The direct-optimal basis via reductions

Estrella Rodríguez-Lorenzo[1], Karell Bertet[2], Pablo Cordero[1], Manuel Enciso[1], and Angel Mora[1]

[1] University of Málaga, Andalucía Tech, Spain,
e-mail: {estrellarodlor,amora}@ctima.uma.es, {pcordero,enciso}@uma.es
[2] Laboratoire 3I, Université de La Rochelle
e-mail: karell.bertet@univ-lr.fr

Abstract. Formal Concept Analysis has become a real approach in the trend Information-Knowledge-Wisdom. It turns around the mining of a data set to built a concept lattice which provides an strong structure of the knowledge. Implications play the role of an alternative specification of this concept lattice and may be managed by means of inference rules. This syntactic treatment is guided by several properties like directness, minimality, optimality, etc. In this work, we propose a method to calculate the direct-optimal basis equivalent to a given Implicational System. Our method deals with unitary and non-unitary implications. Moreover, it shows a better performance that previous methods in the literature by means of the use of Simplification Logic and reduction paradigm, which remains narrow implications in any stage of the process. We have also developed an empirical study to compare our method with previous approaches in the literature.

1 Introduction

Formal Concept Analysis (FCA) is a trending upward area which establishes a proper and fine mixture of formalism, data analysis and knowledge discovering. It is able to analyze and extract information from a context $\mathbf{K}$, rendering a concept lattice. Attribute implications [10] represent implicit knowledge between data and they can be deduced from the concept lattice or using mining techniques from the context directly. An attribute implication is an expression $A \to B$ where A and B are sets of attributes. A context satisfies $A \to B$ if every object that has all the attributes in A has also all the attributes in B.

The study of sets of implications that satisfies some criteria is one of the relevant topics in FCA. An implicational system (IS) of $\mathbf{K}$ is defined as a set Σ of implications of $\mathbf{K}$ from which any valid implication for $\mathbf{K}$ can be deduced by means a syntactic treatment of the implications. This symbolic manipulation introduces the notion of equivalent sets of implications and opens the door to the definition of several criteria to discriminate good sets of implications according to these criteria. Thus, the challenges are the definition of an specific notion of IS, named basis, fulfilling some criteria related with minimality and the introduction of efficient methods to transform an arbitrary IS into a basis.

For instance, if the criteria is to obtain an IS with minimum cardinal we can build the so-called Duquenne-Guigues (or stem) basis [11]. Each application may induces a different criterium. For instance, in [2, 3] some methods to calculate the direct-optimal basis are introduced, joining minimality and directness in the same notion of basis. In [8] a method to obtain a basis with minimal size in the left-hand size of the implications was proposed.

In this paper, we introduce a method to compute the direct-optimal basis. This kind of basis was introduced in [2,3] and it has two interesting properties: it has the minimum number of attributes and it provides a framework to efficiently compute the closure of a set of attributes. The new method introduced in this paper is strongly based on $\mathbf{SL}_{\text{FD}}$ (Simplification Logic) and they are more efficient than previous methods appeared in the literature.

In the following, first we establish the background necessary for the understanding of the paper (Section 2). In Section 3 $\mathbf{SL}_{\text{FD}}$ is summarized and a motivation of the simplification paradigm to remove redundant attributes is provided. Section 4 is focussed on the methods of Bertet et al. to get a direct-optimal basis. In Section 5 the new method is introduced and a comparison among all the methods is showed. Some conclusions are presented in Section 6.

2 Preliminaries

We assume well-known the main concepts in FCA [10]. Only the concepts necessaries will be introduced. In Formal Concept Analysis (FCA) the relationship between a set of objects and a set of attributes are described using a formal context as follows:

Definition 1. *A formal context is a triple $\mathbf{K} = (G, M, I)$ where G is a finite set whose elements are named objects, M is a finite set whose elements are named attributes and $I \subseteq G \times M$ is a binary relation. Thus, $(o, a) \in I$ means the object o has the attribute a.*

This paper focuses on the notion of implication, which can be introduced as follows:

Definition 2. *Let $\mathbf{K} = (G, M, I)$ be a formal context and $A, B \in 2^M$. The implication $A \to B$ holds in $\mathbf{K}$ if every object $o \in G$ satisfies the following: $(o, a) \in I$ for all $a \in A$ implies $(o, b) \in I$ for all $b \in B$.*

An implication $A \to B$ is said to be unitary if the set B is a singleton.

