From an implicational system to its corresponding *D*-basis

Estrella Rodríguez-Lorenzo[1], Kira Adaricheva[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@uma.es, enciso@lcc.uma.es
[2] Nazarbayev University, Kazakhstan
e-mail: kira.adaricheva@nu.edu.kz

Abstract. Closure system is a fundamental concept appearing in several areas such as databases, formal concept analysis, artificial intelligence, etc. It is well-known that there exists a connection between a closure operator on a set and the lattice of its closed sets. Furthermore, the closure system can be replaced by a set of implications but this set has usually a lot of redundancy inducing non desired properties.
In the literature, there is a common interest in the search of the minimality of a set of implications because of the importance of bases. The well-known Duquenne-Guigues basis satisfies this minimality condition. However, several authors emphasize the relevance of the optimality in order to reduce the size of implications in the basis. In addition to this, some bases have been defined to improve the computation of closures relying on the directness property. The efficiency of computation with the direct basis is achieved due to the fact that the closure is computed in one traversal.
In this work, we focus on the *D*-basis, which is ordered-direct. An open problem is to obtain it from an arbitrary implicational system, so it is our aim in this paper. We introduce a method to compute the *D*-basis by means of minimal generators calculated using the Simplification Logic for implications.

1 Introduction

Discovering knowledge and information retrieval are currently active issues where Formal Concept Analysis (FCA) provides tools and methods for data analysis. The notions around the concept lattice may be considered as the main attractions in Formal Concept Analysis and they are strongly connected to the notion of closure.

Closure system is a fundamental concept appearing in several areas such as database theory, formal concept analysis, artificial intelligence, etc. It is well-known that there exists a connection between a closure operator on a set and the lattice of its closed sets. Furthermore, the closure system can be presented, dually, as a set of attribute implications, namely an implicational system but this set has usually a lot of redundancy inducing non-desired properties.

ISBN 978–2–9544948–0–7, ISSN 2311–701X, Blaise Pascal University, LIMOS laboratory, Clermont-Ferrand, 2015.

We can not fail to mention the relevance of the role of the implication notion in different areas. It was the main actor of the normalization theory in database area; it has an outstanding character in Formal Concept Analysis and it was prominently used in Frequent Set Mining and Learning Spaces, see the survey of M. Wild [10]. The latter is devoted to mathematical theory of implications and the different faces of the concept of an implication. Implications linked data represented in several forms going from the relationship between itemsets in transactions (Frequent Set Mining) to the boolean functions (Horn Theory).

Nonetheless, as V. Duquenne says in [6] "it is surprising if not hard to acknowledge that we did not learn much more on *their intimacy* in the meantime, despite many interesting papers using or revisiting them". We believe there is a long way to go, and a deeper theory on properties of implications with automated and efficient methods to manipulate them can be developed.

In this paper, we are focused in the Formal Concept Analysis area and the fundamental notions are assumed (see [7]). The task of information retrieval carried out by the tools in FCA conduits to infer *concepts* from the data set, i.e. to deduce (in an automated way) a set of objects that may be precisely characterized by a set of attributes. Such concepts inherit an order relation induced by attribute set inclusion, providing a lattice structure of the concept set. Here implications are retrieved from a binary table (formal context) representing the relationship between a set of objects and a set of attributes. Implications represent an alternative way for the underlying information contained in the formal context.

Many applications must massively compute closures of sets of attributes and any improvement of execution time is relevant. In [9] the author establishes the necessity of obtaining succinct representation of closure operators to achieve an efficient computational usage. In this direction, properties associated to implications are studied to render equivalent sets fulfilling desired properties, *directness* and *optimality*.

An important matter in FCA is to transform implicational systems in canonical forms for special proposals in order to provide an efficient further management. Hence, some alternative definitions have been established: Duquenne-Guigues basis, direct optimal basis, D-basis, etc. In this work we focus on the last one [1], because it combines, in a balanced way, a brief representation (it has a small number of elements) and a efficient computation of closures (it is computed in just one traversal). To this end, D-basis proposes an order in which implications will be attended.

The major issue is that the execution of the D-basis in one iteration is more efficient that the execution of a shorter, but un-ordered one, for instance the *canonical basis* of Duquenne and Guigues. K. Adaricheva et.al prove in [1] that one can extract the D-basis from any direct unit basis Σ in time polynomial of size of Σ, and it takes only linear time of the number of implications of the D-basis to put it into a proper order.

