \sqcap B)$. The normal form of C is $\sqcap\{\sqcup\{\forall r.A, \sqcap\{\neg B, \exists s.\neg A\}\}, \leq\ 2\, r.A, B\}$.

Definition 13 (Modal Depth) Let C be a concept in the normal form. The *modal depth* of C, denoted by $\mathsf{mdepth}(C)$, is defined to be:

- 0 if C is of the form $\top$, $\bot$, A, $A = d$, $A \neq d$, $A > d$, $A \geq d$, $A < d$ or $A \leq d$,
- $\mathsf{mdepth}(D)$ if C is the normal form of $\neg D$,
- 1 if C is of the form $\exists\sigma.\{d\}$, $\exists r.\mathsf{Self}$, $\geq\ n\ R$ or $\leq\ n\ R$,
- $1+\mathsf{mdepth}(D)$ if C is of the form $\exists R.D, \forall R.D, \geq\ n\ R.D$ or $\leq\ n\ R.D$,
- $\max\{\mathsf{mdepth}(D_1), \mathsf{mdepth}(D_2), \ldots, \mathsf{mdepth}(D_n)\}$ if C is of the form $\sqcap\{D_1, D_2, \ldots, D_n\}$ or $\sqcup\{D_1, D_2, \ldots, D_n\}$.

Definition 14 (Length) Let C be a concept in the normal form. The *length* of C, denoted by $\mathsf{length}(C)$, is defined to be:

- 0 if C is $\top$ or $\bot$,
- 1 if C is of the form A, $A = d$, $A \neq d$, $A > d$, $A \geq d$, $A < d$ or $A \leq d$,
- $\mathsf{length}(D)$ if C is the normal form of $\neg D$,
- 3 if C is of the form $\exists\sigma.\{d\}$, $\exists r.\mathsf{Self}$, $\geq n\ R$ or $\leq n\ R$,
- $2 + \mathsf{length}(D)$ if C is of the form $\exists R.D$ or $\forall R.D$,
- $3 + \mathsf{length}(D)$ if C is of the form $\geq\ n\ R.D$ or $\leq\ n\ R.D$,
- $1 + \mathsf{length}(D_1) + \mathsf{length}(D_2) + \cdots + \mathsf{length}(D_n)$ if C is of the form $\sqcap\{D_1, D_2, \ldots, D_n\}$ or $\sqcup\{D_1, D_2, \ldots, D_n\}$.

In this paper, concepts are represented in the normal form (i.e., the negation constructor appears only in front of atomic concepts). For this reason, $\mathsf{length}(\neg D)$ is defined to be $\mathsf{length}(D)$.

Example 4 Let A and B be concept names, r and s be object role names. We have:

- $\mathsf{mdepth}(\sqcap\{\neg A, \geq\ 2\, r.(\exists r.\top), \exists s.B\}) = 2$,
- $\mathsf{length}(\sqcap\{\neg A, \geq\ 2\, r.(\exists r.\top), \exists r.B\}) = 10$.

Given concepts C and D in the normal form, we say that C is *simpler* than D if:

- $\mathsf{length}(C) < \mathsf{length}(D)$, or
- $\mathsf{length}(C) = \mathsf{length}(D)$ and $\mathsf{mdepth}(C) \leq \mathsf{mdepth}(D)$.

In fact, the modal depth of a concept is usually restricted by a very small value, while the length of a concept is

usually large. Hence, the differences between the modal depths of concepts are usually small. In contrast, the differences between the lengths of concepts may be very large. Thus, we choose the length of a concept as the main factor for deciding whether a concept is simpler than another.

Let $\mathbb{C} = \{C_1, C_2, \ldots, C_n\}$ be a set of concepts. A concept $C_i \in \mathbb{C}$ is said to be the *simplest* if it is simpler than any other concept in $\mathbb{C}$.

5.3 Information gain in the context of DLs

Let X and Y be subsets of $\Delta^{\mathcal{I}}$, where X plays the role of a set of positive examples for the concept to be learnt.

Definition 15 *Entropy* of Y w.r.t. X in $\Delta^{\mathcal{I}}$, denoted by $E_{\Delta^{\mathcal{I}}}(Y/X)$, is defined as follows:

$$E_{\Delta^{\mathcal{I}}}(Y/X) = \begin{cases} 0, \text{ if } Y \cap X = \emptyset \text{ or } Y \subseteq X \\ -\frac{\#XY}{\#Y}\log_2\frac{\#XY}{\#Y} - \frac{\#\overline{X}Y}{\#Y}\log_2\frac{\#\overline{X}Y}{\#Y}, \text{otherwise} \end{cases} \quad (19)$$

where XY stands for the set $X \cap Y$ and $\overline{X}Y$ stands for the set $\overline{X} \cap Y$.

Entropy is an information—theoretic measure of the "uncertainty" contained in an information system treated as a training set, due to the presence of more than one possible classification. It takes the minimum value (zero) iff all the examples have the same classification, in which case there is only one non-empty class, for which the probability is 1. Entropy takes its maximum value when the examples are equally distributed amongst the two possible sets.

Remark 2 According to (19), $E_{\Delta^{\mathcal{I}}}(Y/X) = 0$ iff Y is not split by X.

We want to determine which attribute in a given set of training features is most useful for discriminating between the classes to be learnt. In [25] Quinlan proposed to use information gain for deciding the ordering of attributes in the nodes of a decision tree. In the context of DLs, we formulate information gain for a selector to split a block in a partition as follows.

Definition 16 *Information gain* for a selector D to split Y w.r.t. X in $\Delta^{\mathcal{I}}$, denoted by $IG_{\Delta^{\mathcal{I}}}(Y/X, D)$, is defined as follows:

$$IG_{\Delta^{\mathcal{I}}}(Y/X, D) = E_{\Delta^{\mathcal{I}}}(Y/X) - \left(\frac{\#D^{\mathcal{I}}Y}{\#Y}E_{\Delta^{\mathcal{I}}}(D^{\mathcal{I}}Y/X) + \frac{\#\overline{D^{\mathcal{I}}}Y}{\#Y}E_{\Delta^{\mathcal{I}}}(\overline{D^{\mathcal{I}}}Y/X)\right) \quad (20)$$

where $D^{\mathcal{I}}Y$ stands for the set $D^{\mathcal{I}} \cap Y$ and $\overline{D^{\mathcal{I}}}Y$ stands for the set $\overline{D^{\mathcal{I}}} \cap Y$.

