Computation of Conditional Syntax Splittings for Inductive Inference from Belief Bases

Lars-Phillip Spiegel*, Franziska Herkner and Christoph Beierle

FernUniversität in Hagen, Knowledge-Based Systems, Faculty of Mathematics and Computer Science, Hagen, Germany

Abstract

Nonmonotonic reasoning with conditional belief bases aims at answering the question "Given a belief base Δ and two formulas A and B, does formula A entail formula B in the context of Δ?". Answering these inference tasks is usually quite complex as their semantics involve structures that often grow exponentially in signature size. As a means of localizing such inferences tasks, safe conditional syntax splitting has been introduced in order to split conditional belief bases into smaller subbases, allowing to take only the relevant parts of a given belief base into account. Recently, the kinds of splittings that can be exploited for inductive inference have been refined in two ways: Generalized safe splittings broaden the number of splittings exploitable while genuine splittings identify those splittings that can be meaningfully used for inductive inference. While the theoretical aspect of splittings has been investigated in some detail, their implementation has received far less attention. We present algorithms for computing the different kinds of conditional syntax splittings of a given belief base. These algorithms are capable of computing all conditional syntax splittings and finding all safe, generalized safe, and genuine splittings of a belief base. A modular approach ensures that any combination of safe, generalized safe, and genuine, e.g., the genuine and generalized safe splittings, can be easily computed as well. We evaluate an implementation of these algorithms with respect to the number of splittings computed for each kind and the runtime of each algorithm, analyzing in particular the differences between safe and generalized safe splittings, as well as the benefits of genuine splittings.

Keywords

Splitting, conditional, belief base, conditional syntax splitting, inductive inference, algorithms

1. Introduction

A central aspect of knowledge representation and reasoning (KRR) is that of nonmonotonic reasoning: Given a belief base Δ and two formulas A and B, does formula A entail formula B in the context of Δ? Reasoning mechanisms aiming at answering this question can be formalized as inductive inference operators [1]. Examples of inductive inference operators include p-entailment [2, 3], system Z [3], lexicographic entailment [4], skeptical c-inference [5], and system W [6, 7]. Computation of these inference operators often suffers from the fact that their semantics rely on structures that grow exponentially in the underlying signature size. Splitting techniques aim at tackling this problem, by focusing only on the relevant parts of a belief base, reducing the number of conditionals and signature elements that need to be taken into account for inference. Originally introduced by Parikh for belief revision [8], syntax splitting and its generalization conditional syntax splitting [9] have established themselves in recent years as a useful means to localize reasoning tasks via a set of postulates for inductive inference operators [1, 9, 10]. Multiple kinds of conditional syntax splittings, aiming at identifying splittings useful for inductive inference, have been proposed [9, 10]. However, while their theoretical aspects have been researched, approaches for computing the different kinds of conditional syntax splittings for a given belief base are missing.

In this paper, we present algorithms for computing all conditional syntax splittings of a given belief

Joint proceedings of the Workshop on Theory and Methods for Abstraction (THEMA 2026) and the Workshop on Modularity and Splitting Techniques for Knowledge Representation and Reasoning (MoST 2026) at the 9th Federated Logic Conference (FLoC 2026), July 13–29, Lisbon, Portugal

*Corresponding author.

lars-phillip.spiegel@fernuni-hagen.de (L. Spiegel); herkner.research@web.de (F. Herkner); christoph.beierle@fernuni-hagen.de (C. Beierle)

0009-0001-1962-752X (L. Spiegel); 0000-0002-0736-8516 (C. Beierle)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-15

CEUR-WS.org/Vol-4331/paper6.pdf

base Δ, as well as all safe, generalized safe and genuine splittings of Δ. A modular approach ensures that the algorithms can be easily combined to compute any combination of these kinds of splittings which we illustrate with an algorithm computing the most beneficial splittings of Δ, i.e., the splittings that are both genuine and generalized safe [10]. We implement these algorithms and evaluate them on different belief bases. We observe and evaluate both the number of different kinds of splittings as well as the runtime of the presented algorithms.

The rest of this paper is organized as follows. We briefly recall the necessary technical preliminaries in Sect. 2. In Sect. 3 we give an overview of the state of the art of conditional syntax splitting of belief bases and present our developed algorithms in Sect. 4. We evaluate an implementation of our proposed algorithms in Sect. 5 and point out future research in Sect. 6 before concluding in Sect. 7.