In [5] we have proposed a method to calculate all the minimal generators from a set of implications as a way to remove redundancy in the basis. The method

to compute all the minimal generators is based on the Simplification Logic for implications [8]. Using this logic we are able to remove redundancy in the implications [4] and following the same style of application of the Simplification Rule to the set of implications we can obtain all the minimal generators.

Currently the retrieval of the D-basis from an arbitrary implicational system is an open problem, so it becomes our aim in this paper. We introduce a method to compute the D-basis by means of minimal generators calculated using the Simplification Logic for implications. The relationship among minimal generators, covers, minimal covers and D-basis is presented and an algorithm to calculate D-basis from an arbitrary set of implications is proposed.

Section 2 presents the main notions necessary to the understanding of the new method: closure operators, the D-basis, Simplification Logic and the method to calculate minimal generators. In Section 3, the relationships between covers and generators are presented. In Section 4, the new method to obtain the D-basis from a set of implications is shown, and some conclusions and extensions are proposed in Section 5.

2 Background

2.1 Closure systems

Given a non-empty set M and the set[1] 2^M of all its subsets, *a closure operator* is a map $\phi : 2^M \to 2^M$ that satisfies the following, for all $X, Y \in 2^M$:

(1) increasing: $X \subseteq \phi(X)$;
(2) isotone: $X \subseteq Y$ implies $\phi(X) \subseteq \phi(Y)$;
(3) idempotent: $\phi(\phi(X)) = \phi(X)$.

We will refer to the pair $\langle M, \phi \rangle$ of a set M and a closure operator on it as a *closure system*.

In the next two subsections we will follow the introduction of the implicational system based on the *minimal proper covers*[2] given in [1], which was named there the D-basis.

We will call closure system *reduced*, if $\phi(\{x\}) = \phi(\{y\}) \rightarrow x = y$, for any $x, y \in M$ [3]. If the closure system $\langle M, \phi \rangle$ is not reduced, one can modify it to produce an equivalent one that is reduced, see [1] for more details.

We will now define a closure operator ϕ^*, which is associated with a given operator ϕ.

Definition 1. *Let $\langle M, \phi \rangle$ be a closure system. Define ϕ^* as a self-map on 2^M such that $\phi^*(X) = \bigcup_{x \in X} \phi(x), X \subseteq 2^M$.*

It is straightforward to verify that

[1] In the FCA framework, that set M can be thought a set of attributes of a context.
[2] Although in [1] it was introduced as minimal cover, here we name it minimal proper cover because in this paper we generalize the notion of cover in Section 3.
[3] To clarify the notation $\phi(\{x\})$ will be represented as $\phi(x)$ if no risk of confusion.

Lemma 1. *ϕ^* is a closure operator on M.*

Given a closure system $\langle M, \phi \rangle$, we introduce several important concepts.

Definition 2 ([1]). *For $x \in M$ we call a subset $X \subseteq M$ a proper cover for x if $x \in \phi(X) \setminus \phi^*(X)$. If X is a proper cover for x, it will be denoted as $x \sim_p X$.*

2.2 The *D*-basis

In this subsection, we briefly summarize the introduction of the D-basis in [1]. Its definition is strongly based on the notion of a minimal proper cover:

Definition 3. *A proper cover Y for x is called minimal, if, for any other proper cover Z for x, $Z \subseteq \phi^*(Y)$ implies $Y \subseteq Z$.*

The existence of several proper covers for the same element induces the need to introduce the notion of minimality.

Lemma 2. *If $x \sim_p X$, then there exists Y such that $x \sim_p Y$, $Y \subseteq \phi^*(X)$ and Y is a minimal proper cover for x. In other words, every proper cover can be reduced to a minimal proper cover under the subset relation added with the ϕ^* operator.*

These ideas bring to the following definition of the implicational system defining the reduced closure system by means of the minimal proper covers.

Definition 4. *Given a reduced closure system $\langle M, \phi \rangle$, define the D-**basis** Σ_D as a union of two subsets of implications:*

1. *$\{y \to x : x \in \phi(y) \setminus y, y \in M\}$ (such implications are called* binary*);*
2. *$\{X \to x : X$ is a minimal proper cover for $x\}$.*

Note that the D-basis belongs to the family of the *unit* bases, i.e. implicational sets where each implication $A \to b$ has a singleton $b \in M$ as a consequent.