The information gain is based on the decrease in entropy after an information system is split on a block by a selector. Constructing a decision tree is all about finding a block as well as a selector that returns the highest information gain.

In the case $\Delta^{\mathcal{I}}$ and X are clear from the context, we write $E(Y)$ and $IG(Y, D)$ instead of $E_{\Delta^{\mathcal{I}}}(Y/X)$ and $IG_{\Delta^{\mathcal{I}}}(Y/X, D)$, respectively.

5.4 A bisimulation-based concept learning algorithm

We now describe a bisimulation-based concept learning method for information systems in DLs via Algorithm 1. It is based on the general method of [22]. In this algorithm, the kinds of selectors applied in the granulation process depend on Steps 2 and 19. We can use only basic selectors, only simple selectors or both of simple selectors and extended selectors.

Function `chooseBlockSelector` in Step 4 is used to choose the best block and selector at the current moment in the granulation process. This function applies information gain measure while taking into account also simplicity of selectors. Suppose that we have the current partition $\mathbb{Y} = \{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\}$ and the current set of selectors $\mathbb{D} = \{D_1, D_2, \ldots, D_h\}$.

For each block $Y_{i_j} \in \mathbb{Y}$ (where $1 \leq j \leq k$), let S_{i_j} be the simplest selector from the set $\arg\max D_u \in \mathbb{D}\{IG(Y_{i_j}, D_u)\}$. If a block Y_{i_j} of $\mathbb{Y}$ is chosen to be split then S_{i_j} is the choice for splitting Y_{i_j}. Note that the information gain is used for the selection.

After choosing the selectors for blocks, we decide which block should be split first. We choose a block Y_{i_j} such that applying the selector S_{i_j} to split Y_{i_j} maximizes the information gain. That is, we split a block $Y_{i_j} \in \arg\max Y_{i_j} \in \mathbb{Y}\{IG(Y_{i_j}, S_{i_j})\}$ first.

For Step 19, after splitting a block, we have a new partition. We also create and add new selectors to the current set of selectors. This set is used to continue granulating the new partition.

Similarly to the method proposed in [22], we keep information about whether Y_i is split by E and set *Largest Container*$[i] := j$ if it is not, where $1 \leq j \leq n$ is the subscript of the largest block Y_j such that $Y_i \subseteq Y_j$ and Y_j is not split by E.

For Step 24, the meaning of $\mathbb{J}$ is to collect subscripts l such that Y_l is the largest block which is not split by E and $Y_l \subseteq \{a^{\mathcal{I}} \mid a \in E^+\}$.

Function `chooseBlockSelector(`$\mathbb{Y}, \mathbb{D}$`)`

Output: $\langle Y_{i_j}, S_{i_j} \rangle$ such that $IG(Y_{i_j}, S_{i_j})$ is maximal, where $Y_{i_j} \in \mathbb{Y}$ and $S_{i_j} \in \mathbb{D}$

$BS := \emptyset$;
foreach $Y_{i_j} \in \mathbb{Y}$ **do**
  foreach $D_u \in \mathbb{D}$ **do**
    compute $IG(Y_{i_j}, D_u)$;
  $\mathbb{S} := \arg\max_{D_u \in \mathbb{D}} \{IG(Y_{i_j}, D_u)\}$;
  let S_{i_j} be the simplest concept in $\mathbb{S}$;
  $BS := BS \cup \{\langle Y_{i_j}, S_{i_j} \rangle\}$;
choose $\langle Y_{i_j}, S_{i_j} \rangle \in BS$ such that $IG(Y_{i_j}, S_{i_j})$ is maximal;
return $\langle Y_{i_j}, S_{i_j} \rangle$;

In practice, it is very difficult to obtain the condition "Y_j is not split by E". Therefore, we can approximate this condition by a hight rate r of positive examples in each block (for example, the parameter r may be 90 %).

Remark 3 The expression $\bigsqcup \mathbb{C}$ in Step 27 is understood as follows:

- $\bigsqcup \mathbb{C} = C_{l_1} \sqcup C_{l_2} \sqcup \cdots \sqcup C_{l_p}$, if $\mathbb{C} = \{C_{l_1}, C_{l_2}, \ldots, C_{l_p}\}$,
- $\bigsqcup \mathbb{C} = \bot$, if $\mathbb{C} = \emptyset$.

The decision trees generated in the granulation process can be very large. They may give complex concepts which overfit the training datasets and poorly classify new objects. In Step 28, simplifying a concept can be done using some techniques to reduce the size of the decision trees and the length of the resulting concepts. A validating dataset is used to prune the decision tree as well as to reduce the resulting concept. The goal is to increase the accuracy. Function `Simplify` can use the following techniques:

- The first technique is pruning. Given a decision tree generated by the granulation process, it allows to reduce the size of the tree by removing parts that provide little power in classifying objects. Pruning can be done top-down or bottom-up. For example, using the bottom-up fashion, one can repeatedly choose a node whose successors are leaves and cut off the successors if the average accuracy of the resulting concept is not worse on the training and validating datasets.
- The second technique is based on replacement. The resulting concept is usually a disjunction $C_1 \sqcup C_2 \sqcup \cdots \sqcup C_n$. In the case C_i is a too complex concept, we consider replacing it by a simpler one from the set of selectors if they have the same denotation in the considered information system. The replacement is done only when the accuracy of the resulting concept is not worse on the validating dataset.
- The third technique is simplification. De Morgan's laws are used to reduce the resulting concept to an equivalent one.

6 Illustrative examples

In this section, we present three examples to illustrate Algorithm 1. Example 5 uses only basic selectors, Example 6 uses only simple selectors, and Example 7 uses both simple selectors and extended selectors for the granulation process. The aim is to show the effectiveness of different kinds of selectors.