2. Preliminaries

We consider a finitely generated propositional language $\mathcal{L}$ over a signature Σ with atoms $a, b, c, \ldots$ and with formulas $A, B, C, \ldots$ For conciseness of notation, we may write AB instead of $A \wedge B$, and $\overline{A}$ instead of $\neg A$. With $atoms(A)$ we denote the signature elements occurring in a formula A, i.e., $atoms(a \wedge \overline{b}) = \{a, b\}$. As models of formulas we will use the set Ω of *possible worlds*, or propositional interpretations, over $\mathcal{L}$. We will use ω both for the model and the corresponding conjunction of all positive or negated atoms, allowing us to use ω both as an interpretation and a formula. Since $\omega \models A$, denoting both the satisfaction and logical entailment relation, means the same for both readings of ω, no confusion will arise. For subsets Σ_i of Σ, let $\mathcal{L}(\Sigma_i)$ denote the propositional language defined by Σ_i, with associated set of possible worlds $\Omega(\Sigma_i)$. Note that while each formula of $\mathcal{L}(\Theta)$ can also be considered as a formula of $\mathcal{L}$ if $\Theta \subseteq \Sigma$, the interpretations $\omega^\Theta \in \Omega(\Theta)$ are not elements of $\Omega(\Sigma)$ if $\Theta \neq \Sigma$. Furthermore, each interpretation $\omega \in \Omega$ can be written uniquely in the form $\omega = \omega^\Theta \omega^{\overline{\Theta}}$ with concatenated $\omega^\Theta \in \Omega(\Theta)$ and $\omega^{\overline{\Theta}} \in \Omega(\overline{\Theta})$, where $\overline{\Theta} = \Sigma \backslash \Theta$. Note that the syntactical reading of interpretations aligns with this representation: ω is a conjunction of ω^Θ and $\omega^{\overline{\Theta}}$ (with omitted $\wedge$ symbol). With ω^i we denote the reduct of ω to Σ_i [11]. Conditionals $(B|A)$ are meant to express plausible, yet defeasible rules "If A then usually B". A conditional $(B|A)$ is *verified* by ω if $\omega \models AB$ and *falsified* by ω if $\omega \models A\overline{B}$. For a conditional $(B|A)$ we may write $atoms(B|A)$ as shorthand for $atoms(A) \cup atoms(B)$. A conditional $(B|A)$ is called *self-fulfilling*, or *trivial*, if $A \models B$, i.e., there is no world that falsifies it. A belief base Δ is a finite set of conditionals. As models for conditionals we use ordinal conditional functions (OCFs) $\kappa \to \mathbb{N}_0 \cup \{\infty\}$ that map worlds to *ranks* such that at least one world has rank 0, i.e., $\kappa^{-1}(0) \neq \emptyset$, introduced first in a more general form by Spohn [12]. The intuition is that worlds with lower ranks are more plausible than worlds ranked higher; worlds ranked ∞ are considered entirely infeasible. OCFs are lifted to formulas by $\kappa(A) = \min\{\omega \mid \omega \models A\}$. An OCF κ accepts a conditional $(B|A)$ if $\kappa(AB) < \kappa(A\overline{B})$. An OCF κ is a model of a belief base Δ if κ accepts all conditionals of Δ. A belief base Δ is strongly consistent (in the sense of [13, 3]) if there is a model κ of Δ with $\kappa(\omega) < \infty$ for all $\omega \in \Omega$, i.e., all worlds are at least somewhat plausible. In this paper, we focus on strongly consistent belief bases. To formalize inductive inference from belief bases, the notion of inductive inference operators was introduced [1]. An *inductive inference operator* is a mapping $\mathbf{C}$ that assigns to each belief base $\Delta \subseteq (\mathcal{L} \mid \mathcal{L})$ an inference relation $\mid\!\sim_\Delta \subseteq \mathcal{L} \times \mathcal{L}$ on $\mathcal{L}$, i.e., $\mathbf{C} : \Delta \mapsto \mid\!\sim_\Delta$, such that the two properties (DI): If $(B|A) \in \Delta$ then $A \mid\!\sim_\Delta B$, and (TV): $A \mid\!\sim_\emptyset B$ implies $A \models B$ are satisfied. Examples of inductive inference operators include p-entailment [2, 3], system Z [3], lexicographic inference [4], c-inference [5], and system W [6, 7].

3. Conditional Syntax Splitting

Syntax splittings describe that a belief base Δ contains completely independent information about different parts of the signature. According to [1], Δ *splits* into subbases Δ_1, Δ_2 if $\{\Sigma_1, \Sigma_2\}$ is a partition of Σ such that $\Delta = \Delta_1 \cup \Delta_2$, $\Delta_i \subsetneq (\mathcal{L}_i|\mathcal{L}_i), \mathcal{L}_i = \mathcal{L}(\Sigma_i)$ for $i \in \{1, 2\}$, $\Sigma_1 \cap \Sigma_2 = \emptyset$, and $\Sigma_1 \cup \Sigma_2 = \Sigma$, denoted as $\Delta = \Delta_1 \bigcup_{\Sigma_1, \Sigma_2} \Delta_2$.

Syntax splittings were generalized to *conditional* syntax splittings, which allow subbases to share some atoms in a given subsignature Σ_3 [9].

Definition 1 (conditional syntax splitting [9]). *A belief base* Δ splits into subbases Δ_1,Δ_2 conditional on Σ_3, *if there are* $\Sigma_1, \Sigma_2 \subseteq \Sigma$ *such that* $\Delta_i = \Delta \cap (\mathcal{L}(\Sigma_i \cup \Sigma_3) \mid \mathcal{L}(\Sigma_i \cup \Sigma_3))$ *for* $i \in \{1, 2\}$, *and* $\{\Sigma_1, \Sigma_2, \Sigma_3\}$ *is a partition of* Σ. *This is denoted as* $\Delta = \Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3$.

Unlike syntax splitting, conditional syntax splitting does not require the subbases Δ_1 and Δ_2 to be disjoint. For the rest of this paper, we will often use Δ_3 to denote the intersection of the subbases, i.e., $\Delta_3 = \Delta_1 \cap \Delta_2$.

In the literature different kinds of conditional syntax splittings were introduced.

Definition 2 (safe, generalized safe, genuine conditional syntax splitting [9, 10]). *A conditional syntax splitting* $\Delta = \Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3$ *is called*

- safe, *denoted* $\Delta = \Delta^1 \bigcup^{\mathsf{s}}_{\Sigma_1,\Sigma_2} \Delta^2 \mid \Sigma_3$, *if the following* safety condition *holds [9]:*

$$\text{for every } \omega^i\omega^3 \in \Omega(\Sigma_i \cup \Sigma_3) \text{ there is } \omega^{i'} \in \Omega(\Sigma_{i'}) \text{ s.t. } \omega^i\omega^3\omega^{i'} \not\models \bigvee_{(F|E)\in\Delta_{i'}} E \wedge \neg F.$$