Lemma 3. *Σ_D generates $\langle M, \phi \rangle$.*

2.3 Ordered direct set of implications

Here we recall the notion of the ordered direct basis introduced in [1], which is designed for a quick computation of the closures based on some fixed order of implications. First we recall the definition of the ordered iteration of implications.

Definition 5. *Suppose the set of implications Σ is equipped with some linear order, or equivalently, the implications are indexed as $\Sigma = \{s_1, s_2, \ldots, s_n\}$. Define a mapping $\rho_\Sigma : 2^M \to 2^M$ associated with this ordering as follows. For any $X \subseteq M$, let $X_0 = X$. If X_k is computed and implication s_{k+1} is $A \to B$, then*

$$X_{k+1} = \begin{cases} X_k \cup B, & \text{if } A \subseteq X_k, \\ X_k, & \text{otherwise.} \end{cases}$$

Finally, $\rho_\Sigma(X) = X_n$. We will call ρ_Σ an ordered iteration *of Σ.*

The concept of the ordered iteration is central for the definition of the ordered direct basis. For any given set of implications Σ on set M, by ϕ_Σ we understand the closure operator on M defined by Σ. Equivalently, the fixed points of ϕ_Σ are exactly subsets $X \subseteq M$ which are stable for all implications $A \to B$ in Σ: if $A \subseteq X$, then $B \subseteq X$.

Definition 6. *The set of implications with some linear ordering on it, $\langle \Sigma, < \rangle$, is called an* ordered direct basis*, if, with respect to this ordering, $\phi_\Sigma(X) = \rho_\Sigma(X)$ for all $X \subseteq S$.*

We note that any *direct* basis is ordered direct. By direct basis we understand any set of implications that allows to produce the closure of subsets of the base set while attending each implication only once.

More precisely, if Σ is some set of implications defining the closure system $\langle M, \phi \rangle$, then let $\pi_\Sigma(X) = X \cup \bigcup\{B : A \subseteq X \text{ and } (A \to B) \in \Sigma\}$. In order to obtain $\phi(X)$, for any $X \subseteq M$, one would normally need to repeat several iterations of π_Σ: $\phi(X) = \pi_\Sigma(X) \cup \pi^2_\Sigma(X) \cup \pi^3_\Sigma(X) \ldots$.

The bases for which one can obtain the closure of any set X performing only one iteration of π_Σ, i.e., $\phi(X) = \pi_\Sigma(X)$, are called *direct*. The various direct bases appearing in the literature were surveyed in K. Bertet and B. Monjardet [3]. Important result of their analysis is that in the family of all possible (unit) direct bases for the same closure system, there exists a $\subseteq$-smallest direct basis Σ_{cd}, which was called as *canonical unit direct* basis.

The following result from [1] summarizes the relation between the D-basis Σ_D and Σ_{cd} for a given closure system.

Theorem 1. $\Sigma_D \subseteq \Sigma_{cd}$.

2.4 Minimal Generators via Simplification Logic

In this subsection we summarize how the inference system of Simplification Logic $\mathbf{SL}_{\text{FD}}$ [4, 8] (equivalent to Armstrong's axioms) is used to enumerate all minimal generators. $\mathbf{SL}_{\text{FD}}$ is guided by the idea of simplifying the set of implications by efficiently removing some redundant attributes.

$\mathbf{SL}_{\text{FD}}$ logic considers reflexivity as axiom scheme

$$\texttt{[Ref]} \quad \frac{A \supseteq B}{A \to B}$$

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

$$\texttt{[Frag]} \ \frac{A \to B \cup C}{A \to B} \qquad \texttt{[Comp]} \ \frac{A \to B,\ C \to D}{A \cup C \to B \cup D} \qquad \texttt{[Simp]} \ \frac{A \to B,\ C \to D}{A \cup (C \smallsetminus B) \to D}$$

The important matter is that these rules can be considered as equivalence rules which have been used as the core in automated methods for removing redundancies, obtaining minimal keys, or computing closures. In particular, the last method indicated is based on the following results (see [8] for details, proofs and examples):

Theorem 2 ([8]). *Let $A, B \subseteq M$ and Σ be a set of implications. We have $\Sigma \vdash A \to B$ if and only if $\{\emptyset \to A\} \cup \Sigma \vdash \emptyset \to B$.*

In order to compute closures, since $\phi(A)$ is the biggest subset B such that $\Sigma \vdash A \to B$, the formula $\emptyset \to A$ is used as a seed which guides the reasoning to render the closure $\phi(A)$ just by applying the following equivalences where the [`Simp`] inference rule plays a main role.

