A Domain Partitioning Method for Bisimulation-Based Concept Learning in Description Logics

Thanh-Luong Tran[1], Linh Anh Nguyen[2,3], and Thi-Lan-Giao Hoang[1]

[1] Faculty of Information Technology, Hue University of Sciences
77 Nguyen Hue, Hue city, Vietnam
{ttluong,hlgiao}@hueuni.edu.vn
[2] Institute of Informatics, University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl
[3] Faculty of Information Technology, VNU University of Engineering and Technology
144 Xuan Thuy, Hanoi, Vietnam

Abstract. We have implemented a bisimulation-based concept learning method for description logics-based information systems using information gain. In this paper, we present our domain partitioning method that was used for the implementation. Apart from basic selectors, we also used a new kind of selectors, called extended selectors. Our evaluation results show that the concept learning method is valuable and extended selectors support it significantly.

Keywords: description logic, concept learning, bisimulation.

1 Introduction

Semantic Technologies have been investigated intensively 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) [8]. Constructing useful ontologies is desirable. 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. Concept learning in DLs has been studied by a number of researchers. Apart from the approaches based on "least common subsumers" proposed by Cohen and Hrish [4], and concept normalization

T.V. Do et al. (eds.), *Advanced Computational Methods for Knowledge Engineering,*
Advances in Intelligent Systems and Computing 282,
DOI: 10.1007/978-3-319-06569-4_22,

proposed by Lambrix and Larocchia [10], concept learning was also studied using the approach of inductive logic programming and refinement operators by Badea and Nienhuy-Cheng [2], Iannone et al. [9], Fanizzi et al. [6], and Lehmann et al. [11]. Recently, Nguyen and Szałas [13], Ha et al. [7], and Tran et al. [15,16] used bisimulation for concept learning in DLs.

One of the main settings for concept learning in DLs [7,15] is as follows: given a finite interpretation $\mathcal{I}$ in a DL L, learn a concept C in L such that

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

where E^+ (resp. E^-) contains positive (resp. negative) examples of C. Note that $\mathcal{I} \not\models C(a)$ is the same as $\mathcal{I} \models \neg C(a)$.

In [13], Nguyen and Szałas divide blocks in partitions of DL-based information systems by using basic selectors. They applied bisimulation in DLs to model indiscernibility of objects. Their work is pioneering in using bisimulation for concept learning in DLs. In [16], Tran et al. generalized and extended the concept learning method of [13] for DL-based information systems. The important problem is: which block from the current partition should be divided first and which selector should be used to divide it? These affect both the "quality" of the final partition and the complexity of the process. Nguyen and Szałas [13] and Tran et al. [16] proposed to use basic selectors and information gain to guide the granulation process, where basic selectors are created from blocks of the partitions appeared in the granulation process. These selectors divide blocks in the partitions and bring good results in favorable cases. However, they are not strong 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.

In this paper, apart from the so called *basic selectors* that are created from the blocks of partitions, we also propose to use a new kind of selectors, called *extended selectors*, which are created from available selectors (basic and extended selectors) for dividing blocks. We have implemented the bisimulation-based concept learning method [13,16] using the mentioned selectors and information gain measures. The aim is to study effectiveness of basic and extended selectors as well as to provide experimental results for the concept learning method.

The rest of paper is structured as follows. In Section 2, we recall the notation of DLs and the definition of DL-based information systems. Section 3 outlines bisimulation and indiscernibility in DLs. Section 4 presents concept normalization and storage. Basic and extended selectors, information gain measures as well as techniques of concept reduction are presented in Section 5. Our experimental results are reported in Section 6. We conclude this work in Section 7.

2 Notation and DL-Based Information Systems

A *DL-signature* is a finite set $\Sigma = \Sigma_I \cup \Sigma_C \cup \Sigma_R$, where Σ_I is a set of *individuals*, Σ_C is a set of *concept names*, and Σ_R is a set of *role names*. All the sets Σ_I,

$(r^-)^{\mathcal{I}} = (r^{\mathcal{I}})^{-1}$ $\quad \top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ $\quad \bot^{\mathcal{I}} = \emptyset$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\,[R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)]\}$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\,[R^{\mathcal{I}}(x,y) \wedge C^{\mathcal{I}}(y)]\}$	$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$(\geq n\,R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \mid R^{\mathcal{I}}(x,y) \wedge C^{\mathcal{I}}(y)\} \geq n\}$	$(\geq n\,R)^{\mathcal{I}} = (\geq n\,R.\top)^{\mathcal{I}}$
$(\leq n\,R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \mid R^{\mathcal{I}}(x,y) \wedge C^{\mathcal{I}}(y)\} \leq n\}$	$(\leq n\,R)^{\mathcal{I}} = (\leq n\,R.\top)^{\mathcal{I}}$

Fig. 1. Semantics of complex object roles and complex concepts

Σ_C and Σ_R are pairwise disjoint. We denote individuals by letters like a and b, concept names by letters like A and B, role names by letters like r and s.