Example 5 Consider the information system $\mathcal{I}$ given in Example 2. Assume that we want to learn a definition of the concept *ExcellentPub* (i.e., $E = \langle E^+, E^- \rangle$ with $E^+ = \{P_4, P_6\}$ and $E^- = \{P_1, P_2, P_3, P_5\}$) in the sublanguage

Algorithm 1: bisimualtion-based concept learning for information systems in description logics

Input: $\mathcal{I}$, $\Sigma^\dagger$, $\Phi^\dagger$, $E = \langle E^-, E^+ \rangle$
Output: A concept C such that:
- $\mathcal{I} \models C(a)$ for all $a \in E^+$, and
- $\mathcal{I} \not\models C(a)$ for all $a \in E^-$.

1 $n := 1$; $Y_1 := \Delta^{\mathcal{I}}$; $\mathbb{Y} := \{Y_1\}$; $\mathbb{D} := \emptyset$;
2 create and add selectors to $\mathbb{D}$;
3 **while** $\mathbb{Y}$ *is not consistent with* E **do**
4 $\langle Y_{i_j}, S_{i_j} \rangle :=$ `chooseBlockSelector(`$\mathbb{Y}, \mathbb{D}$`)`;
5 **if** (Y_{i_j} *is not split by* $S_{i_j}^{\mathcal{I}}$) **then**
6 break;
7 $s := n + 1$; $t := n + 2$; $n := n + 2$;
8 $Y_s := Y_{i_j} \cap S_{i_j}^{\mathcal{I}}$; $C_s := C_{i_j} \sqcap S_{i_j}$;
9 $Y_t := Y_{i_j} \cap (\neg S_{i_j})^{\mathcal{I}}$; $C_t := C_{i_j} \sqcap \neg S_{i_j}$;
10 **if** (Y_{i_j} *is not split by* E) **then**
11 $LargestContainer[s] := LargestContainer[i_j]$;
12 $LargestContainer[t] := LargestContainer[i_j]$;
13 **else**
14 **if** (Y_s *is not split by* E) **then**
15 $LargestContainer[s] := s$;
16 **if** (Y_t *is not split by* E) **then**
17 $LargestContainer[t] := t$;
18 $\mathbb{Y} := \mathbb{Y} \cup \{Y_s, Y_t\} \setminus \{Y_{i_j}\}$;
19 create and add new selectors to $\mathbb{D}$;
20 $\mathbb{J} := \emptyset$; $\mathbb{C} := \emptyset$;
21 **if** ($\mathbb{Y}$ *is consistent with* E) **then**
22 **foreach** $Y_{i_j} \in \mathbb{Y}$ **do**
23 **if** (Y_{i_j} *contains some* $a^{\mathcal{I}}$ *with* $a \in E^+$) **then**
24 $\mathbb{J} := \mathbb{J} \cup \{LargestContainer[i_j]\}$;
25 **foreach** $l \in \mathbb{J}$ **do**
26 $\mathbb{C} := \mathbb{C} \cup \{C_l\}$;
27 $C := \bigsqcup \mathbb{C}$;
28 **return** $C_{rs} :=$ `Simplify` (C);
29 **else**
30 **return** failure;

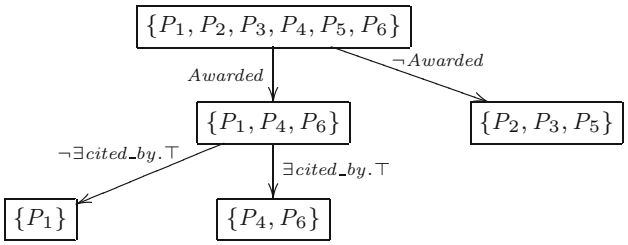

Fig. 3 A decision tree illustrating Example 5

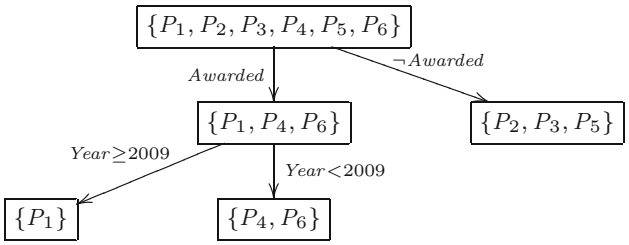

Fig. 4 A decision tree illustrating Example 6

$\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, where $\Sigma^\dagger = \{Awarded, cited\ \_by\}$ and $\Phi^\dagger = \emptyset$. Basic selectors are used in the granulation process. The steps are as follows:

1. $Y_1 := \Delta^{\mathcal{I}}$, $partition := \{Y_1\}$
2. According to the information gain measure, the best basic selector for splitting Y_1 is *Awarded*. Using it we obtain:
   - $Y_2 := \{P_1, P_4, P_6\}$, $C_2 := Awarded$
   - $Y_3 := \{P_2, P_3, P_5\}$, $C_3 := \neg Awarded$
   - $LargestContainer[3] := 3$ (as Y_3 is not split by E)
   - $partition := \{Y_2, Y_3\}$
3. According to the information gain measure, the best basic selectors for splitting Y_2 are $\exists cited\ \_by.\top$, $\exists cited\ \_by.C_2$ and $\exists cited\ \_by.C_3$. All of them split Y_2 in the same way. We choose $\exists cited\ \_by.\top$, as it is the simplest one. Using it we obtain:
   - $Y_4 := \{P_1\}$, $C_4 := C_2 \sqcap \neg\exists cited\ \_by.\top$
   - $LargestContainer[4] := 4$ (as Y_4 is not split by E)
   - $Y_5 := \{P_4, P_6\}$, $C_5 := C_2 \sqcap \exists cited\ \_by.\top$
   - $LargestContainer[5] := 5$ (as Y_5 is not split by E)
   - $partition := \{Y_3, Y_4, Y_5\}$

The obtained partition is consistent with E, having only Y_5 contains P_4, P_6 with $P_4, P_6 \in E^+$. (It is not yet the partition corresponding to $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$.)