- generalized safe, *denoted* $\Delta = \Delta_1 \bigcup^{\mathsf{gs}}_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3$, *if the following* generalized safety *condition holds [10]:*

$$\text{for every } \omega^i\omega^3 \in \Omega(\Sigma_i \cup \Sigma_3) \text{ there is } \omega^{i'} \in \Omega(\Sigma_{i'}) \text{ s.t. } \omega^i\omega^3\omega^{i'} \not\models \bigvee_{(F|E)\in\Delta_{i'}\setminus\Delta_3} E \wedge \neg F.$$

- genuine *if* $\Delta_1 \not\subseteq \Delta_2$ *and* $\Delta_2 \not\subseteq \Delta_1$ *[10].*

Safe splittings were introduced because a purely syntactical conditional syntax splitting is not sufficient to guarantee independence between the subbases Δ_1 and Δ_2 [9]. This is because conditionals in one subbase may put constraints on signature elements in the conditional pivot Σ_3 that are not reflected in the other subbase (cf. [9, Example 6]), making some interpretations over Σ_3 valid in one subbase, but not in the other. Note that the key difference between safe splittings and generalized safe splittings is that the conditionals in the intersection $\Delta_1 \cap \Delta_2$ are not considered for the generalized safety condition.

Example 1 (Δ^e). *Consider the belief base* $\Delta^e = \{(f|b), (\overline{f}|p), (b|p), (b|e)\}$ *over the signature* $\Sigma = \{b, p, f, e\}$ *denoting that an entity is a* b*ird, is a* p*enguin,* f*lies, or is an* e*agle.* Δ^e *has the safe conditional syntax splitting*

$$\Delta^e = \{(f|b), (\overline{f}|p), (b|p)\} \bigcup^{\mathsf{s}}_{\{p,f\},\{e\}} \{(b|e)\} \mid \{b\}.$$

The safety condition can be verified by checking that every $\omega^1\omega^3 \in \Omega(\{p, f\} \cup \{b\})$ *can be extended with* $\omega^2 = \{\overline{e}\}$ *such that* $(b|e) \in \Delta_2 = \{(b|e)\}$ *is not falsified by* $\omega^1\omega^2\omega^3$, *and that every* $\omega^{*2}\omega^{*3} \in \Omega(\{e\} \cup \{b\})$ *can be extended with some* $\omega^{*1} \in \Omega(\{p, f\})$ *such that no conditional in* $\Delta_1 = \{(f|b), (\overline{f}|p), (b|p)\}$ *is falsified by* $\omega^{*1}\omega^{*2}\omega^{*3}$.

It has recently been shown that the safety condition is quite a strong restriction on the intersection of the subbases $\Delta_1 \cap \Delta_2$, allowing only trivial, self-fulfilling conditionals to occur in the intersection [14]. Generalized safe splittings allow for more expressive conditionals in the intersection [10].

Example 2 (Δ^r). *Consider* $\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr), (o|s\overline{r}), (\overline{o}|r), (u|or)\}$ *describing the following: If it is (r)ainy, then usually it is not (s)unny, and if it is sunny, then usually it is not rainy. If it is rainy and sunny at the same time, then we can usually observe a rain(b)ow. We usually spend some time (o)utside if it is sunny and not rainy. If it is rainy, then we usually do not spend time outside. If we do spend time outside*

and it is rainy, then we usually have an (u)mbrella. This belief base has the generalized safe conditional syntax splitting

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr)\} \bigcup_{\{b\},\{o,u\}}^{\mathsf{gs}} \{(\overline{s}|r), (\overline{r}|s), (o|s\overline{r}), (\overline{o}|r), (u|or)\} \mid \{s, r\}$$

with $\Delta_1^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr)\}$ *and* $\Delta_2^r = \{(\overline{s}|r), (\overline{r}|s), (o|s\overline{r}), (\overline{o}|r), (u|or)\}$*. The splitting is not safe because* $(\overline{s}|r) \in \Delta_1^r \cap \Delta_2^r = \{(\overline{s}|r), (\overline{r}|s)\}$ *is not a trivial conditional. Furthermore, every safe splitting of* $\Delta^r = \Delta_1^r \bigcup_{\Sigma_1,\Sigma_2}^{\mathsf{s}} \Delta_2^r \mid \Sigma_3$ *satisfies* $\Delta_1^r \subseteq \Delta_2^r$ *or* $\Delta_2^r \subseteq \Delta_1^r$*, meaning that there is no safe splitting that can be meaningfully exploited for inductive inference (cf. [10]).*

Because every belief base usually has a large amount of conditional syntax splittings (e.g. [10, Ex.13]), genuine splittings were introduced to focus on splittings beneficial for inductive inference and to filter out the large class of *simple* splittings [10].

Example 3 (Δ^e, Δ^r cont.)**.** *Both splittings in Examples 1 and 2 are genuine. The splitting*

$$\Delta^e = \Delta^e \bigcup_{\{p,e\},\emptyset}^{\mathsf{gs}} \{(f|b)\} \mid \{b, f\}.$$

is not genuine because $\Delta_2 = \{(f|b)\} \subseteq \Delta^e = \Delta_1$*.*

Generalized safe conditional syntax splitting, a generalization of safe conditional syntax splitting (cf. [1, 9]), gives rise to a set of postulates for inductive inference, demanding that inferences in the sub-language $\Sigma_i \cup \Sigma_3$ can be made taking only Δ_i into account, and that inferences from one sub-language are independent from formulas over the other sub-language.

(CRel$^{\mathsf{g}}$) [10] An inductive inference operator $\mathbf{C}$ satisfies *generalized conditional relevance* if for any $\Delta = \Delta_1 \bigcup_{\Sigma_1,\Sigma_2}^{\mathsf{gs}} \Delta_2 \mid \Sigma_3$, for $i \in \{1, 2\}$ and any $A, B \in \mathcal{L}(\Sigma_i)$, and a complete conjunction $E \in \mathcal{L}(\Sigma_3)$: $AE \mathrel{|\!\sim_{\Delta}} B$ iff $AE \mathrel{|\!\sim_{\Delta_i}} B$.