We consider some *DL-features* denoted by I (*inverse*), F (*functionality*), N (*unquantified number restriction*), Q (*quantified number restriction*). A *set of DL-features* is a set consisting of some or zero of these names.

Let Σ be a DL-signature and Φ be a set of DL-features. Let $\mathcal{L}$ stand for $\mathcal{ALC}$, which is the name of a basic DL. The DL language $\mathcal{L}_{\Sigma,\Phi}$ allows *roles* and *concepts* defined recursively as follows:

- if $r \in \Sigma_R$ then r is a role of $\mathcal{L}_{\Sigma,\Phi}$,
- if $I \in \Phi$ and $r \in \Sigma_R$ then r^- is a role of $\mathcal{L}_{\Sigma,\Phi}$,
- if $A \in \Sigma_C$ then A is concept of $\mathcal{L}_{\Sigma,\Phi}$,
- if C and D are concepts of $\mathcal{L}_{\Sigma,\Phi}$, R is a role of $\mathcal{L}_{\Sigma,\Phi}$ and n is a natural number then
  - $\top$, $\bot$, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\forall R.C$ and $\exists R.C$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$,
  - if $F \in \Phi$ 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 $Q \in \Phi$ then $\geq n\,R.C$ and $\leq n\,R.C$ are concepts of $\mathcal{L}_{\Sigma,\Phi}$.

An *interpretation* in $\mathcal{L}_{\Sigma,\Phi}$ is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, 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 role name $r \in \Sigma_R$ with a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

To simplicity of notation, we write $C^{\mathcal{I}}(x)$ (resp. $R^{\mathcal{I}}(x,y)$) instead of $x \in C^{\mathcal{I}}$ (resp. $\langle x,y \rangle \in R^{\mathcal{I}}$). The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex concepts and complex roles as shown in Figure 1.

An *information system* in $\mathcal{L}_{\Sigma,\Phi}$ is a finite interpretation in $\mathcal{L}_{\Sigma,\Phi}$.

Example 2.1. Let $\Phi = \{I\}$ and $\Sigma = \Sigma_I \cup \Sigma_C \cup \Sigma_R$, where
$\Sigma_I = \{Ava, Britt, Colin, Dave, Ella, Flor, Gigi, Harry\}$,
$\Sigma_C = \{Male, Female, Nephew, Niece\}$ and $\Sigma_R = \{hasChild, hasSibling\}$.
Consider the information system $\mathcal{I}$ specified by:
$\Delta^{\mathcal{I}} = \{a,b,c,d,e,f,g,h\}$, $Ava^{\mathcal{I}} = a$, $Britt^{\mathcal{I}} = b$, …, $Harry^{\mathcal{I}} = h$,
$Female^{\mathcal{I}} = \{a,b,e,f,g\}$, $Male^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus Female^{\mathcal{I}} = \{c,d,h\}$,
$hasChild^{\mathcal{I}} = \{\langle a,d \rangle, \langle a,e \rangle, \langle b,f \rangle, \langle c,g \rangle, \langle c,h \rangle\}$,
$hasSibling^{\mathcal{I}} = \{\langle b,c \rangle, \langle c,b \rangle, \langle d,e \rangle, \langle e,d \rangle, \langle g,h \rangle, \langle h,g \rangle\}$,
$Niece^{\mathcal{I}} = (Female \sqcap \exists hasChild^-.(\exists hasSibling.\top))^{\mathcal{I}} = \{f,g\}$,

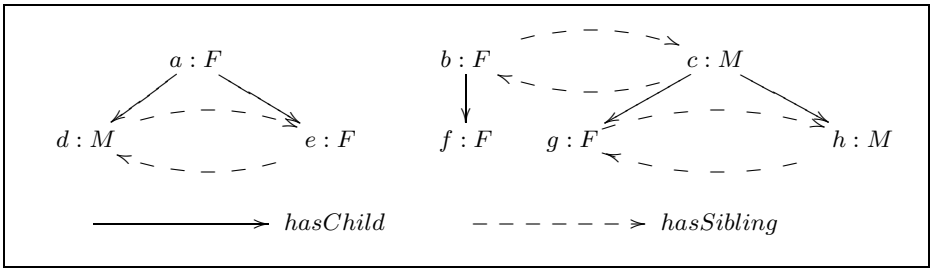

Fig. 2. An illustration for the information system given in Example 2.1

$Nephew^{\mathcal{I}} = (Male \sqcap \exists hasChild^{-}.(\exists hasSibling.\top))^{\mathcal{I}} = \{h\}$.