Since $LargestContainer[5] = 5$, we have $\mathbb{C} = \{C_5\}$. The returned concept is $C = C_5 = Awarded \sqcap \exists cited\ \_by.\top$. It is simple and "comparable" with the original definition $ExcellentPub \equiv (GoodPub \sqcap Awarded) \equiv (Awarded \sqcap \geq 2\, cited\ \_by)$.

The granulation process is illustrated by the decision tree given in Fig. 3.

The following example shows that when no role is used, i.e., when $\Sigma^\dagger_{oR} \cup \Sigma^\dagger_{dR} = \emptyset$, the above proposed method is similar the traditional learning method based on decision trees.

Example 6 Consider again the information system $\mathcal{I}$ given in Example 2. Assume that we want to learn a concept definition of $X = \{P_4, P_6\}$ (i.e., $E = \langle E^+, E^- \rangle$ with $E^+ = \{P_4, P_6\}$ and $E^- = \{P_1, P_2, P_3, P_5\}$) in the sublanguage $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, where $\Sigma^\dagger = \{Awarded, Year\}$ and $\Phi^\dagger = \emptyset$. Simple selectors are used in the granulation process. The steps are as follows:

1. $Y_1 := \Delta^{\mathcal{I}}$, $partition := \{Y_1\}$
2. According to the information gain measure, the best simple selectors for the current step are *Awarded* and $Year \geq 2008$. We choose *Awarded* and obtain:
   - $Y_2 := \{P_1, P_4, P_6\}$, $C_2 := Awarded$
   - $Y_3 := \{P_2, P_3, P_5\}$, $C_3 := \neg Awarded$
   - $LargestContainer[3] := 3$ (as Y_3 is not split by E)
   - $partition := \{Y_2, Y_3\}$.
3. According to the information gain measure, one of the best simple selectors for splitting Y_2 is $Year \geq 2009$. Using it we obtain:
   - $Y_4 := \{P_1\}$, $C_4 := C_2 \sqcap (Year \geq 2009)$
   - $LargestContainer[4] := 4$ (as Y_4 is not split by E)
   - $Y_5 := \{P_4, P_6\}$, $C_5 := C_2 \sqcap (Year < 2009)$
   - $LargestContainer[5] := 5$ (as Y_5 is not split by E)
   - $partition := \{Y_3, Y_4, Y_5\}$.

The obtained partition is consistent with E, having only Y_5 contains P_4, P_6 with $P_4, P_6 \in E^+$. (It is not yet the partition corresponding to $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$).

Since $LargestContainer[5] = 5$, we have $\mathbb{C} = \{C_5\}$. The returned concept is $C = C_5 = Awarded \sqcap (Year < 2009)$.

The granulation process is illustrated by the decision tree given in Fig. 4.

The following example shows the effectiveness of simple selectors and extended selectors.

Example 7 Let $\Phi = \{I\}$ and

$$\Sigma_I = \{Ava, Britt, Colin, Dave, Ella, Flor, Gigi, Harry\},$$
$$\Sigma_C = \{Human, Male, Female, Nephew, Niece\},$$
$$\Sigma_{dA} = \Sigma_C$$
$$\Sigma_{oR} = \{hasChild, hasParent, hasSibling\},\ \Sigma_{dR} = \emptyset,$$

$$\mathcal{T} = \{hasParent \equiv hasChild^-,$$
$$Human \equiv \top, Female \equiv \neg Male,$$

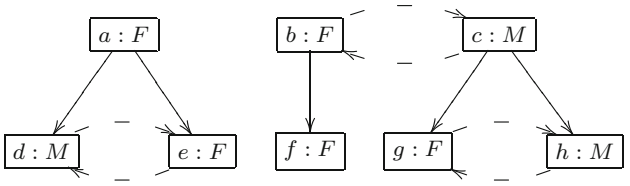

Fig. 5 An illustration for the information system given in Example 7

$$Niece \equiv Female \sqcap \exists hasChild^{-}.(\exists hasSibling.\top),$$
$$Nephew \equiv Male \sqcap \exists hasChild^{-}.(\exists hasSibling.\top)\},$$
$$\mathcal{A} = \{Female(Ava), Female(Britt), Male(Colin),$$
$$Male(Dave), Female(Ella), Female(Flor),$$
$$Female(Gigi), Male(Harry),$$
$$hasChild(Ava, Dave), hasChild(Ava, Ella),$$
$$hasChild(Britt, Flor), hasChild(Colin, Gigi),$$
$$hasChild(Colin, Harry), hasSibling(Britt, Colin),$$
$$hasSibling(Colin, Britt), hasSibling(Dave, Ella),$$
$$hasSibling(Ella, Dave), hasSibling(Gigi, Harry),$$
$$hasSibling(Harry, Gigi)\}.$$

Consider the information system $\mathcal{I}$ specified by:

$$\Delta^{\mathcal{I}} = \{Ava, Britt, Colin, Dave, Ella, Flor, Gigi, Harry\},$$
$$Female^{\mathcal{I}} = \{Ava, Britt, Ella, Flor, Gigi\},$$
$$Male^{\mathcal{I}} = \{Colin, Dave, Harry\},$$
$$hasChild^{\mathcal{I}} = \{\langle Ava, Dave\rangle, \langle Ava, Ella\rangle, \langle Britt, Flor\rangle, \langle Colin, Gigi\rangle, \langle Colin, Harry\rangle\},$$
$$hasParent^{\mathcal{I}} = \{\langle Dave, Ava\rangle, \langle Ella, Ava\rangle, \langle Flor, Britt\rangle, \langle Gigi, Colin\rangle, \langle Harry, Colin\rangle\},$$
$$hasSibling^{\mathcal{I}} = \{\langle Dave, Ella\rangle, \langle Ella, Dave\rangle, \langle Britt, Colin\rangle, \langle Colin, Britt\rangle, \langle Gigi, Harry\rangle, \langle Harry, Gigi\rangle\},$$
$$(hasSibling^{-})^{\mathcal{I}} = hasSibling^{\mathcal{I}},$$
$$Niece^{\mathcal{I}} = \{Flor, Gigi\}, \quad Nephew^{\mathcal{I}} = \{Harry\}.$$

This interpretation is illustrated in Fig. 5. In this figure, each node denotes a person, the letter M stands for $Male$, F stands for $Female$, the solid edges denote assertions of the role $hasChild$, and the dashed edges denote assertions of the role $hasSibling$. We abbreviate Ava to a, $Britt$ to b, …, and $Harry$ to h.