(CInd$^{\mathsf{g}}$) [10] An inductive inference operator $\mathbf{C}$ satisfies *generalized conditional independence* if for any $\Delta = \Delta_1 \bigcup_{\Sigma_1,\Sigma_2}^{\mathsf{gs}} \Delta_2 \mid \Sigma_3$, for $i, j \in \{1, 2\}$, $j \neq i$, and any $A, B \in \mathcal{L}(\Sigma_i)$, $D \in \mathcal{L}(\Sigma_j)$, and a complete conjunction $E \in \mathcal{L}(\Sigma_3)$, such that $DE \not\mathrel{|\!\sim_{\Delta}} \bot$: $AE \mathrel{|\!\sim_{\Delta}} B$ iff $AED \mathrel{|\!\sim_{\Delta}} B$.

(CSynSplit$^{\mathsf{g}}$) [10] An inductive inference operator $\mathbf{C}$ satisfies *generalized conditional syntax splitting* if it satisfies (CRel$^{\mathsf{g}}$) and (CInd$^{\mathsf{g}}$).

Example 4 (Δ^r cont.)**.** *Recall the splitting in Example 2 and let* $C : \Delta \mapsto \mathrel{|\!\sim}_{\Delta}^{rain}$ *be an inductive inference operator satisfying (CSynSplit$^{\mathsf{g}}$). Using (CRel$^{\mathsf{g}}$), we obtain that* $sr \mathrel{|\!\sim}_{\Delta}^{rain} b$ *iff* $sr \mathrel{|\!\sim}_{\Delta_1}^{rain} b$*. Thus, it is sufficient to consider only* Δ_1 *for this inference, reducing the number of conditionals to be taken into account from 6 to 3 and the number of signature elements from 5 to 3. By applying (CInd$^{\mathsf{g}}$), we additionally know that both* $sro \mathrel{|\!\sim}_{\Delta}^{rain} b$ *and* $sr\overline{o} \mathrel{|\!\sim}_{\Delta}^{rain} b$ *hold iff* $sr \mathrel{|\!\sim}_{\Delta}^{rain} b$ *holds.*

Note that the splittings where the postulates (CInd$^{\mathsf{g}}$) and (CRel$^{\mathsf{g}}$) can be meaningfully applied are exactly the genuine and generalized safe splittings [10].

4. Algorithms

In this section, we present algorithms for computing the different kinds of conditional syntax splittings of a given belief base. All algorithms are designed in a modular way, such that any combination of kinds of conditional syntax splittings can easily be computed by employing the corresponding algorithms. We illustrate this by designing an algorithm combining all different kinds of conditional syntax splittings and also an algorithm computing exactly those splittings exploitable for inductive inference.

Algorithm 1: `condSplittings`

Input: Belief base Δ, signature Σ of Δ
Output: Set of all conditional syntax splittings of Δ

1 $allSplittings \leftarrow \emptyset$
2 **forall** *partitions* $\{\Sigma_1, \Sigma_2, \Sigma_3\}$ *of* Σ **do**
3 $\Delta_1 \leftarrow \emptyset; \Delta_2 \leftarrow \emptyset$
4 **foreach** $(B|A) \in \Delta$ **do**
5 **if** $atoms(B|A) \subseteq (\Sigma_1 \cup \Sigma_3)$ **then**
6 $\Delta_1 \leftarrow \Delta_1 \cup \{(B|A)\}$
7 **if** $atoms(B|A) \subseteq (\Sigma_2 \cup \Sigma_3)$ **then**
8 $\Delta_2 \leftarrow \Delta_2 \cup \{(B|A)\}$
9 **if** $\Delta_1 \cup \Delta_2 = \Delta$ **then**
10 $allSplittings \leftarrow allSplittings \cup \{\Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3\}$
11 **return** $allSplittings$

Algorithm 2: `genuineSplittings`

Input: Set *splittings* of conditional syntax splittings
Output: Set of all genuine conditional syntax splittings in *splittings*

1 $genuineSplittings \leftarrow \emptyset$
2 **foreach** $\Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3 \in splittings$ **do**
3 **if** $\Delta_1 \not\subseteq \Delta_2 \wedge \Delta_2 \not\subseteq \Delta_1$ **then**
4 $genuineSplittings \leftarrow genuineSplittings \cup \{\Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3\}$
5 **return** $genuineSplittings$

First, Algorithm 1 computes all conditional syntax splittings of a given Δ. It takes Δ and its signature as input and iterates over all partitions $\{\Sigma_1, \Sigma_2, \Sigma_3\}$ of Σ, computing for each the subbases Δ_1 and Δ_2 by checking the signature elements of each conditional (ll. 4-8). If all conditionals were assigned to Δ_1, Δ_2 or both, the algorithm has found a conditional syntax splitting and adds it to the output (ll. 9-10).

After Algorithm 1 has computed all conditional syntax splittings, the following algorithms can be used in any order dependent on which splittings should be computed.

Algorithm 2 takes a set of conditional syntax splittings and returns all genuine splittings of that set. This is done by checking for each splitting in the input, whether it is genuine, i.e., whether $\Delta_1 \not\subseteq \Delta_2$ and $\Delta_2 \not\subseteq \Delta_1$ (ll. 2-4).

Algorithm 3 takes a set of conditional syntax splittings as input and returns the set of generalized safe splittings in the input. This is done by validating the generalized safety condition (ll. 2-7), adding a splitting to the output if it holds. Algorithm 4 is analogous to Algorithm 3 but validates the safety instead of the generalized safety condition.

