mathematics

MDPI

Article

Towards a Notion of Basis for Knowledge-Based Systems—Applications

Gonzalo A. Aranda-Corral [1], Joaquín Borrego-Díaz [2,*], Juan Galán-Páez [2] and Daniel Rodríguez-Chavarría [2]

[1] Department of Information Technology, Universidad de Huelva, 21004 Huelva, Spain; gonzalo.aranda@dti.uhu.es
[2] Department of Computer Science and Artificial Intelligence, Universidad de Sevilla, 41012 Sevilla, Spain; juangalan@us.es (J.G.-P.); danrodcha@gmail.com (D.R.-C.)
* Correspondence: jborrego@us.es

Abstract: In the paradigm of Knowledge-Based Systems (KBS), the design of methods to simplify the reasoning leads to more efficient processes. A point of view that provides valuable insights is the algebraic one. In this work, a notion of basis (and dimension) for Knowledge Bases in Propositional Logic associated with knowledge forgetting is introduced. It is based on ideas that come from the translation of such logic in (Computer) Algebra, particularly from the interpretation of *variable forgetting*. In this paper, the concept of *weak base* is defined as a set of variables sufficient to decide the consistency using variable forgetting. Several applications of weak bases are presented in order to show their usefulness in KBS reasoning and to justify their study and use in solving problems within this topic.

Keywords: knowledge-based systems; computer algebra; variable forgetting; conservative retraction

MSC: 03F20; 68T27; 68T30; 68T35; 03B70

Citation: Aranda-Corral, G.A.; Borrego-Díaz, J.; Galán-Páez, J.; Rodríguez-Chavarría, D. Towards a Notion of Basis for Knowledge-Based Systems—Applications. *Mathematics* **2021**, *9*, 252. https://doi.org/10.3390/math9030252

Academic Editor: Eugenio Roanes-Lozano
Received: 28 December 2020
Accepted: 25 January 2021
Published: 27 January 2021

1. Introduction

The symbolic paradigm in Artificial Intelligence (AI) maintains a relevant role in the development and research of the discipline. Along with the development of sub-symbolic approaches, symbolic ones are regaining momentum, mainly because they facilitate the achievement of Explainability and Trust [1]. Within the paradigm, Knowledge-Based Systems (KBS), as for instance, the expert systems and, in particular, the Rule-Based Expert Systems (RBES), are especially suitable to reach such achievements.

Another characteristic of some types of KBS is the possibility of being interpretable in Computer Algebra [2–7], which provides a plus of reliability, due to the mathematical specification of its properties. In particular, it is possible to address the need for Knowledge-Based Systems (KBS) specialization in order to particularize their application or to specialize them for certain contexts, as well as for its subsequent refinement (e.g., [8]). This need also encompasses the need for specializing the knowledge before prototyping (see e.g., [9,10]). In this type of task, it is necessary both to formalize and verify the methods in order to preserve the trust in them, for instance in diagnosis [7] or in applications derived from algebraic-based reasoning methods [2].

The modularity enjoyed by logic-based representation can facilitate not only the specialization of the knowledge. It also aids verifying the derived methods, such as partitions [11], contextualizations [7,12], or customizations [12] of the knowledge, as well as providing efficient representations for reasoning [13]. The aforementioned methods can be designed using Computer Algebra Systems (CAS) when the logic is interpretable in Algebra. The use of CAS leads to practical problem-solving in the field of Engineering and Science (see e.g., [10,14]). Another advantage of interpretation in CAS is that once the methods have been designed and interpreted, the specific algebraic resources needed to

perform them can be determined. Moreover, it is possible to program the algorithms, used in the solution, in general-purpose programming languages [2], using when necessary, libraries that calculate the specific algebraic techniques (e.g., [15]). This way the verification task does not depend on the correctness of third-party (opaque) software as some commercial CAS.

1.1. Methods for Specialization of KBS

Turning to the problem of the specialization of KBS, a way of approaching it would be to consider the application of divide-and-conquer strategies at the time of the creation of the KB (see e.g., [16] for micro-theories design). Or, if the KB is already built, to try distilling from it the part relevant to the particular problem (and *forgetting* the rest). Currently, the latter option has attracted extensive interest in the AI community. Some fairly complete studies on this subject, for several fragments of propositional or first-order logic, have been published (see e.g., [17] where authors explore the question, or [2] where the problem is addressed with algebraic tools). Approaches for different logics have been published recently (see e.g., [18,19] in the field of Semantic Web and Description Logics) as well as for agent theories, as in Situation Calculus [20].

Given a KB, K, the specialization of K would aim to reduce K, to other K' where $K \models K'$, and K' shares the language with the formula goal F (thus it is expected K' to be much smaller than K). After this, the problem $K' \models F$ is considered. A fairly complete analysis of such approach, called *Location Strategy*, can be found in [2].

To ensure that such strategy is sound and complete, K should be a *conservative extension* of the reduction K' itself (or, equivalently, K' a conservative retraction of K). That is, any formula in the language of K' entailed by K is also entailed by K' itself (i.e., K does not provide any knowledge expressed in the language of K' that is not already implied by K'). Conservative extensions have been deeply investigated in Mathematical Logic and Computer Science (see e.g., [21–23]), since they allow the formalization of several notions concerning refinements and modularity. However, the use of the dual notion (conservative retraction) is relatively less studied, due to its logical complexity.

Given a logic (propositional logic in our case) and a sublanguage $\mathcal{L}'$ of $\mathcal{L}(K)$ (being $\mathcal{L}(K)$ the language of K) a conservative retraction on the language $\mathcal{L}'$ always exists, namely:

Definition 1. *The **complete conservative retraction** (called canonical retraction in [2]) of K to $\mathcal{L}'$ is*

$$[K, \mathcal{L}'] = \{F \in Form(\mathcal{L}') \ : \ K \models F\}$$

From the definition itself, all conservative retractions of K on the same language are equivalent to $[K, \mathcal{L}']$. Thus, different syntactic representations (that is, axiomatizations) equivalent to $[K, \mathcal{L}']$ can be obtained by applying different methods. Note also that the complete retraction is an infinite KB, and therefore an adequate (finite) equivalent KB is needed. Lastly, please note that $[K, \mathcal{L}'] = Form(\mathcal{L}')$ if and only if the conservative retraction and thus the K itself- is inconsistent.

Example 1. *Let*

$$K_0 = \{p \to q, p \wedge q \leftrightarrow r\} \text{ and } \mathcal{L}' = \{q, r\}$$

Finding a finite axiomatization of $[K, \mathcal{L}']$ is not straightforward, since there is not an evident syntactic separation in the formulas of K between $\mathcal{L}'$ and the remainder of the language of K, $\{p\}$. In this case, it is not hard to prove that

$$[K, \mathcal{L}'] \equiv \{r \to q\}$$

The aforementioned Location Strategy can be then rewritten as follows [2]: Given K and $F \in Form(\mathcal{L}')$ with $\mathcal{L}' \subseteq \mathcal{L}$, the question $K \models F$? is answered by carrying out two tasks:

1. A (finite) axiomatization of $[K, \mathcal{L}']$ has to be computed.
2. The question $[K, \mathcal{L}'] \models F$? has to be answered (e.g., deciding the consistency of $[K, \mathcal{L}'] + \{\neg F\}$) by using the axiomatization from Step (1).

As it has already been pointed out, the interest of the strategy lies in the fact that the second task has lower complexity than the direct question. This is due to the relatively small size of $\mathcal{L}'$ (e.g., $\mathcal{L}'$ can be the set of symbols of $\mathcal{L}$ occurring in the goal F), as long as the axiomatization obtained is acceptable. For example, in [24] authors use a forecasting system with around a quarter of a million formulas on a language with 94 variables. Although there exist 2^{94} potential subsets of variables, in practice, only a few of them appear in the formula-goal or the deduction process. By reducing the KB to a conservative retraction on the language of the formula goal, we aim to reduce, in turn, the complexity of the inference process.

1.2. Forgetting

As Y. Moinard quoted in [25], the problem of *variable forgetting* in propositional logic was already raised by G. Boole in 1854. From there, it was studied both in both Mathematical Logic and in the logic-based AI itself (see e.g., [18,26–29] as well as [30] for the related problem of formula-variable independence). Variable forgetting (see [31] for a recent survey) is a widely studied technique in AI that has been used, among other applications, to update or refine (logical, rule-based or CSP) programs [25,32,33] (including Answer Set Programming [34,35]). The interest in variable forgetting techniques is not limited to classical (monotonous) logic. It also received attention in the field of non-monotonous reasoning. In the (epistemic) modal logics for (multi)agency, the technique is useful to represent knowledge-based games and nonmonotonic reasoning [18,29,36]. Besides, when reasoning under inconsistency, the use of variable forgetting allows weakening the KB to obtain consistent subKBs (by eliminating the variables involved in the inconsistency) [37]. Moreover, it is also useful for KB merging [38].

This article is based on the idea that when the forgetting process is carried out by progressively simplifying the language, the variables involved in forgetting could provide valuable information about the KB itself. The step-by-step process of variable forgetting allows removing one variable without losing the original knowledge that will be expressible in the reduced language. Our idea focuses mainly on consistent KBs. Also, it is possible to use variable forgetting techniques in inconsistency reasoning, for which there is also an extensive literature [39–41].

A problem related to the above-mentioned is how to decide which sub-languages would be the most suitable for computing the corresponding conservative retraction. It could be useful to know the relationship the variables of these sub-languages have, mainly if they are sufficient to ensure the reasoning completeness. In other words, if $[K, \mathcal{L}']$ only depends on $\mathcal{L}'$, how is the semantic relationship among variables of $\mathcal{L}'$ (in the models of K)? For example, is any variable redundant when deciding the entailment? What is the relationship with the consistency decision cost? Let us stop for a moment and discuss how ideas from Algebra could be used within this context.