Let $\Sigma^{\dagger} = \{Female, hasChild, hasSibling\}$, $\Phi^{\dagger} = \{I\}$ and $X = \{f, g\}$ (i.e., $E = \langle E^{+}, E^{-}\rangle$ with $E^{+} = \{f, g\}$ and $E^{-} = \{a, b, c, d, e, h\}$). One can think of X as the set of instances of the concept $Niece \equiv Female \sqcap \exists hasChild^{-}.(\exists hasSibling.\top)$ in $\mathcal{I}$. We illustrate the steps of the granulation process by decision trees.

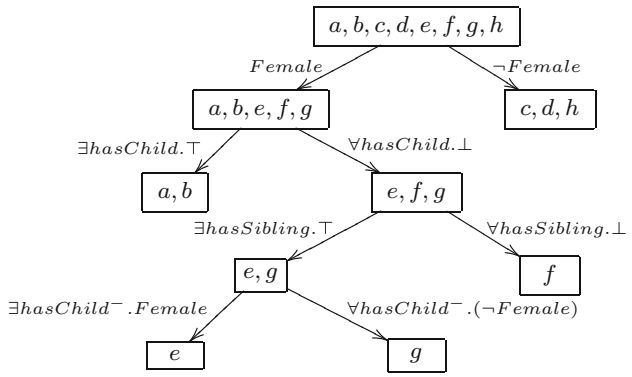

Fig. 6 A decision tree illustrating the granulation process using simple selectors

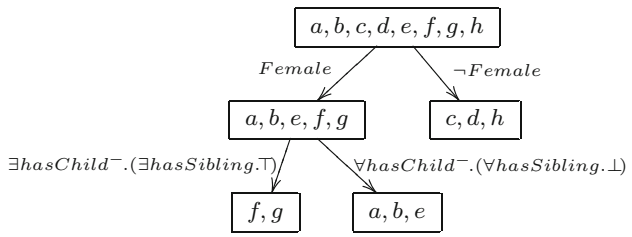

Fig. 7 A decision tree illustrating the granulation process using both simple selectors and extended selector

1. Consider learning a definition of X in $\mathcal{L}_{\Sigma^{\dagger},\Phi^{\dagger}}$ using simple selectors. The steps are illustrated by the decision tree in Fig. 6. The resulting concept is $(Female \sqcap \forall hasChild.\bot \sqcap \exists hasSibling.\top \sqcap \forall hasChild^{-}.(\neg Female)) \sqcup (Female \sqcap \forall hasChild.\bot \sqcap \forall hasSibling.\bot)$. This concept can be simplified to $Female \sqcap \forall hasChild.\bot \sqcap (\forall hasChild^{-}.(\neg Female) \sqcup \forall hasSibling.\bot)$.
2. Consider learning a definition of X in $\mathcal{L}_{\Sigma^{\dagger},\Phi^{\dagger}}$ using both simple selectors and extended selectors. The steps are illustrated by the decision tree in Fig. 7. The resulting concept is $Female \sqcap \exists hasChild^{-}.(\exists hasSibling.\top)$.

The concept $\exists hasChild^{-}.(\exists hasSibling.\top)$ is an extended selector (in the second case). It is created by applying the second form in Definition 12 to $\exists hasSibling.\top$, which is one of the available selectors in the current set of selectors $\mathbb{D}$.

This example demonstrates that using both simple selectors and extended selectors is better than using only simple selectors. The former reduces the number of iterations of the main loop and the length of the resulting concept.

In Examples 5, 6 and 7, the use of *LargestContainer* does not bring benefits. To see its usefulness we refer the reader to [22].

Table 1 Evaluation results on the WebKB, PokerHand and Family datasets using 100 random concepts in the DL $\mathcal{ALCIQ}$

	Avg. Dep. Res./Org.	Avg. Len. Res./Org.	Avg. Acc. [min; max]	Avg. Pre. [min; max]	Avg. Rec. [min; max]	Avg. F1 [min; max]
WebKB dataset						
Simple selectors	0.82/1.02	6.81/4.41	93.84 ± 13.50	92.09 ± 17.04	92.82 ± 17.32	91.59 ± 16.68
			[33.69; 100.0]	[32.08; 100.0]	[23.08; 100.0]	[27.69; 100.0]
Simple and extended	0.84/1.02	3.40/4.41	94.60 ± 12.20	92.81 ± 15.93	93.14 ± 17.17	92.33 ± 16.17
Selectors			[33.69; 100.0]	[32.08; 100.0]	[23.08; 100.0]	[27.69; 100.0]
PokerHand dataset						
Simple selectors	1.41/2.60	37.02/15.97	97.17 ± 08.61	95.96 ± 14.99	94.95 ± 14.40	94.66 ± 14.64
			[50.57; 100.0]	[01.67; 100.0]	[01.67; 100.0]	[01.67; 100.0]
Simple and extended	1.23/2.60	3.47/15.97	99.44 ± 02.15	98.68 ± 09.08	98.06 ± 09.58	98.18 ± 09.14
Selectors			[83.25; 100.0]	[01.67; 100.0]	[01.67; 100.0]	[01.67; 100.0]
Family dataset						
Simple selectors	2.38/3.34	78.50/18.59	88.50 ± 16.65	90.60 ± 18.57	85.66 ± 22.36	86.09 ± 20.10
			[27.91; 100.0]	[04.55; 100.0]	[07.69; 100.0]	[08.70; 100.0]
Simple and extended	2.29/3.34	10.20/18.59	92.79 ± 14.35	91.99 ± 18.40	91.75 ± 19.82	90.39 ± 19.89
Selectors			[27.91; 100.0]	[04.55; 100.0]	[07.69; 100.0]	[08.70; 100.0]

7 Experimental results

We implemented our method and conducted experiments only for the class $\mathcal{L}_{\Sigma,\Phi}$ of DLs such that $\mathcal{L}$ stands for $\mathcal{ALC}$ (i.e., the role constructors of propositional dynamic logic are discared), $\Sigma_{nA} = \Sigma_{dR} = \emptyset$ and $\Sigma_{dA} = \Sigma_C$ (i.e., $\Sigma = \Sigma_I \cup \Sigma_C \cup \Sigma_R$ with $\Sigma_R = \Sigma_{oR}$), and $\Phi \subseteq \{I, Q\}$.