Implications may be syntactically managed by means of inference systems. The former axiomatic system was Armstrong's Axioms [1]. They allows us to introduce the notion of derivation of an implication from an implicational system, the semantic entailment and the equivalence between two implicational systems in the usual way.

3 Simplification Logic

In [6], Cordero et al. introduced the Simplification Logic, $\mathbf{SL}_{\text{FD}}$, that is, an equivalent logic to the Armstrong's Axioms that avoids the use of transitivity and is guided by the idea of simplifying the set of implications by removing redundant attributes efficiently. This logic has proved to be useful for automated reasoning with implications [7, 8, 12, 13].

Definition 3 (Language). *Given a non-empty finite alphabet S (whose elements are named attributes and denoted by lowercase letters a, b, c, etc.), the language of* $\mathbf{SL}_{\text{FD}}$ *is* $\mathcal{L}_S = \{A \to B \mid A, B \subseteq S\}$.

Sets of formulas (implications) will be named implicational systems (IS). In order to distinguish between language and metalanguage, inside implications, AB means $A \cup B$ and A-B denotes the set difference $A \smallsetminus B$. Moreover, when no confusion arises, we omit the brackets, e.g. abc denotes the set $\{a, b, c\}$.

Definition 4 (Semantics). *Let* $\mathbf{K} = (G, M, I)$ *be a context and* $A \to B \in \mathcal{L}_S$. *The context* $\mathbf{K}$ *is said to be a model for* $A \to B$, *denoted* $\mathbf{K} \models A \to B$, *if* $A, B \subseteq M \subseteq S$ *and* $A \to B$ *holds in* $\mathbf{K}$.

For a context $\mathbf{K}$ and an IS Σ, then $\mathbf{K} \models \Sigma$ means $\mathbf{K} \models A \to B$ for all $A \to B \in \Sigma$ and $\Sigma \models A \to B$ denotes that every model for Σ is also a model for $A \to B$. If Σ_1 and Σ_2 are implicational systems, $\Sigma_1 \equiv \Sigma_2$ denotes both IS are equivalent (i.e. $\mathbf{K} \models \Sigma_1$ iff $\mathbf{K} \models \Sigma_2$ for all context $\mathbf{K}$).

Definition 5 (Syntactic derivations). $\mathbf{SL}_{\text{FD}}$ *considers reflexivity axioms*

$$\texttt{[Ref]}\ \frac{B \subseteq A}{A \to B};$$

and the following inference rules named fragmentation, composition and simplification respectively.

$$\texttt{[Frag]}\ \frac{A \to BC}{A \to B};\quad \texttt{[Comp]}\ \frac{A \to B,\ C \to D}{AC \to BD};\quad \texttt{[Simp]}\ \textit{If } A \subseteq C, A \cap B = \emptyset,\ \frac{A \to B,\ C \to D}{C\text{-}B \to D\text{-}B}$$

Given an IS Σ and a formula $A \to B$, $\Sigma \vdash A \to B$ denotes that $A \to B$ can be derived from Σ by using the axiomatic system in a standard way. The above axiomatic system is sound and complete (i.e. $\Sigma \models A \to B$ iff $\Sigma \vdash A \to B$). The main advantage of $\mathbf{SL}_{\text{FD}}$ is that inferences rules may be considered equivalence rules and they are enough to compute all the derivations (see [12] for further details and proofs).

Theorem 1 (Mora et al. [12]). *In* $\mathbf{SL}_{\text{FD}}$ *logic, the following equivalencies hold:*

1. *Fragmentation Equivalency* **[FrEq]**: $\{A \to B\} \equiv \{A \to B\text{-}A\}$.
2. *Composition Equivalency* **[CoEq]**: $\{A \to B, A \to C\} \equiv \{A \to BC\}$.
3. *Simplification Equivalency* **[SiEq]**: *If* $A \cap B = \emptyset$ *and* $A \subseteq C$ *then*
$$\{A \to B, C \to D\} \equiv \{A \to B, C\text{-}B \to D\text{-}B\}$$
4. *Right Simplification Equivalency* **[rSiEq]**: *If* $A \cap B = \emptyset$ *and* $A \subseteq C \cup D$ *then*
$$\{A \to B, C \to D\} \equiv \{A \to B, C \to D\text{-}B\}$$

Note that these equivalencies (reading from left to right) remove redundant information. $\mathbf{SL}_{\text{FD}}$ was conceived as a simplification framework.