1.3. Algebraic Insights for Forgetting

In [2], the authors exploit an algebraic inspiration of the process of forgetting variables in the case of classical propositional logic. Thinking of variables as if they were dimensions, that paper shows that variable forgetting can be interpreted as a projection of the algebraic variety, which represents the set of models of the starting KB. In this way, the consistency decision process would consist of projecting until reaching dimension 0, that is, no variable, reducing itself to a subset of $\{\top, \bot\}$.

Continuing with the analogy, one could think of translating the idea of the Krull dimension to the context of KB in Propositional Logic. That is, estimating the length of the sequences

$$\{\top\} \lhd K_1 \lhd \cdots \lhd K_n = K \tag{1}$$

where $K \lhd K'$ means that $K' \models K$ but $K \not\models K'$. Nevertheless, if one aims adapting the same idea to the forgetting variable paradigm, several changes should be made. The first and most important is that in order to minimize the cost of automated reasoning, one should search for *minimum length* sequences with the structure shown in the Expression 1, where K_i is obtained from K_{i+1} by one variable forgetting step. The fundamental reason is that the computational cost of each step is relevant (e.g., for solving SAT problems using variable forgetting). This problem will be studied in Section 6.

1.4. Aim and Structure of the Paper

The paper aims introducing the idea of weak bases as a tool to reduce the complexity of reasoning tasks, preserving correctness and completeness. The notion is inspired by the idea of projection in Algebraic Geometry, by identifying it with that of variable forgetting (Section 5). Some properties and applications will be also shown. The structure of the paper is as follows. The next section introduces the preliminaries used along the paper. It introduces the variable forgetting operators, as well as the associated operations to work with KBs. In Section 3, we show how the variable forgetting is used for SAT solving using saturation. In Section 4, a variable forgetting operator is introduced to be used in the examples. The concept of weak basis is introduced in Section 6 as minimal variable sets deciding the consistency of the KB (by variable forgetting). Finally, the last two sections present applications of weak bases and related and future work.

2. Preliminaries

2.1. Propositional Logic

A propositional language is a finite set $\mathcal{L} = \{p_1, \ldots, p_n\}$ of propositional symbols (also called propositional variables). The set of formulas $Form(\mathcal{L})$ is built up in the usual way, that is, by using the standard connectives $\neg, \wedge, \vee, \rightarrow$ and $\top$ ($\top$ denotes the constant *true*, and $\bot$ is $\neg\top$). Given two formulas F, G and $p \in \mathcal{L}$, we denote $F\{p/G\}$ the formula obtained after replacing every occurrence of p in F by the formula G.

A formula F is called **reduced** if $F \in \{\top, \bot\}$ or neither $\top$ nor $\bot$ appear in F. The set of reduced formulas is denoted by $Form^r(\mathcal{L})$.

In this paper, a KB is a finite set of formulas. The language of K is denoted by $\mathcal{L}(K)$ (i.e. the set of variables used in K).

An interpretation (or valuation) v is a function $v : \mathcal{L} \rightarrow \{0, 1\}$. It is denoted by $val(\mathcal{L})$ the set of valuations on $\mathcal{L}$. It is said that $v \in val(\mathcal{L})$ is a **model** of $F \in Form(\mathcal{L})$, $v \models F$ if it makes F true in the usual classical truth functional way. Analogously, it is said that v is a model of K ($v \models K$) if v is a model of every formula in K. The set of models of F is denoted by $Mod(F)$ (resp. $Mod(K)$ denotes the set of models of K).

A formula F (or a knowledge base K) is **consistent** if it exhibits at least one model. It is said that K entails F ($K \models F$) if every model of K is a model of F, that is, $Mod(K) \subseteq Mod(F)$. Both notions can be naturally generalized to KB instead of formulas, preserving the same notation. It is said that K and K' are equivalent, $K' \equiv K$, if $K \models K'$ and $K' \models K$. The same notation will also be used for the equivalence with (and between) formulas. The notation $K' \trianglelefteq K$ will also be used when $K \models K'$, and $K' \triangleleft K$ if $K \models K'$ but $K' \not\models K$.

Remark 1. *Any formula F is equivalent to a reduced formula. It suffices to apply the so-called* ***simplification operator****:*

$$\sigma : Form(\mathcal{L}) \rightarrow Form^r(\mathcal{L})$$

defined by

1. $\sigma(s) = s$ *if* $s \in \{\top, \bot\}$
2. $\sigma(\neg\top) = \bot$ *and* $\sigma(\neg\bot) = \top$,
3. $\sigma(F) = F$ *if* $\bot$, $\top$ *do not occur in* F.
4. *If* $\top$ *of* $\bot$ *occurs in* F:

(a) $\sigma(\top \wedge F) = \sigma(F), \sigma(\top \vee F) = \top$
(b) $\sigma(\bot \wedge F) = \bot$ *and* $\sigma(\bot \vee F) = \sigma(F)$
(c) $\sigma(\top \to F) = \sigma(F)$, $\sigma(F \to \top) = \top$, $\sigma(\bot \to F) = \top$ *and* $\sigma(F \to \bot) = \sigma(\neg\sigma(F))$
(d) *If* $F, G \neq \top, \bot$,

$$\sigma(F * G) = \sigma(\sigma(F) * \sigma(G))$$

for $* \in \{\wedge, \vee, \to\}$ *and* $\sigma(\neg F) = \sigma(\neg\sigma(F))$

It is straighforward to see that $\sigma(F) \equiv F$.

2.2. Conservative Retractions

Definition 2. *Let* K, K' *be two KBs. It is said that*

- K *is an* ***extension*** *of* K' *if* $\mathcal{L}(K') \subseteq \mathcal{L}(K)$ *and*

$$\forall F \in Form(\mathcal{L}(K'))[K' \models F \implies K \models F]$$

- K *is a* ***conservative extension*** *of* K' *(or* K' *is a* ***conservative retraction*** *of* K*) if it is an extension verifying that every consequence of* K *expressed in the language* $\mathcal{L}(K')$ *is already consequence of* K',

$$\forall F \in Form(\mathcal{L}(K'))[K \models F \implies K' \models F]$$

The second definition means that K extends K' but no new knowledge expressed by means of $\mathcal{L}(K')$ is entailed by K.

2.3. Complete Retraction Operator

The complete conservative retraction of Definition 1 can be understood as a map

$$[.,.] : 2^{Form(\mathcal{L})} \times 2^{\mathcal{L}} \to 2^{Form(\mathcal{L})}$$

$$(K, \mathcal{L}') \mapsto [K, \mathcal{L}']$$

which suggests defining a first variable forgetting operator.

Definition 3. *Let* $\mathcal{L}$ *be a propositional language.*

- *The* ***complete retraction operator*** *for* p *is defined as:*

$$\delta_p^c[.] : 2^{Form(\mathcal{L})} \to 2^{Form(\mathcal{L}\setminus\{p\})}$$

$$\delta_p^c[K] := [K, \mathcal{L} \setminus \{p\}]$$

- *Given* $\mathcal{L}' \subseteq L$ *it is defined*

$$\delta_{\mathcal{L}'}^c[K] := [K, \mathcal{L} \setminus \mathcal{L}']$$

Remark 2. *To make some proofs more readable, the following notation is used. Given* $\mathcal{L}_0 \subseteq \mathcal{L}$, $[K, \mathcal{L}_0]$ *is also denoted by* $K_{\restriction \mathcal{L}_0}$. *Likewise, given* $F \in Form(\mathcal{L})$, $F_{\restriction \mathcal{L}_0}$ *will be a formula equivalent to a conservative retraction of* $\{F\}$ *to* $\mathcal{L}_0$ *(that is,* $[\{F\}, \mathcal{L}_0]$*).*

Working with the conservative retraction provided by the complete operator has, on the one hand, the disadvantage that the KB obtained is infinite and it does not provide at first a finite equivalent KB. However, on the other hand, it has the advantage that managing all the consequences of the conservative retraction simplifies the demonstration of results of characterization of the entailment restricted to subsets of variables. To address the difficulty, the notion of *variable forgetting operator* is introduced in [2]. The idea is to adopt an algebraic poitn of view of the problem of knowledge restriction, and thus to be able to design new methods for computing conservative retraction.

2.4. Variable Forgetting Operators

In [2], a method for computing conservative retractions by means forgetting operators is presented. We generalize in this section some results of that paper to other operators. The operators that are of the type:

The operators considered in this section are of the type

$$\delta : Form(\mathcal{L}) \times Form(\mathcal{L}) \to Form(\mathcal{L})$$

The idea lies in to replace the complete operators be others which can be computed in practice (actually, which give a finite presentation of the result).

Definition 4. *It is said that two operators δ_1 and δ_2 are **equivalent** when they compute equivalent formulas, that is,*

$$\delta_1(F, G) \equiv \delta_2(F, G), \text{ whatever } F, G \in Form(\mathcal{L})$$

Definition 5. *It is said that an operator δ is*

- ***symmetric** if $\delta(F, G) \equiv \delta(G, F)$ whatever $F, G \in Form(\mathcal{L})$*
- ***sound** if $\{F, G\} \models \delta(F, G)$.*

We are interested in these operators useful for computing conservative retractions.

Definition 6. *$\delta_p : Form(\mathcal{L}) \times Form(\mathcal{L}) \to Form(\mathcal{L} \setminus \{p\})$ is a **forgetting operator** (f.o.) for the variable $p \in \mathcal{L}$ if for any $F, G \in Form(\mathcal{L})$*

$$\delta_p(F, G) \equiv [\{F, G\}, \mathcal{L} \setminus \{p\}]$$

Lemma 1.

1. *Any two f.o. for p are equivalent.*
2. *The forgetting operators are symmetric and sound.*

Remark 3. *Without loss of generality, it will be assumed that f.o. only return reduced formulas, since the simplification operator of Remark 1 can be used, considering the operator $\sigma \circ \delta_p$ instead of δ_p.*

The following lemma is useful as a characterization of f.o. Although it is proved in [28] for a particular forgetting operator, the result is valid for all f.o. (since they are equivalent).