The $\mathcal{L}_{\Sigma,\Phi}$ language is redefined by removing the redundant items from Definition 3.

By $\mathcal{ALCIQ}$ (resp. $\mathcal{ALCI}$, $\mathcal{ALCQ}$) we denote the $\mathcal{L}_{\Sigma,\Phi}$ language with $\mathcal{L} = \mathcal{ALC}$ and $\Phi = \{I, Q\}$ (resp. $\Phi = \{I\}$, $\Phi = \{Q\}$).

The difficulty we encountered is that there were too few available datasets with linked data that can directly be used for concept learning using our setting. We had to build/get datasets for our setting from some resources on the Internet, including the WebKB [28], PokerHand [5] and Family datasets.

The **WebKB** dataset consists of information about web pages of four departments of computer science (Cornell, Washington, Wisconsin, and Texas universities). It contains information about 877 web pages (objects) and 1608 links between them of one relationship ($cites$).

Each object in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary (1703 words). It is assigned one of five concepts indicating the type of web page: $Course$, $Faculty$, $Student$, $Project$ and $Staff$.

We use data from two of the four departments for training (230 objects) and validating (195 objects). The two remaining ones (452 objects) are used for testing.

The **Family** dataset consists of information about people from five families (British Royal, Bush, Roberts, Romanov and Stevens families). It contains information about 943 people (objects) and 11,062 links between them of seven relationships ($hasChild$, $hasSon$, $hasDaughter$, $hasWife$, $hasHusband$, $hasBrother$ and $hasSister$). Each object is an instance of either the concept $Male$ or $Female$. The data from two of the five families are used for training (437 objects) and validating (49 objects). The three remaining ones (457 objects) are used for testing.

The **PokerHand** dataset is a subset taken from UCI Machine Learning Repository. It consists of information about 2542 hands, 12,710 cards, 119 features of cards (15,371 objects in total) and 65,220 links between them of six relationships ($hasCard$, $hasRank$, $hasSuit$, $sameRank$, $nextRank$, $sameSuit$). The goal is to predict which among nine classes should be assigned to a hand. These classes are "$one\ pair$", "$two\ pairs$", "$three\ of\ a\ kind$", "$straight$", "$flush$", "$full\ house$", "$four\ of\ a\ kind$", "$straight\ flush$" and "$royalflush$". Because the number of hands in the classes "$royal\ flush$", "$straight\ flush$" and "$four\ of\ a\ kind$" is very small, we remove these classes from our dataset. The dataset is divided into seven subsets. Two subsets are used for training (1343 objects) and validating (1343 objects). The five remaining ones are used for testing (12,685 objects).

We have used Java language (JDK 1.6) to implement the bisimulation-based concept learning method for information systems in the DL $\mathcal{ALCIQ}$ using simple selectors and extended selectors as well as the information gain discussed in Sect. 5.3. The reduction techniques mentioned in that sec-

Table 2 Evaluation results on the Family dataset using five popular concepts in the DL $\mathcal{ALCI}$

	Dep. Res.	Len. Res.	Avg. Acc. [min; max]	Avg. Pre. [min; max]	Avg. Rec. [min; max]	Avg. F1 [min; max]
Concept: $Grandparent = \exists hasChild.(\exists hasChild.\top)$						
Simple selectors	2.00	4.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Simple and extended	2.00	4.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
Selectors			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Concept: $Grandfather = Male \sqcap \exists hasChild.(\exists hasChild.\top)$						
Simple selectors	2.00	36.00	95.90 ± 01.39	87.38 ± 06.81	79.15 ± 17.38	81.44 ± 08.35
			[94.28; 97.67]	[80.00; 96.43]	[57.45; 100.0]	[72.00; 92.31]
Simple and extended	2.00	07.00	99.46 ± 00.77	100.0 ± 00.00	95.74 ± 6.02	97.73 ± 03.21
Selectors			[98.37; 100.0]	[100.0; 100.0]	[87.23; 100.0]	[93.18; 100.0]
Concept: $Grandmother = Female \sqcap \exists hasChild.(\exists hasChild.\top)$						
Simple selectors	2.00	18.00	89.74 ± 01.30	100.0 ± 00.00	15.32 ± 04.47	26.31 ± 06.85
			[88.37; 91.49]	[100.0; 100.0]	[09.30; 20.00]	[17.02; 33.33]
Simple and extended	2.00	07.00	99.91 ± 00.13	100.0 ± 00.00	99.22 ± 01.10	99.61 ± 00.55
Selectors			[99.73; 100.0]	[100.0; 100.0]	[97.67; 100.0]	[98.82; 100.0]
Concept: $Niece = Female \sqcap \exists hasChild^{-}.(\exists hasBrother.\top \sqcup \exists hasSister.\top)$						
Simple selectors	3.00	151.00	85.57 ± 09.47	57.92 ± 32.09	64.70 ± 29.35	60.69 ± 31.33
			[72.21; 93.02]	[12.66; 83.33]	[23.26; 87.50]	[16.39; 83.33]
Simple and extended	2.00	11.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
Selectors			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Concept: $Nephew = Male \sqcap \exists hasChild^{-}.(\exists hasBrother.\top \sqcup \exists hasSister.\top)$						
Simple selectors	3.00	178.00	91.40 ± 05.74	77.04 ± 26.30	88.40 ± 01.99	79.82 ± 17.72
			[83.38; 95.74]	[40.22; 100.0]	[86.05; 90.91]	[54.81; 93.75]
Simple and extended	2.00	11.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
Selectors			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]

tion have been integrated into our program. The program and datasets can be downloaded from http://www.mimuw.edu.pl/~ttluong/ConceptLearning.rar.