To conclude this section, we introduce the outstanding notion of closure of a set of attributes, which is strongly related with the syntactic treatment of implications.

Definition 6. *Let $\Sigma \subseteq \mathcal{L}_S$ be an IS and $X \subseteq S$. The closure of X wrt Σ is the largest subset of S, noted X^+_Σ, such that $\Sigma \vdash X \to X^+_\Sigma$.*

We omit the subindex (i.e. we write X^+) when no confusion arise. Given a context $\mathbf{K}$ and an IS Σ satisfying $\mathbf{K} \models A \to B$ iff $\Sigma \vdash A \to B$, it is well-known that the closed sets of attributes wrt Σ are in bijection with the concepts of $\mathbf{K}$.

One of the main topics is the computation of the closure of a set of attributes, and for this reason, it is necessary to have an efficient method to calculate closures. We emphasize for this problem, the works of Bertet et al. in [2, 3] and Cordero et al. in [12].

4 Direct-Optimal basis

The study of sets of implications that satisfies some criteria is one of the most important topics in FCA. In [3], Bertet and Monjardet present a survey about implicational systems and basis. They show the equality between five unit basis originating from different works (minimal functional dependencies in database theory, knowledge spaces, etc.) and satisfying various properties including the directness canonical and minimal properties, whence the name canonical direct basis is given to this basis. The direct-optimal basis belong to these five basis. In the following, we show only the concepts used in the rest of the paper of this survey.

Definition 7. *An IS Σ is said to be:*

- minimal *if $\Sigma \smallsetminus \{A \to B\} \not\equiv \Sigma$ for all $A \to B \in \Sigma$.*
- minimum *if $\Sigma' \equiv \Sigma$ implies $|\Sigma| \leq |\Sigma'|$, for all IS Σ'.*
- optimal *if $\Sigma' \equiv \Sigma$ implies $\|\Sigma\| \leq \|\Sigma'\|$, for all IS Σ'.*

where $|\Sigma|$ is the cardinality of Σ and $\|\Sigma\|$ is its size, ie $\|\Sigma\| = \sum\limits_{A \to B \in \Sigma} (|A| + |B|)$.

A minimal set of implications is named a basis, and a minimum basis is then a basis of least cardinality. Let us now introduce the main property used in this paper, namely the direct-optimal property.

Definition 8. *An IS Σ is said to be* direct *if, for all $X \subseteq S$:*

$$X^+ = X \cup \bigcup \{B \mid A \subseteq X \text{ and } A \to B \in \Sigma\}$$

Moreover, Σ is said to be direct-optimal *if it is direct and, for any direct IS Σ', $\Sigma' \equiv \Sigma$ implies $\|\Sigma\| \leq \|\Sigma'\|$.*

In words, Σ is direct if the computation of the closure of any attribute set wrt Σ requires only one iteration, that is, a unique traversal of the set of implications. Obviously, the direct-optimal property is the combination of the directness and optimality properties. In [2], Bertet and Nebut show that a direct-optimal IS is unique and can be obtained from any equivalent IS. We address this procedure in this paper.

As we have said in the preliminaries, one of the most important problems is how to calculate quickly and easily the closure X^+ of any set X because a number of problems related to an IS Σ can be answered by computing closures. For this reason, Bertet et al. propose a type of base called direct-optimal basis [2, 3], so one can compute closures of subsets in only one iteration. Section 4.1. presents the basis proposed in [2] by Bertet and Nebut where they work with non-unitary implicational systems (IS). Section 4.2 shows how to obtain a unit direct-optimal basis [3]. In both sections, we illustrate the algorithms needed to obtain a direct-optimal basis equivalent to any implicational system.