This interpretation is illustrated in Figure 2. In this figure, each node denotes a person, the letter M stands for $Male$, the letter F stands for $Female$, the solid edges denote assertions of the role $hasChild$, and the dashed edges denote assertions of the role $hasSibling$. ∎

Let C and D be concepts of $\mathcal{L}_{\Sigma,\Phi}$. We say that:

- C is *subsumed* by D, denoted by $C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every interpretation $\mathcal{I}$ in $\mathcal{L}_{\Sigma,\Phi}$,
- C and D are *equivalent*, denoted by $C \equiv D$, if $C \sqsubseteq D$ and $D \sqsubseteq C$,
- C is *satisfiable* if there exists an interpretation $\mathcal{I}$ in $\mathcal{L}_{\Sigma,\Phi}$ such that $C^{\mathcal{I}} \neq \emptyset$.

The *modal depth* of a concept C, denoted by $\mathsf{mdepth}(C)$, is defined to be:

- 0 if C is of the form $\top$, $\bot$ or A,
- $\mathsf{mdepth}(D)$ if C is of the form $\neg D$,
- 1 if C is of the form $\geq n\, R$ 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 $\leq n\, R.D$,
- $\max(\mathsf{mdepth}(D), \mathsf{mdepth}(D'))$ if C is of the form $D \sqcap D'$ or $D \sqcup D'$.

The *length* of a concept C, denoted by $\mathsf{length}(C)$, is defined to be:

- 0 if C is $\top$ or $\bot$,
- 1 if C is a concept name A,
- $\mathsf{length}(D)$ if C is the form $\neg D$,
- 3 if C is of the form $\geq n\, R$ 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 $\leq n\, R.D$,
- $1 + \mathsf{length}(D) + \mathsf{length}(D')$ if C is of the form $D \sqcap D'$ or $D \sqcup D'$.

In this paper, concepts are represented in the negation 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 2.2. Consider the language $\mathcal{L}_{\Sigma,\Phi}$ given in Example 1, we have:

- $\mathsf{mdepth}(Male \sqcap \geq 2\, hasChild.(\exists hasChild.\top)) = 2$,
- $\mathsf{length}(Male \sqcap \geq 2\, hasChild.(\exists hasChild.\top)) = 7$. ∎

We say that a concept C is *simpler* than a concept 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. This leads to small differences of the modal depth between concepts. In contrast, differences of the length between concepts may be very large. Thus, we choose the length of concepts 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 $\mathbb{C}$.

3 Bisimulation and Indiscernibility

In this section, we recall the notion of bisimulation for the DLs considered in this paper [13,16]. Let Σ and $\Sigma^\dagger$ be DL-signatures such that $\Sigma^\dagger \subseteq \Sigma$, Φ and $\Phi^\dagger$ be DL-features such that $\Phi^\dagger \subseteq \Phi$, $\mathcal{I}$ and $\mathcal{I}'$ be interpretations in $\mathcal{L}_{\Sigma,\Phi}$. An $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*bisimulation* between $\mathcal{I}$ and $\mathcal{I}'$ is a binary relation $Z \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}'}$ satisfies the following conditions for every $a \in \Sigma_I^\dagger$, $A \in \Sigma_C^\dagger$, $r \in \Sigma_R^\dagger$, $x, y \in \Delta^{\mathcal{I}}$, $x', y' \in \Delta^{\mathcal{I}'}$:

$$Z(a^{\mathcal{I}}, a^{\mathcal{I}'}) \tag{1}$$

$$Z(x, x') \Rightarrow [A^{\mathcal{I}}(x) \Leftrightarrow A^{\mathcal{I}'}(x')] \tag{2}$$

$$[Z(x, x') \wedge r^{\mathcal{I}}(x, y)] \Rightarrow \exists y' \in \Delta^{\mathcal{I}'}[Z(y, y') \wedge r^{\mathcal{I}'}(x', y')] \tag{3}$$

$$[Z(x, x') \wedge r^{\mathcal{I}'}(x', y')] \Rightarrow \exists y \in \Delta^{\mathcal{I}}[Z(y, y') \wedge r^{\mathcal{I}}(x, y)], \tag{4}$$