Next we design two algorithms utilizing the modular construction to compute all kinds of splittings and all splittings exploitable w.r.t. inductive inference. Algorithm 5 computes all types of splittings for a given belief base. This is useful, e.g., for statistical purposes, i.e., to see how many genuine splittings a belief base has or how many generalized safe splittings exist as opposed to safe splittings.

Example 5 (Δ^e, Δ^r cont.). *For Δ^e, Algorithm 5 computes 19 conditional syntax splittings, 3 genuine splittings, 5 safe splittings, 1 genuine and safe splitting, 11 generalized safe splittings, and 2 genuine and generalized safe splittings.*

For Δ^r, Algorithm 5 computes 37 conditional syntax splittings, 5 genuine splittings, 16 safe splittings, 0

Algorithm 3: `genSafeSplittings`

Input: Set *splittings* of conditional syntax splittings
Output: Set of all generalized safe splittings in *splittings*

genSafeSplittings $\leftarrow \emptyset$
foreach $\Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3 \in$ *splittings* **do**
  genSafe_delta2 $\leftarrow \forall\omega^1\omega^3 \in \Omega(\Sigma_1 \cup \Sigma_3),\ \exists\omega^2 \in \Omega(\Sigma_2) : \forall(F|E) \in \Delta_2 \setminus \Delta_3,\ \omega^1\omega^3\omega^2 \not\models E \wedge \neg F$
  if *genSafe_delta2* **then**
    genSafe_delta1 $\leftarrow \forall\omega^2\omega^3 \in \Omega(\Sigma_2 \cup \Sigma_3),\ \exists\omega^1 \in \Omega(\Sigma_1) : \forall(F|E) \in \Delta_1 \setminus \Delta_3,\ \omega^2\omega^3\omega^1 \not\models E \wedge \neg F$
  if *genSafe_delta1* $\wedge$ *genSafe_delta2* **then**
    genSafeSplittings $\leftarrow$ *genSafeSplittings* $\cup \{\Delta_1 \bigcup^{\mathsf{gs}}_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3\}$
return *genSafeSplittings*

Algorithm 4: `safeSplittings`

Input: Set *splittings* of conditional syntax splittings
Output: Set of all safe splittings in *splittings*

safeSplittings $\leftarrow \emptyset$
foreach $\Delta_1 \bigcup_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3 \in$ *splittings* **do**
  safe_delta2 $\leftarrow \forall\omega^1\omega^3 \in \Omega(\Sigma_1 \cup \Sigma_3),\ \exists\omega^2 \in \Omega(\Sigma_2) : \forall(F|E) \in \Delta_2,\ \omega^1\omega^3\omega^2 \not\models E \wedge \neg F$
  if *safe_delta2* **then**
    safe_delta1 $\leftarrow \forall\omega^2\omega^3 \in \Omega(\Sigma_2 \cup \Sigma_3),\ \exists\omega_1 \in \Omega(\Sigma_1) : \forall(F|E) \in \Delta_1,\ \omega^2\omega^3\omega^1 \not\models E \wedge \neg F$
  if *safe_delta1* $\wedge$ *safe_delta2* **then**
    safeSplittings $\leftarrow$ *safeSplittings* $\cup \{\Delta_1 \bigcup^{\mathsf{s}}_{\Sigma_1,\Sigma_2} \Delta_2 \mid \Sigma_3\}$
return *safeSplittings*

Algorithm 5: `allSplittings`

Input: Belief base Δ, signature Σ
Output: All kinds of conditional syntax splittings of Δ

allSplittings $\leftarrow$ `condSplittings`(Δ, Σ)
genuineSplittings $\leftarrow$ `genuineSplittings`$($*allSplittings*$)$
safeSplittings $\leftarrow$ `safeSplittings`$($*allSplittings*$)$
genuineSafeSplittings $\leftarrow$ `safeSplittings`$($*genuineSplittings*$)$
genSafeSplittings $\leftarrow$ `genSafeSplittings`$($*allSplittings*$)$
genuineGenSafeSplittings $\leftarrow$ `genSafeSplittings`$($*genuineSplittings*$)$
return *allSplittings*, *genuineSplittings*, *safeSplittings*, *genuineSafeSplittings*, *genSafeSplittings*, *genuineGenSafeSplittings*

genuine and safe splitting, 33 generalized safe splittings, and 5 genuine and generalized safe splittings.

Algorithm 6 computes all conditional syntax splittings of Δ exploitable for inductive inference, i.e., the genuine and generalized safe splittings. First, the algorithm computes all conditional syntax splittings of Δ. Then, it removes all simple splittings, leaving only the genuine splittings, and finally checks all genuine splittings for generalized safety. By computing the genuine splittings first, Algorithm 6 minimizes the amount of splittings where the generalized safety property must be validated, being much more efficient than Algorithm 5 when only splittings relevant for inductive inference are of interest.

Algorithm 6: `exploitableSplittings`

Input: Belief base Δ, signature Σ
Output: All splittings of Δ exploitable w.r.t. inductive inference

```
1 allSplittings ← condSplittings(Δ, Σ)
2 genuineSplittings ← genuineSplittings(allSplittings)
3 genuineGenSafeSplittings ← genSafeSplittings(genuineSplittings)
4 return genuineGenSafeSplittings
```

Example 6 (Δ^e, Δ^r cont.). *For Δ^e, Algorithm 6 computes the following two genuine and generalized safe splittings:*

$$\Delta^e = \{(f|b), (\overline{f}|p), (b|p)\} \bigcup_{\{p,f\},\{e\}}^{\text{gs}} \{(b|e)\} \mid \{b\}$$

$$\Delta^e = \{(f|b), (\overline{f}|p), (b|p)\} \bigcup_{\{p\},\{e\}}^{\text{gs}} \{(f|b), (b|e)\} \mid \{b, f\}$$