Lemma 2 (Lifting Lemma [2])**.** *Let $v : \mathcal{L} \setminus \{p\} \to \{0, 1\}$ be a valuation, $F, G \in Form(\mathcal{L})$ and δ_p a f.o. for p. The following conditions are equivalent:*

1. *$v \models \delta_p(F, G)$*
2. *There exists a valuation $\hat{v} : \mathcal{L} \to \{0, 1\}$ extending v (that is, $\hat{v} \restriction_{\mathcal{L}\setminus\{p\}} = v$) such that $\hat{v} \models F \wedge G$.*

The extension of f.o. to be applied to KBs is defined as follows:

Definition 7. *Any $\delta_p(.,.)$ f.o. is extended to an operator on KBs,*

$$\delta_p[.] : 2^{Form(\mathcal{L})} \to 2^{Form(\mathcal{L})}$$

defined as

$$\delta_p[K] := \{\delta_p(F, G) \ : \ F, G \in K\}$$

The following result is from [2]. It shows how a conservative retraction (the result of forgetting one single variable) can be obtained.

Proposition 1 ([2]). $\delta_p[K] \equiv [K, \mathcal{L} \setminus \{p\}]$

Thus, Lifting Lemma is also valid for the operator $\delta_p[.]$

Example 2. *Consider K_0 from Example 1 and $\mathcal{L}' = \{q, r\}$.*
Given a f.o. for p, it has that

$$\delta_p[K] \equiv \{r \to q\}$$

For example $v = \{(q,1),(r,0)\} \models [K, \mathcal{L}']$, and therefore it has an extension which is a model of K_0, for example $\hat{v} = v \cup \{(p,0)\}$. However, the countermodel of K_0 given by $v' = \{\{(q,0),(r,1)\}$ has not any extension model of K_0.

3. Saturation

Suppose from now on a f.o. is fixed for each variable of the language $\mathcal{L}$,

$$\Delta_{\mathcal{L}} = \{\delta_p \ : \ p \in \mathcal{L}\}$$

(it will also simply write Δ if the language is fixed).

Definition 8. *The process of successively applying the operators $\delta_p[\cdot]$ (in some order) for all the propositional variables $p \in \mathcal{L}(K)$ will be called* **saturation** *of K. It is denoted the result by $sat_\Delta(K)$.*

Please note that $sat_\Delta(K) \subseteq \{\bot, \top\}$, since only reduced formulas are produced;

$$[K, \emptyset] \equiv sat_\Delta(K) \subseteq \{\top, \bot\}$$

Thus, the result does not depend on the order of applications of operators. Moreover, since forgetting operators are sound, so if K is consistent then necessarily $sat_\Delta(K) = \{\top\}$.

Theorem 1 ([2]). *Suppose that the f.o. only produces reduced formulas. Then K is inconsistent if and only if $\bot \in sat_\Delta(K)$ (or equivalently, K is consistent iff $sat_\Delta(K) \subseteq \{\top\}$).*

Therefore the use of f.o. not only provide a method for computing conservative retractions, but also provides an algorithmic solution of (not necessarily clausal) SAT problems (hence for the entailment problem).

The operator $\delta_p[.]$ can be extended for linear ordered (l.o) variable sets (Or on $\mathcal{L}(K)$, to induce an ordered application of f.o.) $Q \subseteq \mathcal{L}$, $q_1 < \cdots < q_k$ as the operator

$$\delta_{Q,<} := \delta_{q_k} \circ \cdots \circ \delta_{q_1}[.]$$

If we dispense with making an order explicit, the operator is well-defined module logical equivalence. That is, any two orders on Q produce equivalent KBs. This is true since for each $p, q \in \mathcal{L}$, from Proposition 1 it follows that

$$\delta_p \circ \delta_q[K] \equiv \delta_q \circ \delta_p[K]$$

(both are equivalent to $[K, \mathcal{L} \setminus \{p, q\}]$). Thus, given $Q = \{q_1, \ldots, q_k\} \subset \mathcal{L}$, the operator δ_Q (without explicit order) is a well defined modulo logical equivalence. Thus, it holds

$$\delta_Q[K] \equiv [K, \mathcal{L} \setminus Q]$$

A useful feature of the use of conservative retractions is that it allows reducing the problem in a way that only variables of the goal formula are considered, forgetting the rest of variables. Such property is called the *Location Principle* (the second property of the following result).

Proposition 2.

1. *If* $K \models F$ *then* $\delta_p[K] \models \delta_p(F, F)$.
2. ***Location principle**:*
$$K \models F \iff \delta_{\mathcal{L}\setminus var(F)}[K] \models F$$
3. *If* $\delta_Q[K] \models \neg\delta_Q[\{\neg F\}]$ *then* $\delta_Q[K] \models F$.

Proof.

(1): it is true because $\delta_p(G, G) \in [K, \mathcal{L} \setminus \{p\}] \equiv \delta_p[K]$.
(2): $K \models F$ if and only if $F \in [K, var(F)]$, which is equivalent to $\delta_{\mathcal{L}\setminus var(F)}[K]$ by Proposition 1.
(3): Let v be a valuation on $\mathcal{L}$ such that $v \models \delta_Q[K]$. Then, by hypothesis

$$v \restriction_{\mathcal{L}\setminus Q} \models \neg\delta_Q[\{\neg F\}]$$

so there exists no extension of $v \restriction_{\mathcal{L}\setminus Q}$ to the full language that models $\neg F$. Since v itself extends $v \restriction_{\mathcal{L}\setminus Q}$, then $v \models F$ is satisfied.

□

4. Canonical Forgetting Operator

This section defines an o.f. that can be used in the examples.

Definition 9 ([2]). *The **canonical forgetting operator** for p is defined as*

$$\delta^0_p = \sigma \circ \delta^*_p$$

where σ is the simplification operator defined in Remark 1 and

$$\delta^*_p(F, G) := (F \wedge G)\{p/\top\} \vee (F \wedge G)\{p/\bot\}$$

It is straightforward to see (by Lifting Lemma) that δ^0_p is a forgetting operator for p, and therefore, it is equivalent to any other one.

Remark 4. *If no specific f.o. is mentioned, the canonical one will be used in the examples provided throughout the article. An implementation in Haskell of this operator can be found in https://github.com/DanielRodCha/SAT-canonical.*

Example 3. *Let* $F = p \to q$ *and* $G = p \wedge r \to \neg q$.

$$\begin{aligned}
\delta^0_p(p \to q, p \wedge r \to \neg q) = & \\
&= \sigma(\delta^*_p(p \to q, p \wedge r \to \neg q)) = \\
&= \sigma([(p \to q) \wedge (p \wedge r \to \neg q)]\{p/\top\} \vee \\
&\qquad [(p \to q) \wedge (p \wedge r \to \neg q)]\{p/\bot\}) = \ldots \\
&= \sigma([(\top \to q) \wedge (\top \wedge r \to \neg q)] \vee \\
&\qquad [(\bot \to q) \wedge (\bot \wedge r \to \neg q)]) = \\
&= \sigma(\sigma[(\top \to q) \wedge (\top \wedge r \to \neg q)] \vee \\
&\qquad \sigma[(\bot \to q) \wedge (\bot \wedge r \to \neg q)]) = \ldots \\
\cdots &= \sigma([q \wedge (r \to \neg q)] \vee \top) = \top
\end{aligned}$$

so any valuation over q, r can be extended to a model of $\{F, G\}$ (by Lifting Lemma). Using the Haskell implementation:

```
> canonical "p" (Impl (Atom "p") (Atom "q")) (Impl (Conj (Atom "p")
                (Atom "r")) (Neg (Atom "q")))
> True
```

Example 4. *The canonical f.o. allows computing formula retraction. For example, let be*

$$F = \neg p \wedge (q \to r) \text{ and } Q = \{q, r\}$$

A formula $F_{\restriction_Q}$ can be computed with $\delta_p^0(F, F)$, because $\delta_p^0(F, F) \equiv [\{F\}, \{q, r\}]$;

$$\begin{aligned}\delta_p^0(F, F) \quad &= \sigma[((\neg\top \wedge (q \to r)) \wedge (\neg\top \wedge (q \to r)))\vee \\ &\quad \vee((\neg\bot \wedge (q \to r)) \wedge (\neg\bot \wedge (q \to r)))] = ... = ((q \to r) \wedge (q \to r))\end{aligned}$$

Therefore the formula $q \to r$ also can be used as $F_{\restriction_Q}$. By using Haskell implementation of canonical f.o.:

```
> forgetVarKB "p" F
> fromList [((q -> r) & (q -> r))]
```

5. Semantic (Krull) Dimension for Knowledge Bases

The notion of dimension is ubiquitous in Mathematics and its applications (e.g., Computer Science). The idea is to estimate how complex is the stratification of a KB based on forgetting. It bears some resemblance to the *Krull dimension* on Algebraic Geometry, but since the motivation is different, it is necessary to modify the idea. In order to motivate the need for modification, let us examine how the straight translation would be:

Definition 10. *The Krull dimension of K a consistent KB, $dim^k(K)$, is the maximum n of the length of chains*

$$\{\top\} \lhd K_1 \lhd \cdots \lhd K_n \equiv K$$

Example 5. *Let $K = \{p \vee q, p \vee \neg q, \neg p \vee q\}$. It is not hard to see that*

$$\{\top\} \lhd \{p \vee q\} \lhd \{p \vee q, p \vee \neg q\} \lhd K$$

is a chain of maximum length. The reason is that if $K \lhd K'$ then $Mod(K') \subset Mod(K)$ (proper subset) so $|Mod(K')| < |Mod(K)|$. Therefore, any sequence of KBs would have a maximum length of 3, because $|Mod(\top)| = 4$ and $|Mod(K)| = 1$.

The Krull dimension for KB is not useful for our pourposes since it only estimates the number of countermodels of K.