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

$$[Z(x, x') \wedge r^{\mathcal{I}}(y, x)] \Rightarrow \exists y' \in \Delta^{\mathcal{I}'}[Z(y, y') \wedge r^{\mathcal{I}'}(y', x')] \tag{5}$$

$$[Z(x, x') \wedge r^{\mathcal{I}'}(y', x')] \Rightarrow \exists y \in \Delta^{\mathcal{I}}[Z(y, y') \wedge r^{\mathcal{I}}(y, x)], \tag{6}$$

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

$$Z(x, x') \Rightarrow \#\{y \mid r^{\mathcal{I}}(x, y)\} = \#\{y' \mid r^{\mathcal{I}'}(x', y')\}, \tag{7}$$

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

$$Z(x, x') \Rightarrow \#\{y \mid r^{\mathcal{I}}(y, x)\} = \#\{y' \mid r^{\mathcal{I}'}(y', x')\}, \tag{8}$$

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

$$Z(x, x') \Rightarrow [\#\{y \mid r^{\mathcal{I}}(x, y)\} \leq 1 \Leftrightarrow \#\{y' \mid r^{\mathcal{I}'}(x', y')\} \leq 1], \tag{9}$$

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

$$Z(x, x') \Rightarrow [\#\{y \mid r^{\mathcal{I}}(y, x)\} \leq 1 \Leftrightarrow \#\{y' \mid r^{\mathcal{I}'}(y', x')\} \leq 1], \tag{10}$$

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

$$\begin{array}{l}\text{if } Z(x,x') \text{ holds then, for every } r \in \Sigma_R^\dagger, \text{ there exists a bijection} \\ h : \{y \mid r^{\mathcal{I}}(x,y)\} \to \{y' \mid r^{\mathcal{I}'}(x',y')\} \text{ such that } h \subseteq Z,\end{array} \tag{11}$$

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

$$\begin{array}{l}\text{if } Z(x,x') \text{ holds then, for every } r \in \Sigma_R^\dagger, \text{ there exists a bijection} \\ h : \{y \mid r^{\mathcal{I}}(y,x)\} \to \{y' \mid r^{\mathcal{I}'}(y',x')\} \text{ such that } h \subseteq Z.\end{array} \tag{12}$$

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

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

An $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-bisimulation between $\mathcal{I}$ and itself is called an $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-*auto-bisimulation* of $\mathcal{I}$. The *largest* $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$, denoted by $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$, is the one that is larger than or equal to ($\supseteq$) any other $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$-auto-bisimulation of $\mathcal{I}$. It always exists [5].

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

Theorem 3.1. *Let $\mathcal{I}$ be an interpretation in $\mathcal{L}_{\Sigma,\Phi}$ and let $X \subseteq \Delta^{\mathcal{I}}$, $\Sigma^\dagger \subseteq \Sigma$, $\Phi^\dagger \subseteq \Phi$. Then:*

- *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,*
- *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 [13,16,7] only in the studied class of DLs. It can be proved analogously to [13, Theorem 4].

4 Concept Normalization and Storage

There are different normal forms for concepts [1,12]. By using a normal form, one can represent concepts in a unified way. We propose below a new normal form, which extends the one of [12]. 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. $\geq 0\, R.C$ is replaced by $\top$,
13. $\geq 1\, R.C$ is replaced by $\exists R.C$,
14. $\geq n\, R.\bot$ is replaced by $\bot$ when $n > 0$,
15. the rules "dual" to the rules 2–10 (for example, dually to the fourth rule, $\sqcup\{\sqcup\{C_1, C_2, \ldots, C_i\}, C_{i+1}, \ldots, C_n\}$ is replaced by $\sqcup\{C_1, C_2, \ldots, C_n\}$).

Each step of the granulation 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. In addition, our program processes only one interpretation (the considered information system) and many concepts may have the same extension (i.e., the same set of objects in the interpretation). Thus, instead of storing all concepts which have the same extension, only the simplest concept is archived in the catalogue. These techniques allow to reduce the memory for representing concepts and their extensions as well as to increase the performance of our program.