For Δ^r, Algorithm 6 computes the following five genuine and generalized safe splittings:

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr)\} \bigcup_{\{b\},\{o,u\}}^{\text{gs}} \{(\overline{s}|r), (\overline{r}|s), (o|s\overline{r}), (\overline{o}|r), (u|ro)\} \mid \{s, r\}$$

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr), (o|s\overline{r}), (\overline{o}|r)\} \bigcup_{\{b,s\},\{u\}}^{\text{gs}} \{(\overline{o}|r), (u|ro)\} \mid \{r, o\}$$

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr), (o|s\overline{r}), (\overline{o}|r)\} \bigcup_{\{s\},\{u\}}^{\text{gs}} \{(\overline{o}|r), (u|ro)\} \mid \{b, r, o\}$$

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr), (o|s\overline{r}), (\overline{o}|r)\} \bigcup_{\{b\},\{u\}}^{\text{gs}} \{(\overline{s}|r), (\overline{r}|s), (o|s\overline{r}), (\overline{o}|r), (u|ro)\} \mid \{s, r, o\}$$

$$\Delta^r = \{(\overline{s}|r), (\overline{r}|s), (b|sr)\} \bigcup_{\{b\},\{o\}}^{\text{gs}} \{(\overline{s}|r), (\overline{r}|s), (o|s\overline{r}), (\overline{o}|r), (u|ro)\} \mid \{s, r, u\}$$

In the next section we evaluate the approach developed in this section.

5. Implementation and Evaluation

In this section, we evaluate the results of Algorithm 5 by comparing the different numbers of splittings computed, and we evaluate implementations of all algorithms presented in this paper with respect to their runtime. We evaluate 6 different belief bases with signature sizes between 4 and 14, containing between 4 and 20 conditionals. All belief bases as well as their signature size and belief base size are given in Table 1. The belief bases have been taken from the literature [10, 15, 16] and are available in the online repository CLKR [17].

The number of each different kind of splitting as computed by Algorithm 5 can be found in Table 3. It can be seen that the number of splittings for each belief base grows exponentially in the signature size, already yielding more than 63000 conditional syntax splittings for a signature size of 14. Overall, only a fraction of splittings is exploitable for inductive inference, i.e., the genuine, generalized safe splittings, the maximum being 23% for Δ^{sa} and the minimum being just 2% for the largest example Δ^{aw}. If only genuine, safe splittings are taken into account these numbers are significantly smaller,

Δ	$\lvert\Sigma\rvert$	$\lvert\Delta\rvert$	Signature	Conditionals
Δ^e	4	4	b, p, f, e	$(f\lvert b)$, $(\overline{f}\lvert p)$, $(b\lvert p)$, $(b\lvert e)$
Δ^r	5	6	r, s, b, o, u	$(\overline{s}\lvert r)$, $(\overline{r}\lvert s)$, $(b\lvert sr)$, $(o\lvert s\overline{r})$, $(\overline{o}\lvert r)$, $(u\lvert or)$
Δ^{su}	6	7	r, s, b, o, u, g	$(\overline{s}\lvert r)$, $(\overline{r}\lvert s)$, $(b\lvert sr)$, $(o\lvert s\overline{r})$, $(\overline{o}\lvert r)$, $(u\lvert or)$, $(g\lvert b)$
Δ^{sa}	7	8	m, v, u, o, c, e, k	$(v\lvert m)$, $(c\lvert m)$, $(u\lvert m)$, $(\overline{o}\lvert m)$, $(m\lvert e)$, $(\overline{c}\lvert e)$, $(m\lvert k)$, $(\overline{v}\,\overline{c}\lvert k)$
Δ^{w}	11	12	$w, i, m, v, c, e, k, s, g, b, f$	$(i\lvert w)$, $(w\lvert m)$, $(v\lvert m)$, $(c\lvert m)$, $(m\lvert e)$, $(\overline{c}\lvert e)$, $(m\lvert k)$, $(\overline{v}\,\overline{c}\lvert k)$, $(w\lvert s)$, $(g\lvert s)$, $(s\lvert b)$, $(f\lvert b)$
Δ^{aw}	14	20	$f, a, w, b, d, p, c, s, i, r, l, h, k, m$	$(f\lvert aw)$, $(\overline{f}\lvert a\overline{w})$, $(b \Rightarrow a\lvert\top)$, $(w\lvert b)$, $(d\lvert b)$, $(p \Rightarrow b\lvert\top)$, $(\overline{f}\lvert p)$, $(\overline{f}\lvert p)$, $(c \Rightarrow b\lvert\top)$, $(\overline{f}\lvert c)$, $(f\lvert sc)$, $(\overline{s}\lvert c)$, $(i \Rightarrow a\lvert\top)$, $(r \Rightarrow i\lvert\top)$, $(l \Rightarrow i\lvert\top)$, $(l \vee r\lvert i)$, $(\overline{d}\lvert r)$, $(d\lvert l)$, $(h \Rightarrow r\lvert\top)$, $(f\overline{w}\lvert h)$, $(k \Rightarrow m\lvert\top)$

Table 1
Evaluated belief bases from the literature [10, 15, 16].