4.1 Computing Direct-Optimal basis

In this section, the algorithm proposed by Bertet and Nebut in [2] is showed. The key of the method is the so-called "overlap axiom" that can be directly proved by using the axiomatic system from Definition 5.

`[Overlap]` for all $A, B, C, D \subseteq S$: If $B \cap C \neq \emptyset$, $\dfrac{A \to B, C \to D}{A(C\text{-}B) \to D}$

Then, the direct implicational system generated from an IS Σ is defined as the smallest IS that contains Σ and is closed for `[Overlap]`.

Definition 9. *The direct implicational system Σ_d generated from Σ is defined as the smallest IS such that:*

1. *$\Sigma \subseteq \Sigma_d$ and*
2. *For all $A, B, C, D \subseteq S$, if $A \to B, C \to D \in \Sigma_d$ and $B \cap C \neq \emptyset$ then $A(C\text{-}B) \to D \in \Sigma_d$.*

```
Function Bertet-Nebut-Direct($\Sigma$)
  input  : An implicational system $\Sigma$ on $S$
  output: The direct IS $\Sigma_d$ on $S$ equivalent to $\Sigma$
  begin
    $\Sigma_d := \Sigma$
    foreach $A \to B \in \Sigma_d$ do
      foreach $C \to D \in \Sigma_d$ do
        if $B \cap C \neq \emptyset$ then add $A(C\text{-}B) \to D$ to $\Sigma_d$;
    return $\Sigma_d$
```

Theorem 2 (Bertet and Nebut [2]). *Let Σ be an implicational system. Then $\Sigma_d =$ `Bertet-Nebut-Direct`(Σ) is a direct basis.*

Moreover, if an IS Σ is direct but not direct-optimal, then there exists an equivalent IS Σ' of smaller size which is direct-optimal. The properties that it must hold are the following:

Theorem 3 (Bertet and Nebut [2]). *A direct IS Σ is direct-optimal if and only if the following properties hold.*

Extensiveness: *for all $A \to B \in \Sigma$, $A \cap B = \emptyset$.*
Isotony: *for all $A \to B, C \to D \in \Sigma$, $C \subsetneq A$ implies $B \cap D = \emptyset$.*
Premise: *if $A \to B, A \to B' \in \Sigma$ then $B = B'$.*
Not empty conclusion: *if $A \to B \in \Sigma$ then $B \neq \emptyset$.*

Function Bertet-Nebut-Minimize(Σ)

```
input  : An implicational system $\Sigma$ on $S$
output: An smaller IS $\Sigma_m$ on $S$ equivalent to $\Sigma$
begin
    $\Sigma_m := \emptyset$
    foreach $A \to B \in \Sigma$ do
        $B' := B$
        foreach $C \to D \in \Sigma$ do
            if $C = A$ then $B' := B' \cup D$;
            if $C \subsetneq A$ then $B' := B' \smallsetminus D$;
        $B' := B' \smallsetminus A$
        add $A \to B'$ to $\Sigma_m$
    return $\Sigma_m$
```

Function `Bertet-Nebut-DO` computes the direct-optimal basis Σ_{do} generated from an IS Σ. It first computes Σ_d using Function `Bertet-Nebut-Direct` and then minimizes Σ_d using Function `Bertet-Nebut-Minimize`.

Function Bertet-Nebut-DO(Σ)

```
input  : An implicational system $\Sigma$ on $S$
output: The direct-optimal IS $\Sigma_{do}$ on $S$ equivalent to $\Sigma$
begin
    $\Sigma_d$ = Bertet-Nebut-direct($\Sigma$)
    $\Sigma_{do}$ = Bertet-Nebut-Minimize($\Sigma_d$)
    return $\Sigma_{do}$
```

Theorem 4 (Bertet and Nebut [2]). *Let Σ be an implicational system. Then Σ_{do} =* `Bertet-Nebut-DO`*(Σ) is the unique direct-optimal implicational system equivalent to Σ.*

4.2 Direct-Optimal basis by means of unit implicational systems

In some areas, the management of formulas is limited to unitary ones. Thus, the use of Horn Clauses in Logic Programming is widely accepted. Such a language restriction allows an improvement in the performance of the methods, which are more direct and lighter. Nevertheless, the advantages provided by the limited languages have a counterpart: a significant growth of the input set. In this section we are going to present new versions of the definitions and methods introduced above restricted to Unit Implicational System (UIS), i.e. set of implications with unitary right-hand sides. An UIS is named proper if it does not contain implications $A \to a$ such that $a \in A$.