Proposition 1. *Let A, B and C be subsets of M, then the following equivalences hold:*

- **Eq. I***: If $B \subseteq A$ then $\{\emptyset \to A, B \to C\} \equiv \{\emptyset \to A \cup C\}$.*
- **Eq. II***: If $C \subseteq A$ then $\{\emptyset \to A, B \to C\} \equiv \{\emptyset \to A\}$.*
- **Eq. III***: Otherwise $\{\emptyset \to A, B \to C\} \equiv \{\emptyset \to A, B \setminus A \to C \setminus A\}$.*

Based on the above proposition and Theorem 2, in [5] the method `Cls` is introduced. This method receives as input set $A \subseteq M$ and a set of implications Σ and renders the pair $(\phi(A), \Sigma')$ where Σ' is the set of implications simplified with respect to $\emptyset \to \phi(A)$ (i.e. Σ contracted to $2^{M \setminus \phi(A)}$).

NOTATION: From now on, in order to simplify the examples, elements of M are denoted by numbers from 0 to 9 or lowercase letters and we omit brackets and commas in subsets of M.

Example 1. Let Σ be $\{ab \to c, ac \to df, bcd \to ef, f \to c\}$. Then, $\texttt{Cls}(af, \Sigma) = (acdf, \{b \to e\})$ where $\phi(af) = acdf$ and $\Sigma' = \{b \to e\}$ has important information that will be used in the computation of minimal generators.

For $X, Y \subseteq M$ satisfying that $X = \phi(Y)$, it is usual to say that Y is a generator of the closed set X. Notice that any subset of X containing Y is also a generator of X. When the base set M of a closure system is finite, the set of generators of a closed set can be characterized by its minimal ones.

Definition 7 (Minimal Generator). *Let $\langle M, \phi \rangle$ be a closure system. $X \subseteq M$ is said to be a minimal generator if, for all proper subsets $Y \subset X$, one has $\phi(Y) \subset \phi(X)$.*

In [5], a method to compute all minimal generators is presented. This method named `MinGen` is based on above mentioned closure algorithm `Cls` and it takes the advantages of the additional information that it provides. That is, `Cls` is used not only to compute closed sets from generators but also a smaller implicational set which guides us in the search of new subsets to be considered as minimal generators. The input of `MinGen` is a subset of M and a set of implications Σ and the output is the set of closed sets endowed with all the minimal generators that generate them, i.e. $\{\langle C, mg(C) \rangle : C \text{ is a closed set}\}$ where $mg(C)$ is the set of minimal generators D satisfying $\phi(D) = C$. It can be seen as the lattice of the closure system in which we have added labels with the minimal generators to each closed set.

Example 2. Let $M = \{a, b, c, d\}$ and $\Sigma = \{a \to b, c \to bd, bd \to ac\}$. Then the MinGen algorithm returns the following set:

$$\texttt{MinGen}(M, \Sigma) = \{\langle abcd, \{c, ad, bd\}\rangle, \langle ab, \{a\}\rangle, \langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \emptyset, \{\emptyset\}\rangle\}$$

Observe that there is trivial information (e.g. $\langle b, \{b\}\rangle$) included in the output of this algorithm. In [5], a version of MinGen, named MinGen_0, is also presented. The difference between this algorithm and the previous one is that here only non-trivial generators are calculated.

For the same input, $\texttt{MinGen}_0(M, \Sigma) = \{\langle abcd, \{c, ad, bd\}\rangle, \langle ab, \{a\}\rangle, \langle \emptyset, \{\emptyset\}\rangle\}$

3 Covers and Generators: Relationships

The definition of a D-basis is strongly based on the notion of a proper minimal cover, and the latter is connected with the notion of a minimal generator. In this paper we are going to work with covers instead of proper covers because our idea is to manage, in a uniform way, the binary implications and non-binary ones. Throughout this section we will assume that M is the base set of a closure system defined either by closure operator ϕ, or by means of a set of implications Σ and its related operator ϕ_Σ. The following definition introduces a notion that extends the concept of a proper cover.

Definition 8 (Cover). *Given $X \subseteq M$ and $x \in M$, we say X is a cover of x, denoted $x \sim X$, if $x \in \phi(X) \setminus X$.*

The following proposition, whose proof is straightforward from definitions, illustrates the relationship among proper covers, covers and minimal generators.