Proposition 3. *Let K be consistent. Then*

$$dim^k(K) = 2^{|\mathcal{L}(K)|} - |Mod(K)|$$

Proof. For each $v \in val(\mathcal{L})$, let F_v be a propositional formula such that $Mod(F_v) = \{v\}$. Suppose that

$$val(\mathcal{L}(K)) \setminus Mod(K) = \{v_1, \ldots, v_m\}$$

Considering the sequence

$$\{\top\} \equiv \{\bigvee_{i=1}^{m} F_{v_i} \vee \bigvee_{v \models K} F_v\} \lhd \{\bigvee_{i=1}^{m-1} F_{v_i} \vee \bigvee_{v \models K} F_v\} \lhd \cdots \lhd \{F_{v_1} \vee \bigvee_{v \models K} F_v\} \lhd \{\bigvee_{v \models K} F_v\} \equiv K$$

Since in each step, the number of models is reduced in one unit, the sequence is of maximum length. Therefore

$$dim^k(K) = m = 2^{|\mathcal{L}(K)|} - |Mod(K)|$$

□

Therefore, the Krull dimension so defined would be semantic in nature, and it does not gather the idea of understanding the relationship among variables within a deductive process. We concern here in estimating how many basic conservative retractions (computed using δ_p) are necessary to decide consistency. This idea would correspond to a chain induced by f.o. in each step and with a minimum length, in order to optimize the number of operations.

6. Weak Basis

6.1. Dimension and Weak Basis

As it was already mentioned, although all the ordered applications of f.o. associated with variables of a set Q produce equivalent theories, their syntactic representations may differ. Therefore, it may happen that with some order we detect consistency (the KB resulting is $\{\top\}$) or inconsistency (the formula $\bot$ appears) before completing the saturation procedure, or that it could have been done better—with fewer variable forgetting steps—if we had chosen another order on the variables. Minimum sets that decide the consistency should be analysed. Let us see first an illustrative example.

Example 6. *Consider the variable set $Q = \{p, t\}$ for the KB*

$$K_e = \begin{cases} t \wedge p \to s, \\ t \wedge q \to s, \\ t \wedge r \to s, \\ p \wedge q \wedge t \to r \end{cases}$$

Using the Haskell implementation discussed in Remark 4, it is checked that

- $\delta^0_t[K] = \{\top\}$

```
> forgetVarKB "t" Ke
> fromList [True]
```

However,

- $\delta^0_p[K] \neq \{\top\}$ *because* $\delta^0_p(t \wedge p \to s, t \wedge p \to s) \neq \top$;

```
> forgetVarKB "p" Ke
> fromList [True, ((t & q) -> s),((t & r) -> s)]
```

and lastly, by applying δ^0_t,

$$\delta^0_t[\delta^0_p[K]] = \{\top\}$$

Therefore, if we apply the operators according to the order $t < p$, it would produce the answer consistent *in only one retraction step (it only needs to apply $\delta_t[.]$)), whereas the order $p < t$ needs to use the two f.o. to obtain the answer.*

The above example illustrates the issue of how to find a minimal set of variables such that through the (ordered) application of the corresponding f.o., consistency of the KB is decided. It will use f.o. other than the complete ones, since the latter have no algorithmic interest (it handles infinite sets).

When $\delta^c_Q[K]$ is inconsistent (and therefore also K), $\bot$ has to appear in it since it contains all the consequences. Therefore with a single application, inconsistency would be detected. On the other hand, in order to decide the consistency, it is necessary to check whether it is equivalent to $\top$, and therefore it would require checking that all the formulas of the complete retraction are tautologies.

Let $p_1 < q_2 < \cdots < p_d$ be a linear order (l.o.) on Q. In order to specify application order of forgetting operator, the application of the operator according to $<$ will be denoted by $\delta_{Q,<}[K]$, i.e.,

$$\delta_{Q,<}[K] := \delta_{p_d} \circ \cdots \circ \delta_{p_1}[K]$$

The notion of *weak basis* is that of a minimal set of variables that decides the consistency of K by executing its associated set of f.o. in some order

Definition 11. *Let K be a KB, $Q \subseteq \mathcal{L}(K)$ and $\Delta = \{\delta_p \ : \ p \in \mathcal{L}(K)\}$ a family of f.o. It is said that*

1. *K is **decided** if $\bot \in K$ or $K = \{\top\}$.*
2. *$Q \subseteq \mathcal{L}(K)$ Δ-**decides** K if $\delta_{Q,<}[K]$ is decided for some linear order $<$ on Q.*
3. *Q is a **weak basis** with respect to Δ (a Δ-w.b.) for K if Q decides K and none of its proper subsets does.*
4. *The **Δ-dimension** of K is defined as*

$$dim^{\Delta}(K) := min\{|Q| \ : \ Q \text{ is a weak basis for } K \text{ with respect to } \Delta\}$$

5. *Q is a Δ-**basis** if it is a w.b with $|Q| = dim^{\Delta}(K)$.*

If the family Δ is fixed, any reference of it in the notation will be omitted.

Example 7. Consider K_e, the KB from Example 6.

- The set $\{p, t\}$ decides K_e. However it is not a w.b. because the set $\{t\}$ also decides K. Thus, $dim^{\Delta}(K) = 1$.
- The set $\{p, q\}$ is a w.b. but not a basis.

By completeness of forgetting operators (Theorem 1) weak bases exist. From the same theorem, a characterization of consistency using w.b. can be proved:

Proposition 4. *K is inconsistent iff there exists Q a w.b. such that $\bot \in \delta_Q[K]$.*

Proof. Suppose K is inconsistent. Since the language $\mathcal{L}(K)$ itself decides K,

$$\bot \in sat_{\Delta}(K) = \delta_{\mathcal{L}(K)}[K]$$

then there exists a minimal subset satisfying the same property, i.e., a w.b. The other implication is consequence of Theorem 1. □

In the Github repository mentioned in Remark 4 an implementation of the algorithm to find all weak bases is also provided. It will be used later in some examples.

6.2. Estimating the Number of Variable Forgetting Steps to Decide the Entailment

An interesting property is that if a variable p *reduces* knowledge in the following sense

Definition 12. *A variable p* **reduces** *K if $dim^{\Delta}(\delta_p[K]) < dim^{\Delta}(K)$.*

The following result shows that if the application of a f.o. effectively reduces the represented knowledge, then it only reduces the number of f.o. needed to produce the answer in one unit.

Proposition 5. *If p reduces K then*

$$dim^{\Delta}(\delta_p[K]) = dim^{\Delta}(K) - 1$$

Proof. Let $d = dim^{\Delta}(K) - 1$. By reductio ad absurdum, suppose that

$$dim^{\Delta}(\delta_p[K]) = l < d$$

Let Q_0 be a basis for $\delta_p[K]$ (so $p \notin Q_0$). Then the set $Q_0 \cup \{p\}$ decides K, but its size is less than the Δ-dimension,

$$|Q_0 \cup \{p\}| = l + 1 < d + 1 = dim^{\Delta}(K)$$

which is a contradiction. □

We are already in a position to demonstrate that the *Delta*-dimension satisfies the idea that was discussed in Section 5.

Corollary 1. *Let K be a consistent KB. $dim_{\Delta}(K)$ is the minimun lenght of the sequences*

$$K_0 = \{\top\} \lhd_w K_1 \lhd_w \cdots \lhd_w K_n = K$$

where $K \lhd_w K'$ means that K is the result of applying a f.o. to K' ($i < n$).

Proof. Any linear ordered set $Q = \{q_d < \cdots < q_1\}$ satisfying

$$K_0 = \{\top\} \lhd_w K_1 \lhd_w \cdots \lhd_w K_n = K \text{ with } K_{i-1} = \delta_{q_i}[K_i]$$

decides K, thus the Δ-dimension is the shortest of such kind of chains. □

The study of the (in)consistency tests associated with an operator is useful to obtain lower levels of complexity, for example for SAT solvers (see e.g., [42]). In our framework, and thinking about the decision of consistency through f.o., the saturation process itself induces an algorithm (Algorithm 1), which stops under two situations: when it obtains $\bot$ or ends up reducing all to $\{\top\}$.

Algorithm 1: Saturation-based algorithm driven by the selection function *Select*(.).

Input: $K, Select(.), \Delta$;; *Select* : $2^{\mathcal{L}} \to \mathcal{L}$ *such that* $Select(X) \in X$
$\mathcal{L} \leftarrow \mathcal{L}_0$
$K_0 \leftarrow K$
While $K_0 \neq \{\top\}$ or $\bot \notin K_0$ **do**
  $p \leftarrow Select(\mathcal{L}_0)$
  $\mathcal{L}_0 \leftarrow \mathcal{L}_0 \setminus \{p\}$
  $K_0 \leftarrow \delta_p[K_0]$
od
If $\bot \in K_0$ **then Return** INCONSISTENT
  else Return CONSISTENT

Regarding the problem of $K \models F$? (with K consistent) employing the saturation of $K \cup \{\neg F\}$, the consistency case would be the worst in time complexity because it needs to forget all the variables. Therefore, the Δ-dimension could be a lower bound of the number of applications of f.o. in worst-case complexity, since in the process of saturation of $K \cup \{\neg F\}$, K itself is reduced to $\{\top\}$. Thus, the number of forgetting operators to be applied, $t_{\Delta}(K)$, satisfies

$$t_{\Delta}(K \cup \{\neg F\}) \geq dim^{\Delta}(K)$$

Likewise, it is possible to estimate, from the dimension, a lower bound of the number $t_{\Delta}(K)$ of f.o. to be applied for any saturation-based algoriTheorem

Theorem 2. *Suppose that $t_{\Delta}(K)$ is the number of applications of f.o. of an algorithm based on saturation. If K is consistent then*

$$t_{\Delta}(K \cup \{\neg F\}) \geq |\mathcal{L}(K)| - \log |Mod(K)|$$

Proof. Let $\mathcal{L} = \mathcal{L}(K)$. Since the worst case is when $K + \{\neg F\}$ is consistent

$$t_\Delta(K + \neg F) \geq dim^\Delta(K)$$

Let Q be a Δ-basis. Since $\delta_Q[K] = \{\top\}$, by Lifting Lemma for each $v \in val(\mathcal{L} \setminus Q)$ it can be selected an extension $\hat{v} \in val(\mathcal{L})$ verifying that $\hat{v} \models K$. Therefore, it can be defined a map

$$\begin{array}{c} val(\mathcal{L} \setminus Q) \to Mod(K) \\ v \mapsto \hat{v} \end{array}$$

that is injective, hence

$$|val(\mathcal{L} \setminus Q)| = 2^{|\mathcal{L} \setminus Q|} \leq |Mod(K)| \tag{2}$$

Taking logarithm, and since $|Q| = dim^\Delta(K)$;

$$\log |Mod(K)| \geq |\mathcal{L}| - dim^\Delta(K)$$

Therefore

$$t_\Delta(K + \{\neg F\}) \geq dim^\Delta(K) \geq |\mathcal{L}| - \log |Mod(K)|$$

what the theorem proves. □

It is worth noting that the dimension is related to the minimum number of f.o. applications needed to answer the question of (in)consistency, thus it is not just a pure semantic notion. For instance, if there is $F \in K$ unsatisfiable (or the whole KB is) but $\bot \notin K$, then $dim^\Delta(K) \geq 1$, since it must apply at least one f.o.