In this line, Bertet [4] provided versions for unit implicational systems of Functions `Bertet-Nebut-Direct` and `Bertet-Nebut-Minimize`.

```
Function Bertet-Unit-Direct($\Sigma$)
  input  : A proper UIS $\Sigma$ on $S$
  output: The direct UIS $\Sigma_d$ on $S$ equivalent to $\Sigma$
  begin
      $\Sigma_d := \Sigma$
      foreach $A \to a \in \Sigma_d$ do
          foreach $Ca \to b \in \Sigma_d$ do
              if $a \neq b$ and $b \notin A$ then add $AC \to b$ to $\Sigma_d$;
      return $\Sigma_d$
```

```
Function Bertet-Unit-Minimize($\Sigma$)
  input  : A proper UIS $\Sigma$ on $S$
  output: An smaller UIS $\Sigma_m$ on $S$ equivalent to $\Sigma$
  begin
      $\Sigma_m := \Sigma$
      foreach $A \to b \in \Sigma_m$ do
          foreach $C \to b \in \Sigma_m$ do
              if $A \subsetneq C$ then delete $C \to b$ from $\Sigma_m$;
      return $\Sigma_m$
```

The above functions was used in [4] to build a method which transforms an arbitrary UIS into an UIS with the same properties that the direct-optimal basis for general IS. Since any non-unit IS can be trivially turned into an UIS, we may encapsulate both functions to provide another method to get a direct-optimal basis from and arbitrary IS. Thus, the following function incorporates a first step to convert any IS into its equivalent UIS and concludes with the converse switch.

```
Function Bertet-Unit-DO($\Sigma$)
  input  : An implicational system $\Sigma$ on $S$
  output: The direct-optimal IS $\Sigma_{do}$ on $S$ equivalent to $\Sigma$
  begin
      $\Sigma_u := \{A \to b \mid A \to B \in \Sigma$ and $b \in B \smallsetminus A\}$
      $\Sigma_{ud} :=$ Bertet-Unit-Direct($\Sigma_u$)
      $\Sigma_{udo} :=$ Bertet-Unit-Minimize($\Sigma_{ud}$)
      $\Sigma_{do} := \{A \to B \mid B = \{b \mid A \to b \in \Sigma\} \neq \emptyset\}$
      return $\Sigma_{do}$
```

Theorem 5 (Bertet [4]). *Let Σ be an IS. Then Σ_{do} =* `Bertet-Unit-DO`(Σ) *is the unique direct-optimal implicational system equivalent to Σ.*

As we have mentioned at the beginning of this subsection, some authors introduce unitary formulas as a way to provide simpler and more direct methods having a better performance. Thus, in this case, `Bertet-Unit-DO` is more efficient than `Bertet-Nebut-DO`, as we shall see at the end of the paper in Section 5.1.

5 Computing direct-optimal basis by means of reductions

In this paper, our goal is the integration of the techniques proposed by Bertet et al. [2–4] and the Simplification Logic proposed by Cordero et al. [6], that is, the adding of reductions based on the simplification paradigm to build a direct-optimal basis.

In the same way that `Bertet-Unit-DO`, we are going to develop a function to get direct-optimal basis whose first step will be to narrow the implications. However, the use of unit implications has some disadvantages that we are going to avoid by considering another kind of formulas. Thus, we are going to use *reduced* IS and introduce simplification rules which transform it preserving reduceness. A signal which indicates it is a good approach is the fact that at the end of the process, the function renders the direct-optimal basis directly, avoiding the converse switch.