Proposition 2. *Let Σ be a set of implications. For each set $C \subseteq M$ closed with respect to ϕ_Σ and each $X \subsetneq C$,*

1. *X is a generator of C if and only if X is a cover of any $x \in C \setminus X$.*
2. *If X is a proper cover of $x \in M$ then X is also a cover of x.*

Observe that the converse of the second item is not true, e.g. ad is a cover of b ($b \sim ad$) in Example 2 but it is not a proper cover. As we have remarked in Section 2, the above definition of a cover does not coincide with the one introduced in [1].

Corollary 1. *Let Σ be a set of implications. For each set $C \subseteq M$ closed with respect to ϕ_Σ and each $X \subsetneq C$, if X is a minimal generator of C then X is a cover of any $x \in C \setminus X$.*

The following example illustrates these relationships and gives a counterexample to the converse statement in the previous corollary.

Example 3. Let $M = \{a, b, c, d, e\}$ and $\Sigma = \{a \to d, bce \to ad, bde \to ac, ade \to bc, cd \to abe, abd \to ce\}$. In Table 1, the covers of each element of M and the non-trivial minimal generators of each ϕ_Σ-closed set are shown. Moreover, underlined covers are not proper covers whereas the others are.

Attribute	Covers
a	$cd, bcd, bce, bde, cde, bcde$
b	$ac, ae, cd, acd, ace, ade, cde, acde$
c	$ab, ae, abd, abe, ade, bde, abde$
d	$\underline{a}, \underline{ab}, \underline{ac}, \underline{ae}, \underline{abc}, \underline{abe}, \underline{ace}, bce, \underline{abce}$
e	$ab, ac, cd, abc, abd, acd, bcd, abcd$

Closed Sets	Minimal Generators
$abcde$	ab, ac, ae, bce, bde, cd
ad	a

Table 1. Covers, Proper Covers and Minimal Generators.

Observe that, by Proposition 2, X is a cover of an element x if and only if there exist a closed set C and a minimal generator Y of C such that $Y \subseteq X$ and $x \in C \smallsetminus X$. For example, cd is a minimal generator of $abcde$ and $cd \subseteq cde$, thus, cde is a cover of a but not a minimal generator.

The closure operator ϕ^* allows us to introduce the notion of a minimal cover. It will be used later to avoid redundancies in the implications of the basis when we look for an optimal basis.

Definition 9 (Minimal Cover). *Let Σ be a set of implications. Given $X \subseteq M$ and $x \in M$, X is a minimal cover of x if X is a cover and satisfies one of the following conditions:*

1. *X is a singleton, i.e., $\mid X \mid = 1$.*
2. *for every cover Y of x, $Y \subseteq \phi^*(X)$ implies $X \subseteq Y$.*

From this definition we directly obtain the following proposition that relates minimal generators and minimal covers.

Proposition 3. *Let Σ be a set of implications, $X \subseteq M$ and $x \in M$. If X is a minimal cover of $x \in \phi(X) \setminus X$, then X is a minimal generator of $\phi(X)$.*

The following example illustrates the definition of minimal covers and shows that the converse statement to the above proposition is not true.

Example 4. Given the base set and the set of implications of Example 3, Table 2 shows the minimal covers of each element.

Element	Minimal Covers
a	cd, bce, bde
b	ae, cd
c	ab, ae, bde
d	a, bce
e	ab, cd

Table 2. Minimal Covers.

Although ae is a minimal generator of $abcde$, it is not a minimal cover for all $x \in \phi(\{a, b, c, d, e\}) \setminus \{a, e\} = \{b, c, d\}$ but only for b and c.

As the previous examples show, a minimal cover is always a minimal generator, but not conversely. Thus, if we compute all minimal generators for some $x \in M$, we have, at the same time, all minimal covers for x, and the latter are exactly what we need for the computation of the D-basis.

4 *D*-basis by means of Minimal Generators

We remark that it is an open problem to design an algorithm rendering the D-basis from an arbitrary implicational system. In this section, we are going to introduce such an algorithm based on the strong connection between minimal covers and minimal generators. This is shown in Algorithm 1. To begin with, we define the Function `MinimalCovers` to make the algorithm easier to understand.

The input of the Function `MinimalCovers` is a set of covers of a given $x \in M$ and it returns a subset with all its minimal covers.