6.3. Entailment and Weak Basis

The idea behind the w.b. is to decide whether $K \models F$ as efficiently as possible using a minimum set of f.o., which already decides K itself (as is needed to retract it to $\{\top\}$). This idea is also inspired by the following result, which shows a necessary condition for entailment based on a conservative retraction of the goal formula, a much simpler computation as it avoids using the KB itself.

Proposition 6. *Let K be consistent and Q a w.b. for K.*

$$K \models F \quad \Longrightarrow \quad F_{\restriction \mathcal{L} \setminus Q} \equiv \top$$

Proof. Since Q is a w.b. for K, it has $\delta_Q[K] = \{\top\}$. By Proposition 2-(1) it holds that

$$\{\top\} \models \delta_Q[\{F\}]$$

therefore $F_{\restriction \mathcal{L} \setminus Q} \equiv \delta_Q[\{F\}]$ is a tautology. □

The previous proposition may be interpreted as that a KB does not provide specific information about the relationship among the variables outside the w.b., since every entailed formula does not contain such information (as $F_{\restriction \mathcal{L} \setminus Q}$ is a tautology).

Please note that the above result is only a necessary condition. For example, for $K = \{p\}$ and $F = \neg p$ the only weak basis is $\{p\}$. It $F = \neg p$, then

$$\delta_{\{p\}}[K] = \delta_p[\{F\}] = \{\top\}$$

but $K \not\models F$. A sufficient condition involves all the w.b. It reduces the problem to apply the saturation method to $K + \{\neg F\}$ only for w.b.

Theorem 3. *If $\bot \in \delta_Q[K \cup \{\neg F\}]$ holds for every Q w.b. of K, then $K \models F$.*

Proof. Assume this is not true, $K \not\models F$. Then $K \cup \{\neg F\}$ is consistent. Then there exists $Q_0 \subseteq \mathcal{L}$ be a variable set which decides $K \cup \{\neg F\}$, i.e.,

$$\delta_{Q_0}[K \cup \{\neg F\}] = \{\top\}$$

In particular, Q_0 decides K itself (since $\delta_Q[K]$ is contained in that set). Therefore there exists $Q \subseteq Q_0$ a w.b. for K. Then

$$\delta_{Q_0}[K \cup \{\neg F\}] = \{\top\} \equiv \delta_{Q_0 \setminus Q}[\delta_Q[K \cup \{\neg F\}]]$$

Which is a contradiction, because $\bot \in \delta_Q[K \cup \{\neg F\}]$ by hypothesis. □

Although having all the weak bases may be useful, the cost of computing all of them can be higher since the task involves the repeated application of retraction operators on (sub)knowledge bases. For example, it is easy to see that a 3-SAT instance composed of n clauses that do not share variables, has 3^n weak bases. Therefore, it can be concluded that its computation is worthwhile when the KB remains without revisions for an acceptable amount of time (this way, the weak basis could be reused) and for medium-sized Expert Systems (as [5,7]). However, it is possible to use weaker conditions in specific cases. The next section is devoted to show several case studies.

7. Case Studies and Applications

This section aims to illustrate the results and applications of weak bases with examples and case studies. Please note that for some of the examples it could be necessary to use more sophisticated f.o. than δ_p^0, due to the size of the KB. In [2] authors show some experimental advantages of such an operator. The one introduced in the mentioned paper uses the well-known translation of propositional formulas to polynomials on characteristics 2.

7.1. Case Study 1: Weak Basis and Contexts

A case of use where w.b. are useful is some cases in context-based reasoning. An approach to specify a context is based on determining which set of variables $\mathbf{c} \subseteq \mathcal{L}$ provides non-trivial information on this. That is, they are relevant variables for representing the state in this context (for example, those which represent signals/perceptions received by the agent). Thus, a context is identified with a sublanguage $\mathbf{c}$ of $\mathcal{L}$, and a *state* of this is will be specified by the truth value on that sublanguage.

Definition 13. *It is said that K is* **sound for a context $\mathbf{c}$** *if $\mathbf{c}$ is contained in a weak basis of K.*

The idea is that it can reason in the context using retraction with respect to a w.b. that we already know optimizes reasoning with the K itself. Thus, it is worth getting the corresponding conservative retraction since it is often expected to be reused. Formally:

Theorem 4. *Let K be sound for $\mathbf{c}$, and let Q be a w.b. with $\mathbf{c} \subseteq Q$. Let $F \in Form(\mathbf{c})$*

$$K \models F \iff \bot \in \delta_Q[K_{\restriction Q} + \neg F]$$

Proof.

$$\begin{array}{lll} K \models F & \iff & K_{\restriction Q} \models F \qquad [\textit{By Loc. Principle 2}] \\ & \iff & K_{\restriction Q} + \neg F \text{ inconsistent} \\ & \iff & \bot \in \delta_Q[K_{\restriction Q} + \neg F] \qquad [\textit{By Theorem 1, because } \delta_Q[.] \text{ is a saturation here}] \end{array}$$

□

Therefore, $K_{\restriction Q}$, being Q a variable set that decides K and $\mathbf{c} \subseteq Q$, is useful to reason in the context. Selecting a w.b. ensures the minimization of the reasoning process. However, when $\mathbf{c} \not\subseteq Q$, $\mathbf{c} \cup Q$ can be selected as it also decides K (because Q itself already does).

7.2. Case Study 2: Knowledge Bases with Disjoint Weak Basis

The particular case in which the KB has some disjoint w.b. is one in which the reasoning process is drastically shortened in some cases.

Theorem 5. *Let K be consistent and F be a formula such that $var(F) \cap Q = \emptyset$ for some Q w.b. of K*

$$K \models F \iff F \text{ is a tautology}$$

Proof. If $K \models F$ then by Proposition 2(2) $[K, var(F)] \models F$ holds.

In this case we have that:

$$\begin{aligned} [K, var(F)] &\equiv \delta_{\mathcal{L}\setminus var(F)}[K] \equiv \\ &\equiv \delta_{\mathcal{L}\setminus (var(F)\cup Q)}(\delta_Q[K]) \equiv \\ &\equiv \delta_{\mathcal{L}\setminus (var(F)\cup Q)}\{\top\}) = \qquad \text{[Because } Q \text{ is a w.b.]} \\ &= \{\top\} \end{aligned}$$

Therefore $\{\top\} \models F$, hence F is a tautology. □

According to Theorem 5, only formulas that share variables with *all* the w.b. should be specifically handled, since for the others it is sufficient to check whether it is a tautology. Such checking is relatively simple to verify since the goal formulas are usually significantly smaller and with much fewer variables than the full KB (as it usually happens in the *tell-ask* paradigm of expert systems).

Example 8. *The following example uses a KB composed of Horn clauses. Although no specific f.o. will be used for this case, more efficient f.o. than the canonical ones can be used, for example, those designed for implications in [43]. Let*

$$K = \begin{cases} t \\ p \wedge t \rightarrow s \\ t \wedge q \rightarrow s \\ t \wedge r \rightarrow s \\ p \wedge q \wedge t \rightarrow r \end{cases}$$

By using the program:

```
> weakBasis k
  [["t","s","r"],["t","s","q"],["t","p","q","r"],
    ["t","p","s"]]
```

Therefore

- $Q_1 = \{t, s, r\}$ *and* $[K, Q_1] = \{r \rightarrow s\}$
- $Q_2 = \{t, s, q\}$ *and* $[K, Q_2] = \{q \rightarrow s\}$
- $Q_3 = \{t, p, q, r\}$ *and* $[K, Q_3] = \{p \wedge q \rightarrow r\}$
- $Q_4 = \{t, p, s\}$ *and* $[K, Q_4] = \{p \rightarrow s\}$

Given a formula, it is interesting to check a priori *whether $var(F)$ intersects each w.b. For example, for $F = p \rightarrow q$ it has $K \not\models F$ since $var(F) \cap Q_1 = \emptyset$ but F is not a tautology.*

It is possible to reformulate the proof of Theorem 5 in order to obtain a significant result on the conservative retraction to a w.b. which is disjoint with others.