Definition 10. *An IS Σ is* reduced *if $A \to B \in \Sigma$ implies $B \neq \emptyset$ and $A \cap B = \emptyset$ for all $A, B \subseteq S$.*

Obviously, any IS Σ can be turned into a reduced equivalent one Σ_r as follows

$$\Sigma_r := \{A \to B\text{-}A \mid A \to B \in \Sigma, B \not\subseteq A\}$$

The method proposed begins with this transformation and, once the IS is reduced, this property is preserved. For this reason, `[Overlap]` must be substituted. Thus, we introduce a new inference rule covering directness without losing reduceness and, at the same time, it makes progress on the minimization task following the simplification paradigm. The kernel of the new method is the following inference rule, named strong simplification:

$$\texttt{[sSimp]} \quad \text{If } B \cap C \neq \emptyset \text{ and } D \not\subseteq A \cup B, \ \frac{A \to B, C \to D}{A(C\text{-}B) \to D\text{-}(AB)}$$

Regardless the conditions, the inference rule always holds. Nevertheless, the conditions ensure a precise application of the rule in those cases where it is necessary.

Definition 11. *Given a reduced IS Σ, the* direct-reduced implicational system *Σ_{dr} generated from Σ is defined as the smallest IS such that*

1. *$\Sigma \subseteq \Sigma_{dr}$ and*
2. *For all $A, B, C, D \subseteq S$, if $A \to B, C \to D \in \Sigma_{dr}$, $B \cap C \neq \emptyset$ and $D \not\subseteq A \cup B$ then $AC\text{-}B \to D\text{-}(AB) \in \Sigma_{dr}$*

Theorem 6. *Given a reduced IS Σ, then Σ_{dr} =*`Direct-Reduced`*(Σ) is a direct and reduced IS.*

Function Direct-Reduced(Σ)

```
input  : A reduced implicational system $\Sigma$ on $S$
output: The direct-reduced IS $\Sigma_{dr}$ on $S$
begin
    foreach $A \to B \in \Sigma_{dr}$ and $C \to D \in \Sigma_{dr}$ do
        if $B \cap C \neq \emptyset \neq D \smallsetminus (A \cup B)$ then add $AC$-$B \to D$-$(AB)$ to $\Sigma_{dr}$;
    return $\Sigma_{dr}$
```

Theorem 1 provides four equivalencies which allow to remove redundant information when they are read from left to right. An implicational system in which these equivalences are used to remove redundant information is going to be named simplified implicational system.

Definition 12. *A reduced IS Σ is* simplified *if the following conditions hold: for all $A, B, C, D \subseteq S$,*

1. *$A \to B$, $A \to C \in \Sigma$ implies $B = C$.*
2. *$A \to B$, $C \to D \in \Sigma$ and $A \subsetneq C$ imply $C \cap B = \emptyset = D \cap B$.*

Then, Function `RD-Simplify` turns any direct-reduced IS into a direct-reduced-simplified equivalent one by systematically applying the equivalences provided in Theorem 1.

Function RD-Simplify(Σ)

```
input  : A direct-reduced implicational system $\Sigma$ on $S$
output: The direct-reduced-simplified IS $\Sigma_{drs}$ on $S$ equivalent to $\Sigma$
begin
    $\Sigma_{drs} := \emptyset$
    foreach $A \to B \in \Sigma$ do
        foreach $C \to D \in \Sigma$ do
            if $C = A$ then $B := B \cup D$;
            if $C \subsetneq A$ then $B := B \smallsetminus D$;
        if $B \neq \emptyset$ then add $A \to B$ to $\Sigma_{drs}$;
    return $\Sigma_{drs}$
```

Function doSimp(Σ)

```
input  : An implicational system $\Sigma$ on $S$
output: The direct-optimal IS $\Sigma_{do}$ on $S$
begin
    $\Sigma_r := \{A \to B\text{-}A \mid A \to B \in \Sigma, B \not\subseteq A\}$
    $\Sigma_{dr} :=$ Direct-Reduced($\Sigma_r$)
    $\Sigma_{do} :=$ RD-Simplify($\Sigma_{dr}$)
    return $\Sigma_{do}$
```

Theorem 7. *Let Σ be an implicational system on S. Then, $\Sigma_{do} = \texttt{doSimp}(\Sigma)$ is the direct-optimal basis equivalent to Σ.*

Note that, unlike `Bertet-Unit-DO` where a final step was needed to revert the effects of the first transformation, `doSimp` do not need to revert the first step. We conclude this section with an experiment which illustrates the advantages of the new method.