Δ	condSplits	safeGenuSplits	genSafeGenuSplits	allKinds	genuSafeSplits	explSplits
Δ^e	0.0006	0.0056	0.0059	0.0140	0.0027	0.0053
Δ^r	0.0015	0.0219	0.0294	0.0545	0.0051	0.0103
Δ^{su}	0.0043	0.0929	0.1441	0.2434	0.0375	0.0725
Δ^{sa}	0.0142	0.2123	0.5569	0.7874	0.0748	0.3707
Δ^{w}	1.6186	51.7818	278.0983	331.5116	36.5009	243.6230
Δ^{aw}	70.6094	3240.6019	19335.6374	22647.1372	1907.3076	18014.1865

Table 2
Runtime (in seconds). The column condSplits shows the time to find all conditional syntax splittings of a belief base (cf. Algorithm 1). The columns safeGenuSplits and genSafeGenuSplits show the runtime of calculating first all safe, and generalized safe splittings respectively (cf. Algorithms 4, 3) and then the genuine splittings (cf. Algorithm 2). The column allKinds shows the runtimes for our implementation of Algorithm 5, computing all kinds of splittings. The columns genuSafeSplits and explSplits show the runtime of computing first the genuine, then the safe respectively generalized safe splittings (cf. Algorithm 6).

leading to a maximum of 9% for Δ^{su} and a minimum of 0% for Δ^r, where no genuine and safe splitting exists. For Δ^{aw} the fraction of splittings that are both genuine and safe is less than 0.3%. Generalized safe splittings vastly outnumber safe splittings for both simple and genuine splittings, reaching up to more than 10 times their amount for some examples, which further corroborates that generalized safe splittings greatly increase the applicability of splitting postulates [10]. Furthermore, the proportion of genuine splittings to all conditional syntax splittings increases with the amount of splittings. The proportion of genuine safe and genuine generalized safe to all safe and generalized safe splittings is lower than the proportion of genuine splittings to all splittings. This shows that filtering for genuine splittings can effectively reduce the amount of splittings to be considered for a notion of safety by around 50% in most cases.

A first implementation of the algorithms in this paper is based on the software library underlying the InfOCF-Web system [18, 19]. We implemented the algorithms using Python 3. The runtimes were measured on an AMD Ryzen 7 PRO 7840U Processor with 32GB working memory.

We measured the runtime of all algorithms presented in this paper. Additionally we implemented and measured the runtime of an algorithm computing the genuine and safe splittings analogous to Algorithm 6. For $\Delta^e, \Delta^r, \Delta^r, \Delta^{su}, \Delta^{sa}$, and Δ^w we took the average runtime across ten different executions of the program. The measured times can be seen in Table 2. Evaluating these runtimes reveals that the computation of the generalized safe splittings (cf. Algorithm 3) takes by far the longest time of all components in Algorithms 6 and 5. Already for small belief bases, computing the genuine and generalized safe splittings takes double the amount of time of computing the genuine and safe splittings, growing up to almost 10 times the runtime for the largest example we considered. This is likely caused by the significantly larger number of generalized safe as opposed to safe splittings which grows similarly. In comparison, the computation of all conditional syntax splittings only took a negligible amount of time. Computing the genuine splittings was almost instantaneous for all belief

Δ	condSplits	safeSplits	genSafeSplits	genuSplits	genuSafeSplits	explSplits
Δ^e	19	5	11	3	1	2
Δ^r	37	16	33	5	0	5
Δ^{su}	90	32	73	26	8	21
Δ^{sa}	231	20	102	103	3	54
Δ^w	7530	166	1835	5482	97	1363
Δ^{aw}	63949	348	2209	47565	174	1249

Table 3
Number of splittings. The column condSplits shows the number of all conditional syntax splittings of the given belief base. The columns safeSplits and genSafeSplits show the number of safe and generalized safe splittings respectively. The column genuSplits shows the number of genuine splittings and the columns genuSafeSplits and explSplits show the numbers of genuine and safe respectively genuine and generalized safe splittings.

bases and is thus not separately considered here. The runtime advantage of genuine splittings is most prominent in smaller examples, leading to a runtime reduction of up to 75% for safe and up to 65% for generalized safe splittings in the case of Δ^r. For the two largest examples, computing the genuine splittings first achieves a runtime reduction of about 35% for safe and 10% for generalized safe splittings. This loss of reduction for the larger examples is likely connected to the overall higher fraction of genuine splittings in the larger examples.

6. Outlook and future work

Ultimately, the goal of splittings as presented in this paper is to enhance inductive reasoning by allowing queries to be solved locally on one of the subbases. This is formalized via the postulates (CRel^g) and (CInd^g). Thus, in order to make use of splittings, an inductive inference operator must satisfy at least one of these postulates. Notably system Z satisfies (CRel^g) but not (CInd^g) while skeptical c-inference and system W satisfy both (CRel^g) and (CInd^g) [10]. From a practical point of view, there are two approaches to utilizing splittings of a belief base Δ for inductive inference: If Δ is static or changes rarely it is reasonable to compute all exploitable splittings of Δ once and then utilize the "best" splitting for each given query. If Δ undergoes regular changes, it is more promising to compute a "good" splitting on the fly when answering queries. In either case what constitutes a "good" splitting is not immediately clear and neither is its computation. This problem is left for future work; the aim of this paper is to provide a first stepping stone for practical applications that can be further refined in future work.

In future work we will perform a more elaborate evaluation with many different belief bases of varying size to investigate empirically if our discoveries can be generalized. We will also investigate how splittings can be exploited for inductive inference in practice by analyzing the computational gains of splittings for implementations of inductive inference operators like system Z, lexicographic inference, c-inference, and system W, available, e.g., in the online reasoning platform InfOCF-Web [18].

7. Conclusion

In this paper, we presented algorithms for computing different kinds of conditional syntax splittings of a belief base and evaluated their implementations. A modular approach to the algorithms ensures that any combination of any kinds of splittings can be easily computed as well, in particular the set of genuine and generalized safe splittings, providing the largest potential for inductive inference. We evaluated the results of our algorithms with respect to the number of splitting kinds computed and also evaluated their runtime. This analysis revealed that, while taking much longer to compute, generalized safe splittings greatly increase the amount of splittings exploitable for inductive inference by up to 10 times as opposed to safe splittings. We have also analyzed the runtime advantage of genuine splittings and seen that, while most effective for smaller examples and safe splittings, they are also relevant for larger examples and generalized safe splittings.