5 Granulating Partitions Using Selectors and Information Gain

Let $\mathcal{I}$ be a finite interpretation in $\mathcal{L}_{\Sigma,\Phi}$ given as a training information 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 $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$, 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 condition:

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

In [13] Nguyen and Szałas proposed a bisimulation-based method for this learning problem, and in [16] Tran et al. extended the method for a large class of DLs. The idea of those works is based on the following observation:

> if A_d is definable in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ by a concept C then, by the first assertion of Theorem 3.1, $C^{\mathcal{I}}$ must be the union of some equivalence classes of $\Delta^{\mathcal{I}}$ w.r.t. $\sim_{\Sigma^\dagger,\Phi^\dagger,\mathcal{I}}$.

1. A, where $A \in \Sigma_C^\dagger$,
2. $\exists R.\top$, $\forall R.\bot$,
3. $\exists R.A$ and $\forall R.A$, where $A \in \Sigma_C^\dagger$,
4. $\exists R.C_i$ and $\forall R.C_i$, where $1 \leq i \leq n$,
5. $\leq 1\, R$ if $F \in \Phi^\dagger$,
6. $\leq l\, R$ and $\geq m\, R$, if $N \in \Phi^\dagger$, $0 < l \leq \#\Delta^\mathcal{I}$ and $0 \leq m < \#\Delta^\mathcal{I}$,
7. $\leq l\, R.C_i$ and $\geq m\, R.C_i$, if $Q \in \Phi^\dagger$, $1 \leq i \leq n$, $0 < l \leq \#C_i^\mathcal{I}$ and $0 \leq m < \#C_i^\mathcal{I}$,

where R is a role of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$.

Fig. 3. Basic selectors

1. $\exists R.D_u$ and $\forall R.D_u$,
2. $\leq l\, R.D_u$ and $\geq m\, R.D_u$, if $Q \in \Phi^\dagger$, $0 < l \leq \#D_u^\mathcal{I}$ and $0 \leq m < \#D_u^\mathcal{I}$,

where R is a role of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ and $1 \leq u \leq h$.

Fig. 4. Extended selectors

5.1 Basic Selectors and Extended Selectors

In the granulation process, we denote the blocks created so far in all steps by $Y_1, Y_2, \ldots, Y_n$. We always use a new subscript for each newly created block by increasing n. We take care that, for each $1 \leq i \leq n$, Y_i is characterized by a concept C_i such that $C_i^\mathcal{I} = Y_i$.

Let $\mathbb{Y} = \{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\} \subseteq \{Y_1, Y_2, \ldots, Y_n\}$ be the current partition of $\Delta^\mathcal{I}$ and $\mathbb{D} = \{D_1, D_2, \ldots, D_h\}$ be the current set of selectors, which are concepts of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$. (At the beginning, $\mathbb{Y} = \{\Delta^\mathcal{I}\}$ and $\mathbb{D} = \emptyset$.) Consider dividing a block $Y_{i_j} \in \mathbb{Y}$. We want to find a selector $D_u \in \mathbb{D}$ to divide Y_{i_j}.

Such a selector D_u should actually divide Y_{i_j} into two non-empty parts (i.e., Y_{i_j} should be divided by $D_u^\mathcal{I}$). We divide Y_{i_j} by D_u as follows:

- $s := n+1, t := n+2, n := n+2$,
- $Y_s := Y_{i_j} \cap D_u^\mathcal{I}, C_s := C_{i_j} \sqcap D_u$,
- $Y_t := Y_{i_j} \cap (\neg D_u)^\mathcal{I}, C_t := C_{i_j} \sqcap \neg D_u$,
- the new partition of $\Delta^\mathcal{I}$ is $\{Y_{i_1}, Y_{i_2}, \ldots, Y_{i_k}\} \cup \{Y_s, Y_t\} \setminus \{Y_{i_j}\}$.

In [16], the used selectors are concepts of $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ of the forms listed in Figure 3, which were called *basic selectors*. The work [16] showed that, 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 by using only basic selectors.

We have implemented the bisimulation-based concept learning method using basic selectors and information gain [12,16]. Our experiments showed that the resulting concepts obtained by this method 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 basic selectors are not advanced enough. In our implementation, we used a new kind of selectors, called *extended selectors*. They are created from the current set $\mathbb{D}$ of selectors. Each extended selector is a concept of one of the forms listed in Figure 4. Extended selectors play an important role in the granulation process. By using them, we have more selectors that can be used to divide blocks and we can obtain better partitions than by using only basic selectors. Furthermore, by using extended selectors, the number of iterations of the main loop is usually reduced significantly. This leads to simpler resulting concepts with higher accuracy in classifying new objects.