We tested our method on the above mentioned three datasets using 100 random origin concepts in the DL $\mathcal{ALCIQ}$. For each random origin concept C, we used $E^+ = \{a \mid a^{\mathcal{I}} \in C^{\mathcal{I}}\}$ as the set of positive examples and $E^- = \{a \mid a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}\}$ as the set of negative examples, where $\mathcal{I}$ is the considered interpretation used as the training information system. These concepts have different depths and lengths. We ran the program on each dataset and each concept in two cases: using only simple selectors and using both simple selectors and extended selectors. Table 1 summarizes the evaluation of our experiments in:

- the average (**Avg.**) modal depth (**Dep.**) of the origin concepts (**Org.**),
- the average length (**Len.**) of the origin concepts,
- the average modal depth of the resulting concepts (**Res.**),
- the average length of the resulting concepts,
- the average accuracy (**Acc.**), precision (**Pre.**), recall (**Rec.**) and F1 measures,
- the standard variant, minimum (**Min**) and maximum (**Max**) values of accuracy, precision, recall and F1 measures.

As can be seen in Table 1, the accuracy, precision, recall and F1 measures of the resulting concepts in classifying new objects are usually very high. This demonstrates that the bisimulation-based concept learning method is valuable.

In addition, we tested the method using specific concepts on the Family and PokerHand datasets. For the former, we use the following five popular concepts in the DL $\mathcal{ALCI}$:

1. $Grandparent \equiv \exists hasChild.(\exists hasChild.\top)$,
2. $Grandfather \equiv Male \sqcap \exists hasChild.(\exists hasChild.\top)$,
3. $Grandmother \equiv Female \sqcap \exists hasChild.(\exists hasChild.\top)$,
4. $Nephew \equiv Male \sqcap \exists hasChild^{-}.(\exists hasBrother.\top \sqcup \exists hasSister.\top)$,

Table 3 Evaluation results on the PokerHand dataset using six sets of objects in the DL $\mathcal{ALCQ}$

	Dep. Res.	Len. Res.	Avg. Acc. [min; max]	Avg. Pre. [min; max]	Avg. Rec. [min; max]	Avg. F1 [min; max]
One pair						
Simple selectors	4.0	109.00	42.57 ± 01.48	16.74 ± 00.87	76.00 ± 4.03	27.44 ± 01.42
			[40.71; 45.24]	[15.64; 18.05]	[71.67; 81.67]	[25.67; 29.45]
Simple and extended selectors	5.00	15.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Two pairs						
Simple selectors	4.00	25.00	36.33 ± 00.47	17.16 ± 0.53	90.33 ± 4.14	28.83 ± 00.96
			[35.48; 36.67]	[16.34; 17.70]	[83.33; 95.00]	[27.32; 29.84]
Simple and extended selectors	5.00	15.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Three of a kind						
Simple selectors	4.00	48.00	52.52 ± 02.16	20.92 ± 1.01	83.33 ± 01.83	33.43 ± 01.39
			[50.71; 56.67]	[19.75; 22.77]	[80.00; 85.00]	[31.68; 35.92]
Simple and extended selectors	3.00	11.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Straight						
Simple selectors	5.00	97.00	81.24 ± 02.01	39.65 ± 04.62	58.33 ± 04.94	47.13 ± 4.41
			[80.00; 85.24]	[36.36; 48.72]	[53.33; 65.00]	[43.24; 55.07]
Simple and extended selectors	5.00	32.00	98.67 ± 00.68	96.35 ± 03.44	94.33 ± 02.00	95.31 ± 02.35
			[97.62; 99.52]	[90.32; 100.0]	[91.67; 96.67]	[91.80; 98.31]
Flush						
Simple selectors	2.00	10.00	94.33 ± 00.80	71.71 ± 02.79	100.0 ± 00.00	83.49 ± 01.92
			[92.86; 95.24]	[66.67; 75.00]	[100.0; 100.0]	[80.00; 85.71]
Simple and extended selectors	3.00	7.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]
Full house						
Simple selectors	4.00	68.00	60.48 ± 03.05	25.95 ± 01.45	94.67 ± 2.45	40.71 ± 01.73
			[57.62; 64.76]	[24.23; 28.00]	[91.67; 98.33]	[38.33; 43.08]
Simple and extended selectors	2.00	6.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00	100.0 ± 00.00
			[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]	[100.0; 100.0]

5. $Niece \equiv Female \sqcap \exists hasChild^{-}.(\exists hasBrother.\top \sqcup \exists hasSister.\top)$.

For the PokerHand dataset, we tested the method using six sets of objects corresponding to six concepts (classes) in the DL $\mathcal{ALCQ}$. They are described below:

1. "*one pair*": one pair of equal ranks within five cards,
2. "*two pairs*": two pairs of equal ranks within five cards,
3. "*three of a kind*": three equal ranks within five cards,
4. "*straight*": five cards, sequentially ranked with no gaps,
5. "*flush*": five cards with the same suit,
6. "*full house*": pair + different rank three of a kind.

Table 2 provides the evaluation results on the Family dataset using the mentioned popular concepts.

Table 3 provides the evaluation results on the PokerHand dataset using the above six classes.

From Tables 1, 2 and 3, it is clear that extended selectors are highly effective for reducing the length of the resulting concepts and for obtaining better classifiers. This demonstrates that extended selectors efficiently support the bisimulation-based concept learning method.

8 Conclusions

We have generalized and extended the bisimulation-based concept learning method [22] for DL-based information systems. As demonstrated by Examples 1, 5 and 6, by taking attributes as basic elements of the language, our approach is more general and much more suitable for practical DL-based

information systems than the one of [22]. In comparison with [22], we allow also data roles and the concept constructors "functionality" and "unqualified number restrictions". Furthermore, we have formulated and proved an important theorem on basic selectors.