5.1 Empirical results

Logic programming has been used as a natural framework in the areas in which it is neccessary to develop automatic deduction methods. The Prolog prototypes provides a declarative and pedagogical point of departure and illustrates the behavior of new techniques in a very fast and easy way.

Some authors have explored the use of Logic Programming in the framework of Formal Concept Analysis. Even, in [5] the authors consider the framework of FCA and its implementation in logic programming as a previous step to achieve the first order logic FCA theory. In Eden et al. [9], the authors present a PROLOG-based prototype tool and show how the tool can utilize formulas to locate pattern instances.

In a first step, the methods proposed in this paper have been developed in a Logic Programming language (Prolog) that is a well-known tool to develop fast prototypes. In our case, the implementation in Prolog is close because the method proposed in this paper is based on logic.

The methods of Bertet et al. [2,3] and our `doSimp` method have been implemented in Swi-Prolog.[1] Since there does not exist a benchmark for implications in this experiment, we have collected some sets of implications from the literature, searching papers and books with works about algorithms for implications, functional dependencies and minimal keys. Now, we are going to show the results of the execution of a first Prolog prototype of Bertet et al. for UIS [3], Bertet et al. for IS [2] and the new doSimp (proposed in this paper) methods.

The following table and figures summarize the results obtained. We show in the columns the results of Prolog: Lips (logical inferences per second lips - used to describe the performance of a logical reasoning system), Time (execution time in seconds), and Comp (the number of couple of implications in which a rule is applied). Areas in Figure 2 show the percentages of each algorithm with respect the number of comparisons.

[1] Available at `http://www.lcc.uma.es/~enciso/doSimp.zip`

Lips/Time/Comp.	Bertet-Nebut-DO			Bertet-Unit-DO			Direct-Reduced		
Ex.1	5297080	1247	1978	116905	0.019	36	**4281**	**0.001**	**12**
Ex.2	2395	0.003	23	923	**0**	3	**606**	**0**	**2**
Ex.3	2183	**0**	15	1440	**0**	**4**	**1122**	**0**	**4**
Ex.a	83403	0.019	297	44109	0.007	33	**3048**	**0.001**	**4**
Ex.a3red	27613	0.005	100	16938	0.003	20	**3698**	**0.001**	**15**
Ex.derivation5	10302	0.002	120	3522	**0.001**	**8**	**1782**	**0.001**	12
Ex.Olomouc	15399581	4528	4337	1526818	0.331	180	**15568**	**0.003**	**72**
Ex.Ganter	116514	0.025	230	72153	0.16	36	**3756**	**0.001**	**12**
Ex.CLA14	102971	0.022	204	7449	0.001	12	**704**	**0**	**3**
Ex.Saedian1	18754	0.004	97	10349	0.002	**14**	**4064**	**0.001**	16
Ex.Saedian2	19452	0.004	160	10549	0.002	**13**	**2619**	**0.001**	**13**
EX.Saedian3	5753962	1262	1986	166566	0.028	67	**24643**	**0.005**	**55**
Ex.Wastl10	1242	**0**	18	381	**0**	**1**	**327**	**0**	**1**
Ex.Wastl13	10543	0.002	86	4674	**0**	10	**1029**	**0**	**5**
Example1	5594556921	7008.890	134175	2662181973	1351.950	5389	**1199498**	**0.197**	**1103**

	$IS\ Bertet - Nebut$	$UIS\ Bertet$	$doSimp$
Lips - logical inferences	$374,760,194.4$	$177,610,983.3$	$84,449.66667$
Time of execution (seconds)	$467,728.5$	$90,130.03693$	0.014
Number of comparisons	9588.4	388.4	88.6

Fig. 1. Summary of the experiment (average)