5.2 Information Gain for Granulating Partitions

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. The *entropy* of Y w.r.t. X in $\Delta^{\mathcal{I}}$, denoted by $E_{\Delta^{\mathcal{I}}}(Y/X)$, is defined as follows, where XY stands for $X \cap Y$ and $\overline{X}Y$ stands for $\overline{X} \cap Y$:

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

The *information gain* of a selector D for dividing Y w.r.t. X in $\Delta^{\mathcal{I}}$, denoted by $IG_{\Delta^{\mathcal{I}}}(Y/X, D)$, is defined as follows, where $D^{\mathcal{I}}Y$ stands for $D^{\mathcal{I}} \cap Y$ and $\overline{D^{\mathcal{I}}}Y$ stands for $\overline{D^{\mathcal{I}}} \cap Y$:

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

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

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 $\arg\max_{D_u \in \mathbb{D}}\{IG(Y_{i_j}, D_u)\}$. For the current partition $\mathbb{Y}$, if Y_{i_j} is chosen to be divided then S_{i_j} is the choice for dividing 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 divided first. We choose a block Y_{i_j} such that applying the selector S_{i_j} to divide Y_{i_j} maximizes the information gain. That is, we divide a block $Y_{i_j} \in \arg\max_{Y_{i_j} \in \mathbb{Y}}\{IG(Y_{i_j}, S_{i_j})\}$ first.

After dividing a block, we have a new partition. We also add new selectors which are created by the rules in Figures 3 and 4 to the current set of selectors. This set is used to continue granulating the new partition.

Example 5.1. Consider the information system $\mathcal{I}$ given in Example 2.1, the sublanguage $\mathcal{L}_{\Sigma^\dagger, \Phi^\dagger}$ with $\Sigma^\dagger = \{Female, hasChild, hasSibling\}$ and $\Phi^\dagger = \{I\}$, and the set $X = \{f, g\}$. One can think of X as the set of instances of the concept $Niece = Female \sqcap \exists hasChild^-.(\exists hasSibling.\top)$ in $\mathcal{I}$.

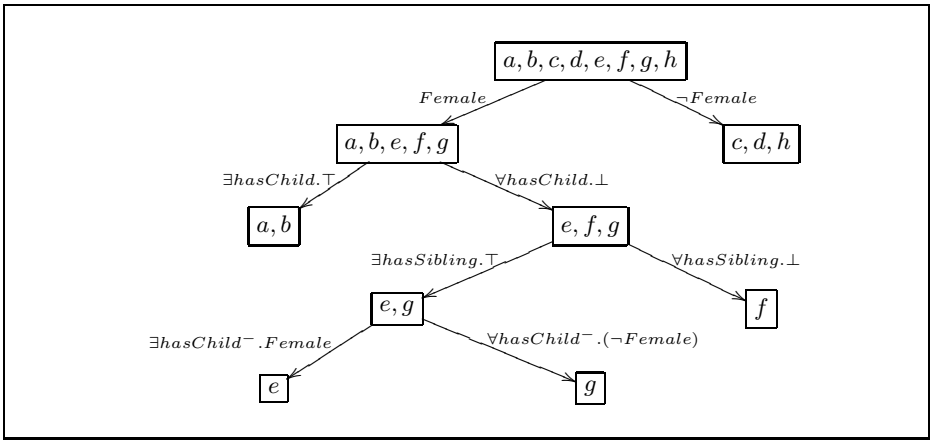

Fig. 5. A tree illustrating the granulation process using only basic selectors

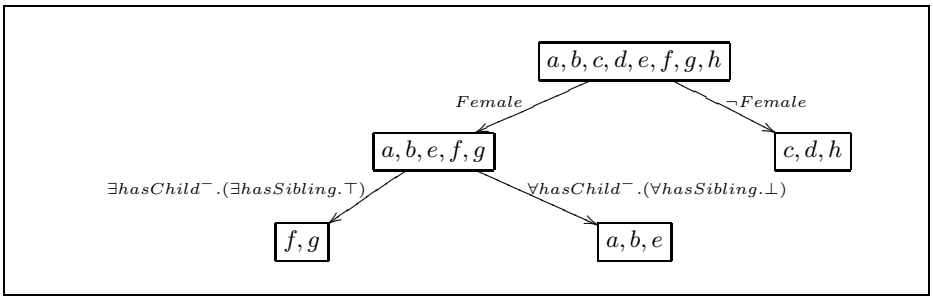

Fig. 6. A tree illustrating the granulation process using basic and extended selectors

1. Learning a definition of X in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ using only basic selectors:
   The steps are illustrated by the tree in Figure 5. 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. Learning a definition of X in $\mathcal{L}_{\Sigma^\dagger,\Phi^\dagger}$ using basic and extended selectors:
   The steps are illustrated by the tree in Figure 6. The resulting concept is

   $$Female \sqcap \exists hasChild^-.(\exists hasSibling.\top).$$

In the second case, the concept $\exists hasChild^-.(\exists hasSibling.\top)$ is an extended selector. It is created by applying the rule 1 in Figure 4 to $\exists hasSibling.\top$, which is one of the available selectors.