Proposition 7. *Let K be consistent and Q a w.b. such that there exists other w.b. disjoint with it. Then*

$$[K, Q] \equiv \{\top\}$$

Proof. Let Q_1, Q_2 be two disjoint w.b. Then it has:

$$\begin{array}{lll} [K, Q_1] \equiv \delta_{\mathcal{L}\setminus Q_1}[K] & \equiv \delta_{\mathcal{L}\setminus(Q_1\cup Q_2)}(\delta_{Q_2}[K]) & [Q_2 \subseteq \mathcal{L}\setminus Q_1] \\ & \equiv \delta_{\mathcal{L}\setminus(Q_1\cup Q_2)}(\{\top\}) = & [Q_2 \text{ is a w.b.}] \\ & = \{\top\} & \end{array}$$

□

Example 9. *Consider the KB*

$$K = \begin{cases} p \to s \\ t \wedge q \to s \\ t \wedge r \to s \\ t \wedge p \to s \\ p \wedge q \wedge t \to r \end{cases}$$

```
> weakBasis K
> Weak Basis of instance of K are:
[["s","r"],["s","q"],["s","t"],["p","q","r"],
["p","t"],["p","s"]]
```

K has the following w.b.:

$$Q_1 = \{s, r\},\ Q_2 = \{s, q\},\ Q_3 = \{s, t\},\ Q_4 = \{p, q, r\},\ Q_5 = \{p, t\},\ Q_6 = \{p, s\}$$

Since $Q_1 \cap Q_5 = \emptyset$, the reasoning can be simplified in some cases.
For example, let $F = p \to t$. It verifies $F_{\restriction Q_5} = F$ and $[K, Q_5] = \top$ (since $Q_1 \cap Q_5 = \emptyset$). Thus, $K \not\models F$.

Example 10. *The w.b. of K_e (Example 6) are:*

$$Q_1 = \{p, q, r\},\ Q_2 = \{q, s\},\ Q_3 = \{r, s\},\ Q_4 = \{t\}$$

Let us see some properties:

- *Note that for each w.b. there is another w.b. that is disjoint with this one, thus*

$$[K, Q_i] = \{\top\} (1 \leq i \leq 4)$$

Thus, it can apply Proposition 7:

  - *If $K \models F$ for $F \in Form(\{p, q, r, s, t\})$ not tautology, then $var(F) \cap Q_i \neq \emptyset$* for any *$i$.*
  - *It has that K_e is sound for the context $\mathbf{c}_1 = \{r, s\} \subseteq Q_3$. Let us consider*

$$G = s \to r, \text{ and } H = t \to r \wedge s$$

    - *G is not entailed by K_e, since $var(G) \cap Q_4 = \emptyset$ (that is, $G_{\restriction Q_4} = G$ is not a tautology whilst $[K, Q_4] \equiv \top$).*
    - *Since $H_{\restriction Q_1} = t \to s$ and $[K, Q_1] = \top$, then $K \not\models H$.*

The KB is sound for the context $\mathbf{c} = \{p, q, r\} \subseteq Q_1$. Let $F \in Form(\mathbf{c})$. Then, by Theorem 4

$$K \models F \iff \bot \in \delta_{Q_1}[K_{\restriction Q_1} + \neg F]$$

but, since $K_{\restriction Q_1} \equiv \top$ by Theorem 7 it has that $K \models F$ iff $\bot \in \delta_{Q_1}[\neg F]$. It is therefore sufficient to check whether the formula $\delta_{Q_1}[\neg F]$ is inconsistent.

7.3. Case Study 3: Knowledge Harnessing and Weak Basis

The work [44] introduces the notion of Knowledge Harnessing (KH). The problem of KH poses how to extract valid information about a specific context from conflicting or uncertain information received by a system (or agent). With this aim, forgetting variable operators are used to both characterize the problem from the logical point of view and provide a theoretical solution as an acceptance-rejection problem.

Definition 14 ([44])**.**

- *A language* $\mathcal{L}_0$ *is* **informative** *for* F *if* $F_{\restriction \mathcal{L}_0} \not\equiv \top$.
- *Let* K *be a KB,* F *be a formula and* $\mathcal{L}_0 \subseteq \mathcal{L}$ *a sublanguage. It is said that* F *contains* **harnessed** $\mathcal{L}_0$**-knowledge** *with respect to* K*, (notation:* $K \models_{\mathcal{L}_0} F$*) if the following two conditions hold*
  - $\mathcal{L}_0$ *is informative for* F*, and*
  - $K \models F_{\restriction \mathcal{L}_0}$

The following result is from [44]

Proposition 8 ([44])**.** *The following properties hold for harnessed knowledge:*

1. *If* $K \models F$ *then* $K \models_{\mathcal{L}_0} F$ *for any* $\mathcal{L}_0$
2. $K \models_{\mathcal{L}_0} F$ *if and only if* $[K, \mathcal{L}_0] \models F_{\restriction \mathcal{L}_0}$

Theorem 6. *Let* F *be satisfiable. The following conditions are equivalent:*

1. $\mathcal{L}_0$ *is informative for* F.
2. $\mathcal{L}_0 \cap Q \neq \emptyset$ *for all* Q *w.b. of* $\{F\}$.

Proof.

(1) $\Longrightarrow$ (2): From (1) it follows that $F_{\restriction \mathcal{L}_0} \not\equiv \top$.
Since $F_{\restriction \mathcal{L}_0} \equiv \delta_{\mathcal{L} \setminus \mathcal{L}_0}[\{F\}]$, then the set $\mathcal{L} \setminus \mathcal{L}_0$ does not contain any w.b. of $\{F\}$
So if Q is a w.b. then $Q \not\subseteq \mathcal{L} \setminus \mathcal{L}_0$, hence $Q \cap \mathcal{L}_0 \neq \emptyset$.
(2) $\Longrightarrow$ (1): If $\mathcal{L} \setminus \mathcal{L}_0$ intersects all the w.b., then $\mathcal{L}_0$ does not contain any w.b., hence it does not decide F. Therefore it is informative.

□

Corollary 2. *If* $K \models_{\mathcal{L}_0} F$ *then* $\mathcal{L}_0$ *has non-empty intersection with all w.b. of* K

Proof. If $K \models_{\mathcal{L}_0} F$, then $\mathcal{L}_0$ is informative and $K \models F_{\restriction \mathcal{L}_0}$. Then

$$\delta_{\mathcal{L} \setminus \mathcal{L}_0}[K] \equiv [K, \mathcal{L}_0] \models F_{\restriction \mathcal{L}_0} \not\equiv \{\top\}$$

Which implies that $\mathcal{L} \setminus \mathcal{L}_0$ does not decide K. Then, reasoning as in the above theorem, $\mathcal{L} \setminus \mathcal{L}_0$ has non-empty intersection with all w.b. of K. □

7.4. Case Study 4: Checking Knowledge Bases Partitions Are Conservative Retractions

A strategy to reduce the complexity (both in size and language) of a KB is to partition it a posteriori in order to address specific use cases. The problem will be that we must ensure that it is a conservative retraction, to preserve the knowledge represented in the KB concerning the use case. One option would be to choose the sub-languages for use cases/contexts that are w.b. This will be illustrated with an example.

The KB shown in Figure 1, $\mathcal{A}$, has been taken from [11]. The w.b. of K have been computed with the help of the implementation above referred. They are listed in Figure 2.

In [11], the authors obtain three subKBs $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$. Their languages can be used for computing conservative retractions. In [2] it is proved that

$$\mathcal{A}_i \equiv [\mathcal{A}, \mathcal{L}(\mathcal{A}_1)] \ (i \leq i \leq 3)$$

$ok\_pump \wedge on\_pump \Rightarrow water$	$man\_fill \Rightarrow \neg on\_pump$
$water \wedge ok\_boiler \wedge on\_boiler \Rightarrow steam$	$\neg ok\_boiler \Rightarrow \neg steam$
$steam \wedge coffee \Rightarrow hot\_drink$	$steam \wedge teabag \Rightarrow hot\_drink$
$man\_fill \Rightarrow water$	$\neg man\_fill \Rightarrow on\_pump$
$\neg water \Rightarrow \neg steam$	$\neg on\_boiler \Rightarrow \neg steam$
$coffee \vee teabag$	

Figure 1. Knowledge Base for the espresso machine from [11].

$$
\begin{array}{l}
Q_1 = \{teabag, man\_fill, on\_pump, steam, on\_boiler, ok\_pump\} \\
Q_2 = \{teabag, man\_fill, on\_pump, ok\_boiler, steam, ok\_pump\} \\
Q_3 = \{teabag, man\_fill, water, on\_pump, steam, ok\_pump\} \\
Q_4 = \{teabag, man\_fill, water, on\_pump, steam, on\_boiler\} \\
Q_5 = \{teabag, man\_fill, water, on\_pump, ok\_boiler, steam\} \\
Q_6 = \{coffee, man\_fill, on\_pump, steam, on\_boiler, ok\_pump\} \\
Q_7 = \{coffee, man\_fill, on\_pump, ok\_boiler, steam, ok\_pump\} \\
Q_8 = \{coffee, man\_fill, water, on\_pump, steam, ok\_pump\} \\
Q_9 = \{coffee, man\_fill, water, on\_pump, steam, on\_boiler\} \\
Q_{10} = \{coffee, man\_fill, water, on\_pump, ok\_boiler, steam\}
\end{array}
$$

Figure 2. All the w.b. of the KB for expresso machine.

Therefore, with the ideas introduced in this paper we can certify the completeness (and soundness) of the partitions made by the authors. In general terms, it is not necessary for computing all w.b. unless it is required for some kind of global analysis of the KB. A weak basis can be selected, for each use case (and the variables involved in this) that can be considered as context for the machine (for making teabag, coffee and water, respectively), see Figure 3.

$$
[K, Q_5] = [K, Q_{10}] = \begin{cases}
((water \wedge ok\_boiler) \rightarrow steam) \vee \neg\, steam \\
man\_Fill \rightarrow\ water \\
man\_Fill \rightarrow \neg on\_pump \\
\neg man\_fill \rightarrow on\_pump \\
\neg ok\_boiler \rightarrow \neg\, steam \\
\neg water \rightarrow \neg steam
\end{cases}
$$

$$
[K, Q_4] = \begin{cases}
((water \wedge on\_Boiler) \rightarrow steam) \vee \neg\, steam \\
man\_Fill \rightarrow\ water \\
man\_Fill \rightarrow \neg on\_Pump \\
\neg man\_Fill \rightarrow on\_Pump \\
\neg on\_Boiler \rightarrow \neg\, steam \\
\neg water \rightarrow \neg steam
\end{cases}
$$

Figure 3. Three conservative retractions for w.b. from Figure 1.