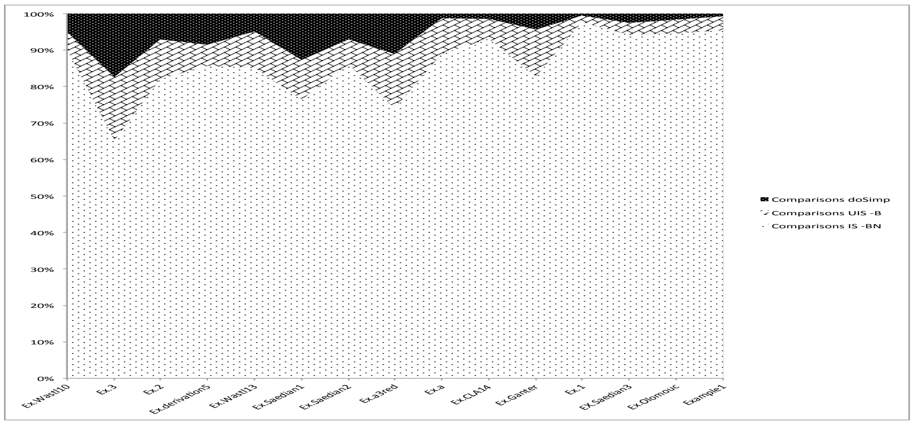

Fig. 2. Results: Comparisons

6 Conclusion

In this work, we have presented another algorithm to calculate the direct-optimal basis in a further way, in the most of the cases, than the algorithms which exist in the literature. It is shown with a test that we have realised by running different examples with the methods of Bertet et al. for UIS [3], Bertet et al. for IS [2] and the new `doSimp`.

Our aim is to reduce the cost of the algorithm by using the Simplification Logic as a useful tool to work with implications. By the time, we have improved the algorithms that existed but we are going to go on working in that way to try to cut down the cost of our method.

The perspectives we have are improvements by pretreatments: reduction, canonical basis, etc in order to reach our main objective which would be to directly compute the direct-optimal basis without extra implication generation.

Acknowledgment

Supported by grant TIN11-28084 of the Science and Innovation Ministry of Spain.

References

1. W W. Armstrong, *Dependency structures of data base relationships*, Proc. IFIP Congress. North Holland, Amsterdam: 580–583, 1974.
2. K. Bertet, M. Nebut, *Efficient algorithms on the Moore family associated to an implicational system*, DMTCS, 6(2): 315–338, 2004.
3. K. Bertet, B. Monjardet, *The multiple facets of the canonical direct unit implicational basis*, Theor. Comput. Sci., 411(22-24): 2155–2166, 2010.
4. K. Bertet, *Some Algorithmical Aspects Using the Canonical Direct Implicationnal Basis*, CLA:101–114, 2006.
5. L. Chaudron, N. Maille, *1st Order Logic Formal Concept Analysis: from logic programming to theory*, Computer and Informations Science: (13:3),1998.
6. P Cordero, A. Mora, M. Enciso, I.Pérez de Guzmán, *SLFD Logic: Elimination of Data Redundancy in Knowledge Representation*, LNCS, 2527: 141–150, 2002.
7. P. Cordero, M. Enciso, A. Mora, M. Ojeda-Aciego, *Computing Minimal Generators from Implications: a Logic-guided Approach*, CLA: 187–198, 2012.
8. P. Cordero, M. Enciso, A. Mora, M. Ojeda-Aciego, *Computing Left-Minimal Direct Basis of implications.* CLA: 293–298, 2013.
9. A. Eden, Y. Hirshfeld, K. Lundqvist, *EHL99 , LePUS Symbolic Logic Modeling of Object Oriented Architectures: A Case Study*, In: Proc. Second Nordic Workshop on Software Architecture (NOSA'99), 1999.
10. B. Ganter, *Two basic algorithms in concept analysis*, Technische Hochschule, Darmstadt, 1984.
11. J.L. Guigues and V. Duquenne, *Familles minimales d'implications informatives résultant d'un tableau de données binaires*, Math. Sci. Humaines: 95, 5–18, 1986.
12. A. Mora, M. Enciso, P. Cordero, and I. Fortes, *Closure via functional dependence simplification*, I. J.of Computer Mathematics, 89(4): 510–526, 2012.
13. A. Mora, M. Enciso, P. Cordero, and I. Pérez de Guzmán, *An Efficient Preprocessing Transformation for Functional Dependencies Sets Based on the Substitution Paradigm*, LNCS, 3040: 136–146, 2004.