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

5.3 Reducing the Resulting Concepts

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 our implementation, we use 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.

- The first technique is pruning. Given a decision tree generated by the granulation process, the technique 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. In our implementation, we use the bottom up fashion: we repeatedly choose a node whose successors are leafs and cut 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 set of objects 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.

In our experiments, the above three techniques allow to reduce the length of the resulting concepts significantly.

6 Experimental Results

The difficulty we encountered is that there are too few available datasets with linked data that can directly be used for concept learning using our setting. We have to build/get datasets for our setting from some resources on the Internet, including the WebKB [14], PockerHand [3] 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

Table 1. Evaluation results on the WebKB, PockerHand 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						
Basic Selectors	0.82/1.02	6.81/4.41	93.84±13.50 [33.69;100.0]	92.09±17.04 [32.08;100.0]	92.82±17.32 [23.08;100.0]	91.59±16.68 [27.69;100.0]
Ba.&Ex. Selectors	0.84/1.02	3.40/4.41	94.60±12.20 [33.69;100.0]	92.81±15.93 [32.08;100.0]	93.14±17.17 [23.08;100.0]	92.33±16.17 [27.69;100.0]
PockerHand dataset						
Basic Selectors	1.41/2.60	37.02/15.97	97.17±08.61 [50.57;100.0]	95.96±14.99 [01.67;100.0]	94.95±14.40 [01.67;100.0]	94.66±14.64 [1.67;100.0]
Ba.&Ex. Selectors	1.23/2.60	3.47/15.97	99.44±02.15 [83.25;100.0]	98.68±09.08 [01.67;100.0]	98.06±09.58 [01.67;100.0]	98.18±09.14 [1.67;100.0]
Family dataset						
Basic Selectors	2.38/3.34	78.50/18.59	88.50±16.65 [27.91;100.0]	90.60±18.57 [04.55;100.0]	85.66±22.36 [07.69;100.0]	86.09±20.10 [08.70;100.0]
Ba.&Ex. Selectors	2.29/3.34	10.20/18.59	92.79±14.35 [27.91;100.0]	91.99±18.40 [04.55;100.0]	91.75 ±19.82 [07.69;100.0]	90.39±19.89 [08.70;100.0]

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 11062 links between them of seven relationships (`hasChild`, `hasSon`, `hasDaughter`, `hasWife`, `hasHusband`, `hasBrother`, `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 **PockerHand** dataset is a subset taken from UCI Machine Learning Repository. It consists of information about 2542 hands, 12710 cards, 119 features of cards (15371 objects in total) and 65220 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 "`royal flush`". Because the number of hands in the classes "`four of a kind`", "`straight flush`" and "`royal flush`" 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 (12685 objects).

Given a signature Σ, by $\mathcal{ALCIQ}$ (resp. $\mathcal{ALCI}$ or $\mathcal{ALCQ}$) we denote the logic $\mathcal{L}_{\Sigma,\Phi}$ with $\Phi = \{I, Q\}$ (resp. $\Phi = \{I\}$ or $\Phi = \{Q\}$).

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

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 basic and extended selectors as well as the information gain discussed in Section 5. The reduction techniques mentioned in that section 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 basic selectors and using