Function MinimalCovers(L)

input : A set of covers L
output: The set of minimal covers
begin
  foreach $g \in L$ **do**
    foreach $h \in L \setminus g$ **do**
      if $h \subseteq g$ **then**
        remove g from L
      else if $\mid g \mid \neq 1$ **and** $h \subseteq \bigcup_{x \in g} \phi(x)$ **then**
        remove g from L
  return L

Notice that although the condition $h \subseteq g$ implies $h \subseteq \bigcup_{x \in g} \phi(x)$ and both of them lead to the same action, it is more efficient to split them off because the cost of the first one is lower. Thus, we previously check the first one and, if it is not fulfilled, then the other one is tested.

As we have mentioned before, our goal is to obtain the D-basis from an arbitrary implicational system. In the first stage, the algorithm computes all minimal generators by means of the MinGen_0 method proposed in [5]. Then, in a second stage, it associates each minimal generator with all elements for which it is a cover. In this stage, we have calculated a subset of covers containing all minimal covers of a given attribute a. Finally, the algorithm removes those intermediate covers which are not minimal covers, obtaining the D-basis. This sequence of tasks is illustrated in Figure 1.

As Figure 1 depicts, the transition from stage 1 to stage 2 needs a way to associate the minimal generators with some of the elements in its closed set.

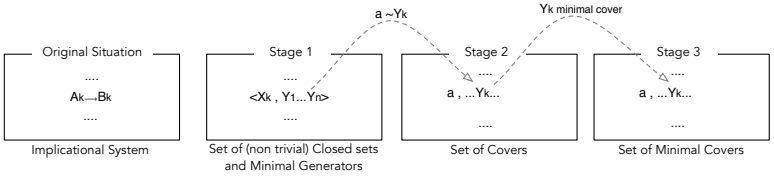

Fig. 1. Stages of D-basis algorithm

Thus, let a be an attribute and mg be the set of minimal generators such that its closure contains a. We write this association as a pair $\langle a, mg\rangle$. Let Φ be a set of such pairs of attributes with their generators. We define the Function `Add` which builds the set of covers produced in Stage 2 as follows:

$$\texttt{Add}(\langle a, mg\rangle, \Phi) = \{\langle a, \{g \in mg | a \notin g\} \cup \{mg_a\}\rangle : \langle a, mg_a\rangle \in \Phi\}$$

Then, in stage 3, the algorithm picks up the set of minimal covers from the set obtained in stage 2 using the Function `MinimalCovers`. The method ends with the Function `OrderedComp` which applies Composition Rule at the same time that it orders the implications in the following sense: the first implications in the D-basis are the binary ones (those with the left-hand side being a singleton).

Algorithm 1: D-basis

input : An implicational system Σ on M
output: The D-basis Σ_D on M
begin
  MinGen:=$\texttt{MinGen}_0(M, \Sigma)$
  C:= $\emptyset$
  foreach $\langle C, mg(C)\rangle \in MinGen$ **do**
    foreach $a \in C$ **do**
      C:=$\texttt{Add}(\langle a, mg(C)\rangle$, C)
  $\Sigma_D := \emptyset$
  foreach $\langle a, mg_a\rangle \in C$ **do**
    mg_a :=`MinimalCovers`(mg_a)
    foreach $g \in mg_a$ **do** $\Sigma_D := \Sigma_D \cup \{g \to a\}$;
  OrderedComp(Σ_D)
  return Σ_D

Example 5. Algorithm 1 returns the following D-basis from the input implicational system of Example 3:

$$\Sigma_D = \{a \to d, bce \to ad, ab \to ce, ae \to bc, bde \to ac, cd \to abe\}$$

We emphasize that although ac is a minimal generator, it is not a minimal cover, thus an implication with ac in the left-hand side is redundant (deduced from inference axioms) and hence should not appear in the D-basis.