Apart from simple selectors, we introduced and used extended selectors for our method. We used information gain to choose blocks and selectors for granulating partitions. We also proposed a normal form which is used to represent concepts in a unified way. By Example 7, we showed that using both simple selectors and extended selectors is more effective than using only simple selectors. We have also implemented our method for a number of DLs.

We tested the method using simple selectors and extended selectors for different datasets. Our experimental results show that the method is valuable and extended selectors support it significantly.

Acknowledgments This work was supported by Polish National Science Centre (NCN) under Grant No. 2011/01/B/ST6/02759 as well as by Hue University under Grant No. DHH2013-01-41.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Reading, MA (1995)
2. Alvarez, J.: A formal framework for theory learning using description logics. In: ILP Work-in-progress reports, vol. 35. CEUR-WS.org (2000)
3. Baader, F., Nutt, W.: Basic description logics. In: Description Logic Handbook, pp. 43–95. Cambridge University Press (2003)
4. Badea, L., Nienhuys-Cheng, S.H.: A refinement operator for description logics. In: Proceedings of the 10th International Conference on Inductive Logic Programming, ILP'2000, pp. 40–59. Springer, New York (2000)
5. Cattral, R., Oppacher, F., Deugo, D.: Evolutionary data mining with automatic rule generalization (2002)
6. Cohen, W.W., Hirsh, H.: Learning the classic description logic: theoretical and experimental results. In: Proceedings of KR'1994, pp. 121–133 (1994)
7. Distel, F.: Learning description logic knowledge bases from data using methods from formal concept analysis. PhD thesis, Dresden University of Technology (2011)
8. Divroodi, A., Ha, Q.-T., Nguyen, L. A., Nguyen, H.S.: On C-learnability in description logics. In: Proceedings of ICCCI'2012 (1), vol. 7653 of LNCS, pp. 230–238. Springer (2012)
9. Divroodi, A.R., Nguyen, L.A.:. On bisimulations for description logics. CoRR, abs/1104.1964 (2011)
10. Divroodi, A.R., Nguyen, L.A.: On bisimulations for description logics. Inf. Sci. **295**, 465–493 (2015)
11. Fanizzi, N., d'Amato, C., Esposito, F.: DL-FOIL concept learning in description logics. In: Proceedings of ILP'2008, LNCS, pp. 107–121. Springer (2008)
12. Frazier, M., Pitt, L.: Classic learning. Mach. Learn. **25**(2–3), 151–193 (1996)
13. Ha, Q.-T., Hoang, T.-L.-G., Nguyen, L.A., Nguyen, H.S., Szałas, A., Tran, T.-L.: A bisimulation-based method of concept learning for knowledge bases in description logics. In: Proceedings of SoICT'2012, pp. 241–249. ACM (2012)
14. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\updownarrow \mathcal{SROIQ}$. In: KR, pp. 57–67. AAAI Press (2006)
15. Iannone, L., Palmisano, I., Fanizzi, N.: An algorithm based on counterfactuals for concept learning in the semantic web. Appl. Intell. **26**(2), 139–159 (2007)
16. Kietz, J.: Learnability of description logic programs. In: Proceedings of ILP'2002, vol. 2583 of LNCS, pp. 117–132. Springer (2002)
17. Konstantopoulos, S., Charalambidis, A.: Formulating description logic learning as an inductive logic programming task. In: Proceedings of FUZZ-IEEE, pp. 1–7 (2010)
18. Lambrix, P., Larocchia, P.: Learning composite concepts. In: Proceedings of DL'1998 (1998)
19. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Mach. Learn. **78**(1–2), 203–250 (2010)
20. Luna, J., Revoredo, K., Cozman, F.: Learning probabilistic description logics: a framework and algorithms. In: Proceedings of MICAI'2011, vol. 7094 of LNCS, pp. 28–39. Springer (2011)
21. Nguyen, L.A.: An efficient tableau prover using global caching for the description logic $\mathcal{ALC}$. Fundam. Inform. **93**(1–3), 273–288 (2009)
22. Nguyen, L.A., Szałas, A.: Logic-based roughification. In: Rough Sets and Intelligent Systems, vol. 1, pp. 517–543. Springer (2013)
23. Pawlak, Z.: Rough sets: theoretical aspects of reasoning about data. Kluwer Academic Publishers (1992)
24. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Inf. Sci. **177**(1), 3–27 (2007)
25. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
26. Schild, K.: A correspondence theory for terminological logics: preliminary report. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence, IJCAI'91, vol. 1, pp. 466–471. Morgan Kaufmann Publishers Inc. (1991)
27. Schmidt-Schaubß, M., Smolka, G.: Attributive concept descriptions with complements. Artif. Intell. **48**(1), 1–26 (1991)
28. Sen, P., Namata, G.M., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Mag. **29**(3), 93–106 (2008)
29. Tran, T.-L., Ha, Q.-T., Hoang, T.-L.-G., Nguyen, L.A., Nguyen, H.S.: Bisimulation-based concept learning in description logics. Fundam. Inform. **133**(2–3), 287–303 (2014)
30. Tran, T.-L., Ha, Q.-T., Hoang, T.-L.-G., Nguyen, L.A., Nguyen, H.S., Szałas, A.: Concept learning for description logic-based information systems. In: Proceedings of KSE'2012, pp. 65–73. IEEE Computer Society (2012)
31. Tran, T.-L., Nguyen, L.A., Hoang, T.-L.-G.: A domain partitioning method for bisimulation-based concept learning in description logics. In: Proceedings of ICCSAMA'2014, vol. 282 of Advances in Intelligent Systems and Computing, pp. 297–312. Springer (2014)
32. van Benthem, J.: Modal Logic and Classical Logic. Bibliopolis, Indices. Monographs in philosophical logic and formal linguistics (1983)
33. van Benthem, J.: Correspondence theory. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic. Synthese Library, vol. 165, pp. 167–247. Springer, Netherlands (1984)