7.5. Case Study 5: Checking Preserving Consistency by Extension with the Assistance of Weak Bases

Although the idea of w.b. is not originally intended to address the SAT problem (included the non-clausal case), it can be useful for some related issues. One of these is to analyze consistency preservation by adding new formulas when a w.b. is available.

The idea is as follows: Suppose that K is the consistent partial KB that we have in a stage to build the final KB, and that we have a w.b. Q for K. The aim is to add new knowledge represented by a formula F, obtaining $K_1 = K \cup \{F\}$.

If K_1 was inconsistent, then $K \models \neg F$, and then, by Proposition 6,

$$\{\top\} \equiv \delta_Q[K] \models \delta_Q[\neg F]$$

Therefore, if $\delta_Q[\neg F] \not\equiv \top$, then the new KB K_1 is consistent. This test saves the computational cost of inconsistency testing on K_1 through a computation on F. In several cases F is not so complex (e.g., in RBES), thus checking that fact is straightforward.

Example 11. *It is considered a KB formed by clauses,* `exampleSat0.txt` $= K \cup \{F\}$*, from the mentioned Github repository in Remark 4, where F is the last clause,* $F = p_6 \vee \neg p_{45} \vee p_{32}$ *and K the rest of formulas. K has 79 clauses and it is known as consistent. Computing w.b. of K is:*

$Q = \{$"$p90$", "$p82$", "$p77$", "$p28$", "$p67$", "$p70$", "$p21$", "$p20$", "$p38$", "$p80$", "$p41$", "$p66$", "$p48$", "$p59$", "$p50$", "$p63$", "$p13$", "$p78$", "$p49$", "$p88$", "$p72$", "$p73$", "$p98$", "$p45$", "$p76$", "$p19$", "$p83$", "$p65$", "$p85$", "$p75$", "$p94$", "$p95$", "$p26$", "$p39$", "$p92$", "$p54$", "$p24$", "$p8$", "$p32$", "$p60$", "$p3$", "$p47$", "$p97$", "$p31$", "$p36$", "$p100$", "$p18$", "$p81$", "$p87$"$\}$

Since $(\neg F)_{\restriction Q} \equiv \neg p_6$ *is not tautology, the KB represented by* `exampleSat0.txt` *is consistent.*

It would be necessary to study whether the set that is w.b. preserves such property on the expanded KB $K_1 = K \cup \{F\}$. Let us see that when the w.b. is not preserved, then F brings quality information to the system, in the sense that it discards some K models, expressed with variables not belonging to the old w.b.

Theorem 7. *Let Q be a w.b. of K and* $F \in Form(L)$ *be a consistent formula. If* $\delta_Q[K + \{F\}] \not\equiv \{\top\}$ *then there exists a (non tautological)* $G \in Form(\mathcal{L} \setminus Q)$ *such that:*

1. $K \cup \{F\} \models G$
2. $K + \neg G$ *is consistent.*

Proof. Suppose that $\delta_Q[K + \{F\}] \not\equiv \{\top\}$. Then there exists a non tautology $G \in Form(\mathcal{L} \setminus Q)$ in that set such that $G \not\equiv \{\top\}$. Thus

$$\delta_Q[K + \{F\}] \models G$$

The fact (1) is trivial for such an formula G.

of (2): Consider $v \in val(\mathcal{L} \setminus Q)$ such that $v \models \neg G$. Since Q is a w.b. of K, then there exists $\hat{v} \in val(\mathcal{L})$ which is an extension of v satisfying $\hat{v} \models K$. Therefore $\hat{v} \models K + \neg G$, then it is consistent. □

In summary, the above result allows for the analysis of all cases when new information is added to a KB:

- $\delta_Q[K + \{F\}] \equiv \{\top\}$: in this case Q is kept as w.b.
- $\delta_Q[K + \{F\}] \not\equiv \{\top\}$: in this case, by Theorem 7 relevant information has been added,

$$K \lhd K \cup \{F\}$$

because $Mod(K \cup \{F\}) \subset Mod(K)$ since there exists $v \models K \cup \neg G$.

8. Materials and Methods

The software can be found in the following GitHub repository https://github.com/DanielRodCha/SAT-canonical.

9. Conclusions and Future Work

An analysis has been done on the variable sets (weak bases) that allows to facilitate the reasoning with KB, preserving completeness. The idea of weak bases represents a computational-logic view of algebraic projection, inspired by the geometrical view of the operation *variable forgetting* shown in a previous work of the authors [2]. In particular:

- The present work is inspired by the aforementioned vision to transfer the idea of the Krull dimension from Algebraic Geometry.
- The study of weak bases also allowed introducing a concept of dimension associated with variable forgetting.
- The dimension allows finding bounds for the complexity of saturation-based algorithms (with respect to the number of variable forgetting operations).

- The usefulness of weak bases has been shown in several use cases: contextual reasoning, KBs with distinct weak bases, some conditions equivalent to entailment, and consistency preserving under extensions of KB, among others.
- The distribution of w.b. has been related to the informative languages, in the case of Knowledge Harnessing (Case Study 3).
- Its usefulness has been proved in several general use cases (not necessarily composed by clauses, since the operators work on any formula).

The work is mainly focused on foundations. Complexity issues on variable forgetting in propositional logic have been studied in considerable depth due to its relationship with the SAT problem. (see e.g., [28,45]). Experimental results can be found in [2]. The size of the forgetting result is exponential in worst case so specific strategies have to be designed [46]. However, most works are focused on KB with bounded syntactic complexity (e.g., Horn clauses). In contrast to such works, our approach does not presuppose any syntactic limitation; it applies to any propositional formulas. The conservative retraction is a robust knowledge reduction preserving all the knowledge that is expressible in the reduced language. Our approach focuses on sublanguages that are w.b. Thus, the knowledge preservation drives the selection of the sublanguage, which differentiates our work from other solutions. An example is the already cited [11]. Its authors start from a preset family of sublanguages, being the goal formula also expressed in one of these. Therefore, they only need to detect whether other sublanguages provide information about the goal formula, information that is transferred using bridge variables. Our approach starts without these constraints, and conservative extensions keep all knowledge in the corresponding sublanguage. Therefore, our approach applies to the partitioning of KB as in the mentioned work, without prefixing sub-languages. This way an inadequate selection of them is avoided. Moreover, although a sub-language was prefixed, the use of w.b. would allow obtaining conservative retractions for the associated contexts.

In [37], the relation between consistent subKBs of an inconsistent KB is studied, by applying variable forgetting to enable a version of consequence based on maximal consistent subsets. In our approach, w.b. are interesting in the case of consistent K, since the approach is focused on entailment in KBs. It is assumed that when using a w.b. the inconsistency of the KB is detected, the knowledge base is no longer used and the repair process is moved on, for which several tools exist (see e.g., [47,48]). In sum, the notion of w.b. is mainly introduced for reasoning with consistent KB's. Future work will address the issue of inconsistent KBs.

It has already been described that an objection to our proposal could be that the computing of all w.b. can be expensive. Rather the idea is to build weak bases for particular variable sets (for example, contexts where the system/agent will work) or to compute w.b. when the KB remains stable. Moreover, the most interesting case is building the knowledge base at the same time as its w.b. (Case Study of Section 7.5). One of the future work goals is the development of a methodology achieving this, within the Knowledge Engineering paradigm.

Another limitation may be that although the language is reduced, the formulas obtained could be syntactically more complex (or larger). This does not happen in practice when f.o. that are more sophisticated than the canonical ones, are used (e.g., those introduced in [2]). These have not been used in this work in order to keep, as far as possible, the paper self-contained. In practice, language reduction makes entailment problems relatively simpler. Even the saturation based solver of that paper can be used to solve complex (not necessarily clausal) instances of SAT (included clausal instances from SAT competitions) [2].

A future research line is to study the role that w.b. can play in distributed RBES. The idea would be based on to use of w.b. for distributing both the knowledge and the reasoning. In this way, the control strategies, suitable for distributed environments for Knowledge-based systems, could be enriched [49].