A detailed illustrative example

In the conclusion of this section we show the execution of the method, in all its stages, on a set of implications from [3], which was used later to illustrate the D-basis definition in [1].

$$\Sigma = \{5 \to 4, 23 \to 4, 24 \to 3, 34 \to 2, 14 \to 235, 25 \to 134, 35 \to 124, 15 \to 24, 123 \to 45\}$$

As a first step in the algorithm, $\texttt{MinGen}_0$ renders the following set of pairs of closed sets and its non-trivial minimal generators, see Figure 2:

$$\{\langle 12345, \{123, 14, 15, 25, 35\}\rangle, \langle 234, \{23, 24, 34\}\rangle, \langle 45, \{5\}\rangle, \langle \emptyset, \emptyset\rangle\}$$

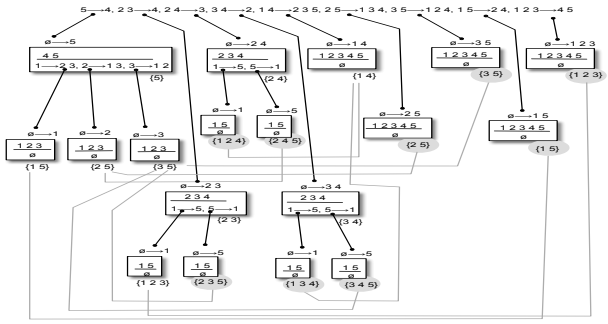

Fig. 2. $\texttt{MinGen}_0$ Execution

Then, for each closed set and each of its elements, our algorithm renders the following set of pairs of elements and covers:

$$\{\langle 1, \{25, 35\}\rangle, \langle 2, \{14, 15, 35, 34\}\rangle, \langle 3, \{14, 15, 25, 24\}\rangle,$$
$$\langle 4, \{123, 15, 25, 35, 5, 23\}\rangle, \langle 5, \{123, 14\}\rangle\}$$

For each element, the Function `MinimalCovers` picks up its minimal covers:

$$\{\langle 1, \{25, 35\}\rangle, \langle 2, \{14, 34\}\rangle, \langle 3, \{14, 24\}\rangle, \langle 4, \{5, 23\}\rangle, \langle 5, \{14, 123\}\rangle\}$$

Finally, at the last step, the algorithm turns these pairs into implications and applies ordered composition resulting in the D-basis.

$$\Sigma_D = \{5 \to 4, 23 \to 4, 24 \to 3, 34 \to 2, 14 \to 235, 25 \to 1, 35 \to 1, 123 \to 5\}$$

5 Conclusion and future works

In this work we have presented a way to obtain the D-basis from any implicational system. In [1] the algorithm was proposed to compute the D-basis from any direct basis, but the computation from any implicational system was left open. There exists also an efficient algorithm for the computation of the D-basis from the context using the method of finding the minimal transversals of the associated hypergraphs [2], but this assumes the different input for the closure system which is outside the scope of this paper.

The Function `MinimalCovers` renders the D-basis within the framework of the closure systems without the need of any transformation. A key point of our work is the connection between covers and generators. Using minimal generators, the D-basis is obtained by reducing the set of minimal generators and transforming it into a set of minimal covers.

As future work, we propose to develop an algorithm which computes the D-basis with better integration of the minimal generator computation to render the minimal covers in a more direct way. In addition, we are planning to design an empirical study and to make a comparison between this algorithm and other techniques proposed in previous papers.

Acknowledgment

Supported by Grants TIN2011-28084 and TIN2014-59471-P of the Science and Innovation Ministry of Spain.

References

1. K. Adaricheva and J. B. Nation and R. Rand, *Ordered direct implicational basis of a finite closure system*, Discrete Applied Mathematics, 161 (6): 707–723, 2013.
2. K. Adaricheva and J.B. Nation, *Discovery of the D-basis in binary tables based on hypergraph dualization*, `http://arxiv.org/abs/1504.02875`, 2015.
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. P. Cordero, A Mora, M. Enciso, I.Pérez de Guzmán, *SLFD Logic: Elimination of Data Redundancy in Knowledge Representation*, LNCS, 2527: 141–150, 2002.
5. P. Cordero, M. Enciso, A Mora, M. Ojeda-Aciego, *Computing Minimal Generators from Implications: a Logic-guided Approach*, CLA 2012: 187–198, 2012.
6. V. Duquenne, Some variations on Alan Day's Algorithm for Calculating Canonical Basis of Implications, CLA 2007: 192–207, 2007.
7. B. Ganter, *Two basic algorithms in concept analysis*, Technische Hochschule, Darmstadt, 1984.
8. A. Mora, M. Enciso, P. Cordero, I. Fortes, *Closure via functional dependence simplification*, International Journal of Computer Mathematics, 89(4): 510–526, 2012.
9. S. Rudolph, *Some Notes on Managing Closure Operators*, LNCS, 7278: 278–291, 2012.
10. M. Wild, The joy of implications, aka pure Horn functions: mainly a survey, `http://arxiv.org/abs/1411.6432`, 2014.