Table 3. Evaluation results on the Family dataset using six sets of objects and 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						
Basic Selectors	4.0	109.00	42.57±01.48 [40.71;45.24]	16.74±00.87 [15.64;18.05]	76.00±4.03 [71.67;81.67]	27.44±01.42 [25.67;29.45]
Ba.&Ex. Selectors	5.00	15.00	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]
Two pairs						
Basic Selectors	4.00	25.00	36.33±00.47 [35.48;36.67]	17.16±0.53 [16.34;17.70]	90.33±4.14 [83.33;95.00]	28.83±00.96 [27.32;29.84]
Ba.&Ex. Selectors	5.00	15.00	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]
Three of a kind						
Basic Selectors	4.00	48.00	52.52±02.16 [50.71;56.67]	20.92±1.01 [19.75;22.77]	83.33±01.83 [80.00;85.00]	33.43±01.39 [31.68;35.92]
Ba.&Ex. Selectors	3.00	11.00	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]
Straight						
Basic Selectors	5.00	97.00	81.24±02.01 [80.00;85.24]	39.65±04.62 [36.36;48.72]	58.33±04.94 [53.33;65.00]	47.13±4.41 [43.24;55.07]
Ba.&Ex. Selectors	5.00	32.00	98.67±00.68 [97.62;99.52]	96.35±03.44 [90.32;100.0]	94.33±02.00 [91.67;96.67]	95.31±02.35 [91.80;98.31]
Flush						
Basic Selectors	2.00	10.00	94.33±00.80 [92.86;95.24]	71.71±02.79 [66.67;75.00]	100.0±00.00 [100.0;100.0]	83.49±01.92 [80.00;85.71]
Ba.&Ex. Selectors	3.00	7.00	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]
Full house						
Basic Selectors	4.00	68.00	60.48±03.05 [57.62;64.76]	25.95±01.45 [24.23;28.00]	94.67±2.45 [91.67;98.33]	40.71±01.73 [38.33;43.08]
Ba.&Ex. Selectors	2.00	6.00	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]	100.0±00.00 [100.0;100.0]

both basic 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 PockerHand datasets. For the former, we use the following five popular concepts in the DL $\mathcal{ALCI}$:

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

For the PockerHand 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 PockerHand 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.

7 Conclusions

We have implemented the bisimulation-based concept learning method for description logics-based information systems [13,16]. Apart from basic selectors proposed in [13,16], we introduced and used extended selectors for this method. We tested the method using basic and extended selectors for different datasets. Our experimental results show that the method is valuable and extended selectors support it significantly.

Acknowledgments. This paper was written during the first author's visit at Warsaw Center of Mathematics and Computer Sciences (WCMCS). He would like to thank WCMCS for the support. We thank Prof. Hung Son Nguyen for allowing us to use a server to run our program. This work was also 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. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
2. Badea, L., Nienhuys-Cheng, S.-H.: A refinement operator for description logics. In: Cussens, J., Frisch, A.M. (eds.) ILP 2000. LNCS (LNAI), vol. 1866, pp. 40–59. Springer, Heidelberg (2000)
3. Cattral, R., Oppacher, F., Deugo, D.: Evolutionary data mining with automatic rule generalization (2002)
4. Cohen, W.W., Hirsh, H.: Learning the CLASSIC description logic: Theoretical and experimental results. In: Proceedings of KR 1994, pp. 121–133 (1994)
5. Divroodi, A., Nguyen, L.: On bisimulations for description logics. In: Proceedings of CS&P 2011, pp. 99–110 (2011) (see also arXiv:1104.1964)
6. Fanizzi, N., d'Amato, C., Esposito, F.: DL-FOIL concept learning in description logics. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 107–121. Springer, Heidelberg (2008)
7. 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 the Third Symposium on Information and Communication Technology, SoICT 2012, pp. 241–249. ACM (2012)
8. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: KR, pp. 57–67. AAAI Press (2006)
9. Iannone, L., Palmisano, I., Fanizzi, N.: An algorithm based on counterfactuals for concept learning in the semantic web. Applied Intelligence 26(2), 139–159 (2007)
10. Lambrix, P., Larocchia, P.: Learning composite concepts. In: Proceedings of DL 1998 (1998)
11. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Machine Learning 78(1-2), 203–250 (2010)
12. Nguyen, L.A.: An efficient tableau prover using global caching for the description logic $\mathcal{ALC}$. Fundam. Inform. 93(1-3), 273–288 (2009)
13. Nguyen, L.A., Szałas, A.: Logic-based roughification. In: Skowron, A., Suraj, Z. (eds.) Rough Sets and Intelligent Systems. ISRL, vol. 42, pp. 517–543. Springer, Heidelberg (2013)
14. Sen, P., Namata, G.M., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine 29(3), 93–106 (2008)
15. Tran, T.-L., Ha, Q.-T., Hoang, T.-L.-G., Nguyen, L.A., Nguyen, H.S.: Bisimulation-based concept learning in description logics. In: Proceedings of CS&P 2013, pp. 421–433. CEUR-WS.org (2013)
16. Tran, T.-L., Ha, Q.-T., Hoang, T.-L.-G., Nguyen, L.A., Nguyen, H.S., Szalas, A.: Concept learning for description logic-based information systems. In: Proceedings of the 2012 Fourth International Conference on Knowledge and Systems Engineering, KSE 2012, pp. 65–73. IEEE Computer Society (2012)