Author Contributions: Conceptualization, J.B.-D. and D.R.-C.; methodology, J.B.-D., D.R.-C., G.A.A.-C. and J.G.-P.; software, D.R.-C., G.A.A.-C. and J.G.-P.; formal analysis, J.B.-D.; investigation, J.B.-D. and D.R.-C.; writing—original draft preparation, J.B.-D. and J.G.-P.; writing—review and editing, J.B.-D. and J.G.-P. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by State Investigation Agency *(Agencia Estatal de Investigación)*, project PID2019-109152GB-I00/AEI/10.13039/501100011033.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Calegari, R.; Ciatto, G.; Denti, E.; Omicini, A. Logic-Based Technologies for Intelligent Systems: State of the Art and Perspectives. *Information* **2020**, *11*, 167. [CrossRef]
2. Alonso-Jiménez, J.A.; Aranda-Corral, G.A.; Borrego-Díaz, J.; Fernández-Lebrón, M.M.; Hidalgo-Doblado, M. A logic-algebraic tool for reasoning with Knowledge-Based Systems. *J. Log. Algebr. Meth. Program.* **2018**, *101*, 88–109. [CrossRef]
3. Agudelo-Agudelo, J.C.; Agudelo-González, C.A.; García-Quintero, O.E. On polynomial semantics for propositional logics. *J. Appl. Non-Class. Logics* **2016**, *26*, 103–125. [CrossRef]
4. Agudelo-Agudelo, J.C.; Echeverri-Valencia, S. Polynomial semantics for modal logics. *J. Appl. Non-Class. Logics* **2019**, *29*, 430–449. [CrossRef]
5. Hernando, A.; Roanes-Lozano, E. An algebraic model for implementing expert systems based on the knowledge of different experts. *Math. Comput. Simul.* **2015**, *107*, 92–107. [CrossRef]
6. Roanes-Lozano, E.; Hernando, A.; Laita, L.M.; Roanes-Macías, E. A Groebner bases-based approach to backward reasoning in rule based expert systems. *Ann. Math. Artif. Intell.* **2009**, *56*, 297–311. [CrossRef]
7. Roanes-Lozano, E.; García, J.L.G.; Venegas, G.A. A portable knowledge-based system for car breakdown evaluation. *Appl. Math. Comput.* **2015**, *267*, 758–770. [CrossRef]
8. Roanes-Lozano, E.; García, J.L.G.; Venegas, G.A. A prototype of a RBES for personalized menus generation. *Appl. Math. Comput.* **2017**, *315*, 615–624. [CrossRef]
9. Roanes-Lozano, E.; Casella, E.A.; Sánchez, F.; Hernando, A. Diagnosis in Tennis Serving Technique. *Algorithms* **2020**, *13*, 106. [CrossRef]
10. Hernando, A.; Maestre, R.; Roanes-Lozano, E. A New Algebraic Approach to Decision Making in a Railway Interlocking System Based on Preprocess. *Math. Probl. Eng.* **2018**, *2018*, 1–14. [CrossRef]
11. Amir, E.; McIlraith, S. Partition-based Logical Reasoning for First-order and Propositional Theories. *Artif. Intell.* **2005**, *162*, 49–88. [CrossRef]
12. Roanes-Lozano, E.; Angélica Martínez-Zarzuelo, M.J.F.D. An Application of Knowledge Engineering to Mathematics Curricula Organization and Formal Verification. *Math. Probl. Eng.* **2020**, 1–12. [CrossRef]
13. Fernandez-Amoros, D.; Bra, S.; Aranda-Escolástico, E.; Heradio, R. Using Extended Logical Primitives for Efficient BDD Building. *Mathematics* **2020**, *8*, 1253. [CrossRef]
14. Piury, J.; Laita, L.; Roanes-Lozano, E.; Hernando, A.; Piury-Alonso, F.J.; Gomez-Arguelles, J.; Laita, L. A Gröbner bases-based rule based expert system for fibromyalgia diagnosis. *Rev. Real Acad. Cienc. Exactas Fis. Nat. Ser. A Mat.* **2012**, *106*, 443–456. [CrossRef]
15. Ishii, H. A Purely Functional Computer Algebra System Embedded in Haskell. In *Computer Algebra in Scientific Computing*; Gerdt, V.P., Koepf, W., Seiler, W.M., Vorozhtsov, E.V., Eds.; Springer: Berlin, Germany, 2018; pp. 288–303.
16. Blair, P.; Guha, R.V.; Pratt, W. *Microtheories: An Ontological Engineer's Guide*; Technical Report; CyC Corp.: Austin, TX, USA, 1992.
17. Wang, Y. On Forgetting in Tractable Propositional Fragments. *arXiv* **2015**, arXiv:1502.02799.
18. Su, K.; Sattar, A.; Lv, G.; Zhang, Y. Variable Forgetting in Reasoning about Knowledge. *J. Artif. Intell. Res.* **2009**, *35*, 677–716. [CrossRef]
19. Wang, Z.; Wang, K.; Topor, R.W.; Pan, J.Z. Forgetting for knowledge bases in DL-Lite. *Ann. Math. Artif. Intell.* **2010**, *58*, 117–151. [CrossRef]
20. Rajaratnam, D.; Levesque, H.J.; Pagnucco, M.; Thielscher, M. Forgetting in Action. In *Principles of Knowledge Representation and Reasoning, Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, 22–24 July 2014*; Baral, C., Giacomo, G.D., Eiter, T., Eds.; AAAI Press: Palo Alto, CA, USA, 2014; pp. 498–507.
21. Lutz, C.; Wolter, F. Conservative Extensions in the Lightweight Description Logic EL. In *Automated Deduction, Proceedings of the CADE-21, 21st International Conference on Automated Deduction, Bremen, Germany, 17–20 July 2007*; Springer: Cham, Switzerland, 2007; pp. 84–99.
22. Hidalgo-Doblado, M.J.; Alonso-Jiménez, J.A.; Borrego-Díaz, J.; Martín-Mateos, F.; Ruiz-Reina, J. Formally Verified Tableau-Based Reasoners for a Description Logic. *J. Autom. Reason.* **2014**, *52*, 331–360. [CrossRef]

23. Martín-Mateos, F.; Alonso, J.; Hidalgo, M.; Ruiz-Reina, J. Formal Verification of a Generic Framework to Synthetize SAT-Provers. *J. Autom. Reason.* **2004**, *32*, 287–313. [CrossRef]
24. Aranda-Corral, G.A.; Borrego-Díaz, J.; Galán-Páez, J. Complex concept lattices for simulating human prediction in sport. *J. Syst. Sci. Complex.* **2013**, *26*, 117–136. [CrossRef]
25. Moinard, Y. Forgetting Literals with Varying Propositional Symbols. *J. Log. Comput.* **2007**, *17*, 955–982. [CrossRef]
26. Lin, F.; Reiter, R. Forget It! In Proceedings of the AAAI Fall Symposium on Relevance, New Orleans, LA, USA, 4–6 November 1994; pp. 154–159.
27. Lin, F. On strongest necessary and weakest sufficient conditions. *Artif. Intell.* **2001**, *128*, 143–159. [CrossRef]
28. Lang, J.; Liberatore, P.; Marquis, P. Propositional Independence—Formula-Variable Independence and Forgetting. *J. Artif. Intell. Res.* **2003**, *18*, 391–443. [CrossRef]
29. Wang, Y.; Wang, K.; Wang, Z.; Zhuang, Z. Knowledge Forgetting in Circumscription: A Preliminary Report. In Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 1649–1655.
30. Lang, J.; Liberatore, P.; Marquis, P. Conditional independence in propositional logic. *Artif. Intell.* **2002**, *141*, 79–121. [CrossRef]
31. Eiter, T.; Kern-Isberner, G. A Brief Survey on Forgetting from a Knowledge Representation and Reasoning Perspective. *Künstliche Intell.* **2019**, *33*, 9–33. [CrossRef]
32. Bledsoe, W.W.; Hines, L.M. Variable Elimination and Chaining in a Resolution-based Prover for Inequalities. *CADE Lect. Notes Comput. Sci.* **1980**, *87*, 70–87.
33. Larrosa, J.; Morancho, E.; Niso, D. On the Practical use of Variable Elimination in Constraint Optimization Problems: 'Still-life' as a Case Study. *J. Artif. Intell. Res.* **2005**, *23*, 421–440. [CrossRef]
34. Eiter, T.; Wang, K. Semantic forgetting in answer set programming. *Artif. Intell.* **2008**, *172*, 1644–1672. [CrossRef]
35. Wang, Y.; Wang, K.; Zhang, M. Forgetting for Answer Set Programs Revisited. In Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013), Beijing, China, 3–9 August 2013; pp. 1162–1168.
36. Zhang, Y.; Zhou, Y. Knowledge forgetting: Properties and applications. *Artif. Intell.* **2009**, *173*, 1525–1537. [CrossRef]
37. Lang, J.; Marquis, P. Reasoning under inconsistency: A forgetting-based approach. *Artif. Intell.* **2010**, *174*, 799–823. [CrossRef]
38. Xu, D.; Zhang, X.; Lin, Z. A forgetting-based approach to merging knowledge bases. In Proceedings of the 2010 IEEE International Conference on Progress in Informatics and Computing, Shanghai, China, 10–12 December 2010; Volume 1, pp. 321–325.
39. Thimm, M.; Wallner, J.P. On the complexity of inconsistency measurement. *Artif. Intell.* **2019**, *275*, 411–456. [CrossRef]
40. Jabbour, S.; Ma, Y.; Raddaoui, B.; Sais, L. Quantifying conflicts in propositional logic through prime implicates. *Int. J. Approx. Reason.* **2017**, *89*, 27–40. [CrossRef]
41. Xiao, G.; Ma, Y. Inconsistency Measurement based on Variables in Minimal Unsatisfiable Subsets. *Front. Artif. Intell. Appl.* **2012**, *242*, 864–869.
42. Peitl, T.; Szeider, S. Finding the Hardest Formulas for Resolution. In *Principles and Practice of Constraint Programming*; Simonis, H., Ed.; Springer: Cham, Switzerland, 2020; pp. 514–530.
43. Aranda-Corral, G.A.; Borrego-Díaz, J.; Galán-Páez, J.; Caballero, A.T., On Experimental Efficiency for Retraction Operator to Stem Basis. In *Trends in Mathematics and Computational Intelligence*; Springer: Cham, Switzerland, 2019; Chapter 8, pp. 73–79.
44. Aranda-Corral, G.A.; Borrego-Díaz, J.; Galán-Páez, J. A model of three-way decisions for Knowledge Harnessing. *Int. J. Approx. Reason.* **2020**, *120*, 184–202. [CrossRef]
45. Marques-Silva, J.; Janota, M.; Mencía, C. Minimal sets on propositional formulae. Problems and reductions. *Artif. Intell.* **2017**, *252*, 22–50. [CrossRef]
46. Zhou, Y. Polynomially Bounded Forgetting. In *PRICAI 2014: Trends in Artificial Intelligence*; Pham, D.N., Park, S.B., Eds.; Springer: Cham, Switzerland, 2014; pp. 422–434.
47. Besnard, P. Forgetting-Based Inconsistency Measure. In *Scalable Uncertainty Management, Proceedings of the 10th International Conference, SUM 2016, Nice, France, 21–23 September 2016*; Lecture Notes in Computer Science; Schockaert, S., Senellart, P., Eds.; Springer: Berlin, Germany, 2016; Volume 9858, pp. 331–337.
48. Besnard, P.; Grant, J. Relative inconsistency measures. *Artif. Intell.* **2020**, *280*, 103231. [CrossRef]
49. Boaye-Belle, A.; Lethbridge, T.C.; Garzón, M.; Adesina, O.O. Design and implementation of distributed expert systems: On a control strategy to manage the execution flow of rule activation. *Expert Syst. Appl.* **2018**, *96*, 129–148. [CrossRef]