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 512363537, grant BE 1700/12-1 awarded to Christoph Beierle. Lars-Phillip Spiegel was supported by this grant.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] G. Kern-Isberner, C. Beierle, G. Brewka, Syntax splitting = relevance + independence: New postulates for nonmonotonic reasoning from conditional belief bases, in: D. Calvanese, E. Erdem, M. Thielscher (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the 17th International Conference, KR 2020, IJCAI Organization, 2020, pp. 560–571. doi:10.24963/kr.2020/56.

[2] E. W. Adams, The Logic of Conditionals: An Application of Probability to Deductive Logic, Synthese Library, Springer Science+Business Media, Dordrecht, NL, 1975.

[3] M. Goldszmidt, J. Pearl, Qualitative probabilities for default reasoning, belief revision, and causal modeling, Artificial Intelligence 84 (1996) 57–112.

[4] D. Lehmann, Another perspective on default reasoning, Ann. Math. Artif. Intell. 15 (1995) 61–82.

[5] C. Beierle, C. Eichhorn, G. Kern-Isberner, S. Kutsch, Properties and interrelationships of skeptical, weakly skeptical, and credulous inference induced by classes of minimal models, Artificial Intelligence 297 (2021) 103489. doi:10.1016/j.artint.2021.103489.

[6] C. Komo, C. Beierle, Nonmonotonic inferences with qualitative conditionals based on preferred structures on worlds, in: U. Schmid, F. Klügl, D. Wolter (Eds.), KI 2020: Advances in Artificial Intelligence - 43rd German Conference on AI, volume 12325 of *LNCS*, Springer, 2020, pp. 102–115. doi:10.1007/978-3-030-58285-2_8.

[7] C. Komo, C. Beierle, Nonmonotonic reasoning from conditional knowledge bases with system W, Ann. Math. Artif. Intell. 90 (2022) 107–144. doi:10.1007/s10472-021-09777-9.

[8] R. Parikh, Beliefs, belief revision, and splitting languages, Logic, Language, and Computation 2 (1999) 266–278.

[9] J. Heyninck, G. Kern-Isberner, T. Meyer, J. P. Haldimann, C. Beierle, Conditional syntax splitting for non-monotonic inference operators, in: B. Williams, Y. Chen, J. Neville (Eds.), Proceedings of the 37th AAAI Conference on Artificial Intelligence, volume 37, 2023, pp. 6416–6424. doi:10.1609/aaai.v37i5.25789.

[10] L.-P. Spiegel, J. Haldimann, J. Heyninck, G. Kern-Isberner, C. Beierle, Generalized safe conditional syntax splitting of belief bases, in: J. Kwok (Ed.), Proceedings of the 34th International Joint Conference on Artificial Intelligence, IJCAI-25, IJCAI Organization, 2025, pp. 4678–4686. doi:10.24963/ijcai.2025/521.

[11] J. P. Delgrande, A knowledge level account of forgetting, J. Artif. Intell. Res. 60 (2017) 1165–1213. URL: https://doi.org/10.1613/jair.5530. doi:10.1613/jair.5530.

[12] W. Spohn, Ordinal conditional functions: a dynamic theory of epistemic states, in: W. Harper, B. Skyrms (Eds.), Causation in Decision, Belief Change, and Statistics, II, Kluwer Academic Publishers, 1988, pp. 105–134.

[13] J. Pearl, System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning, in: Proc. TARK'1990, Morgan Kaufmann, 1990, pp. 121–135.

[14] C. Beierle, L.-P. Spiegel, J. Haldimann, M. Wilhelm, J. Heyninck, G. Kern-Isberner, Conditional splittings of belief bases and nonmonotonic inference with c-representations, in: P. Marquis,

M. Ortiz, M. Pagnucco (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the 21st International Conference, KR 2024, 2024, pp. 106–116. doi:10.24963/kr.2024/10.

[15] J. Haldimann, A. Osiak, C. Beierle, Modelling and reasoning in biomedical applications with qualitative conditional logic, in: U. Schmid, F. Klügl, D. Wolter (Eds.), KI 2020: Advances in Artificial Intelligence - 43rd German Conference on AI, volume 12325 of *LNCS*, Springer, 2020, pp. 283–289. doi:10.1007/978-3-030-58285-2_24.

[16] M. Wilhelm, D. Howey, G. Kern-Isberner, K. Sauerwald, C. Beierle, Activation-based conditional inference, Journal of Applied Logics 10 (2023) 221–248. URL: https://collegepublications.co.uk/downloads/ifcolog00058.pdf.

[17] C. Beierle, J. Haldimann, L. Schwarzer, CLKR – Conditional Logic and Knowledge Representation, KI – Künstliche Intelligenz 38 (2024) 61–67. doi:10.1007/s13218-024-00842-z.

[18] C. Beierle, J. Haldimann, A. Sanin, L. Schwarzer, A. Spang, L. Spiegel, M. von Berg, Scaling up reasoning from conditional belief bases, in: S. Destercke, M. V. Martinez, G. Sanfilippo (Eds.), Scalable Uncertainty Management - 16th International Conference, SUM 2024, Proceedings, volume 15350 of *LNCS*, Springer, 2024, pp. 29–44. doi:10.1007/978-3-031-76235-2_3.

[19] C. Beierle, J. Haldimann, A. Sanin, A. Spang, L. Spiegel, M. von Berg, The InfOCF library for reasoning with conditional belief bases, in: G. Casini, B. Dundua, T. Kutsia (Eds.), Logics in Artificial Intelligence, 19th European Conference (JELIA 2025), Proceedings, Part II, volume 16094 of *LNAI*, Springer, 2025, pp. 19–27. doi:10.1007/978-3-032-04590-4_2.