Provability in BI’s Sequent Calculus is Decidable

ALEXANDER GHEORGHIU, SIMON DOCHERTY, and DAVID PYM, University College London, United Kingdom

The logic of Bunched Implications (BI) combines both additive and multiplicative connectives, which include two primitive intuitionistic implications. As a consequence, contexts in the sequent presentation are not lists, nor multisets, but rather tree-like structures called bunches. This additional complexity notwithstanding, the logic has a well-behaved metatheory admitting all the familiar forms of semantics and proof systems. However, the presentation of an effective proof-search procedure has been elusive since the logic’s debut. In this paper we show that one can reduce the proof-search space for any given sequent to a primitive recursive set, an argument generalizing Gentzen’s decidability proof for classical propositional logic and combining key features of Dyckhoff’s contraction-elimination argument for intuitionistic logic. An effective proof-search procedure, and hence decidability of provability, follows as a corollary.

CCS Concepts: • Theory of computation → Proof theory: Logic and verification; Separation logic; Problems, reductions and completeness: Complexity theory and logic; • Mathematics of computing → Combinatorial algorithms; • Computing methodologies → Symbolic calculus algorithms; Theorem proving algorithms.

Additional Key Words and Phrases: Logic, Proof-search, Sequent Calculus, BI, Decidability

1 INTRODUCTION

This paper contains a mistake in Section 4.2. Perhaps some of the ideas presented within are of use to others.

The logic of Bunched Implication (BI) [21] is a logic in which multiplicative and additive intuitionistic implications live side-by-side. Consequently, there are two different types of data (additive and multiplicative), which are distinguished by introducing two different context-formers: multiplicative composition \((\Delta, \Gamma)\) denies the structural rules of weakening and contraction, whereas additive composition \((\Delta /_{\text{fatsemi}} \Gamma)\) admits them. It follows that contexts in BI’s sequents are not a flat data structure, such as lists or multisets, but instead are tree structures called bunches, a term that derives from the relevance logic literature (see, for example, Read [24]). The uniqueness of BI lies in the presence of two primitive implications (additive \(\to\) and multiplicative \(\multimap\)); the logic arises naturally from investigating the proof-theoretic relationship between conjunction and implication [21].

A natural question to ask about a logic is whether or not logical consequence is decidable. The approaches to the problem can be partitioned into two broad categories, the syntactic and the semantic, which are characterized by making use of the proof theory and model theory of the logic, respectively.
logic respectively. In this paper, we take a purely syntactic approach, based on proof-search in BI’s sequent calculus $\text{LBI}$ [22]. Although BI enjoys the full gamut of Hilbert, natural deduction, tableaux, and display calculi [3, 11, 22], we choose the sequent calculus because it has local correctness and analytic proofs are complete, meaning that one can structure the proof-search space for any given sequent through combinatorics on the sub-formulas appearing in it. For example, the following naive procedure for generating a proof-search space would certainly produce all the sequents present in a proof of a given sequent if there is one:

- combine multisets of formulas and sub-formulas using additive context-formers to form a zeroth generation of bunches;
- combine multisets of previously constructed bunches using multiplicative context-formers to form a next generation of bunches; and,
- combine multisets of previously constructed bunches using additive context-formers to complete the next generation of bunches.

Unfortunately, the process is infinite, yielding an infinite space, and is therefore unusable for proof-search. Yet, if one could restrict to a bounded search (i.e., to a computation using only primitive recursive functions), then a proof-search procedure would follow immediately by arranging the constructed sequents into proof candidates and checking correctness. This is the essence of Gentzen’s decidability argument for classical propositional logic (CPL) [12], and it is the approach to decidability taken in this paper. Here the boundedness is taken literally: the maximum size of multisets combined at each stage in the outlined construction is bounded and so is the number of generations considered.

To show that bounded search is adequate, Gentzen’s proof for CPL proceeds by demonstrating the completeness of a class of proofs in which every formula occurs at most three times in any sequent. There are two insights that make this work: first, that there is a uniform bound for the increase in the multiplicity of formulas in the conclusion from the premises with respect to a single inference step in the proof calculus; second, that after each deductive inference step one can use the structural rules to reduce the resulting sequent by removing extraneous occurrences, thereby maintaining a bound for the multiplicity. In this paper, we call the first insight the bounded growth property (BGP), and regard the second insight as defining a class of proofs, dubbed regimented, that have the property of stopping the growth in multiplicity from compounding, allowing for a uniform bound on proofs.

The bunched structure of BI’s sequents means that the situation is more complex than for CPL since there are now three measures to control, each corresponding to one aspect of the naive construction above: additive width (size of additive combinations in the even steps), multiplicative width (size of multiplicative combinations in the odd steps), and depth (number of steps). Moreover, the extra complexity afforded by, and interactions of, the additive and multiplicative structures renders the notions of extraneous and reduction quite delicate. In particular, under intuitive readings of these terms, LBI does not satisfy the BGP due to the behaviour of weakening, and depth cannot be bounded due to the certain atomic structures. Thus the argument in this paper has two key steps: we first restrict the structural rules to work in a system that has the BGP, and then we eliminate the rules that allow the atomic structures to exhibit bad behaviours, these rules are a type of contraction and their removal is in the spirit of Dyckhoff’s contraction-elimination argument for intuitionistic logic [9] (i.e., we embed a sufficient amount of them into the logical rules so as to render them impotent); the resulting system is called $d\text{LBI}$. It is over this system that the decision procedure takes place.

The argument begins in Section 2 by defining the logic of Bunched Implications, along relevant auxiliary terminology used later on.
The study of control in BI’s sequent calculi begins in Section 3 with a formal characterization of reduction for controlling the measures. It here that dLBI is defined as it makes use of the associated notion of a normal (i.e., fully reduced) bunch; the system arises from LBI by requiring that the structural rules are small-step in the sense that they can only introduce a bounded amount of extraneous data, and that the unit laws for each context-former, thought of as contractions, are no longer present.

Formal measures for additive width, multiplicative width, and depth are given in Section 4 where we also establish uniform bounds for these measures with respect to regimented proofs in dLBI. An effective proof-search procedure, and decidability, follows immediately.

The paper concludes in Section 5 with a summary of the argument, a discussion of related results, and a consideration of future work.

2 THE LOGIC OF BUNCHED IMPLICATIONS

2.1 Syntax

As a logic, BI can be regarded as the free combination of (additive) intuitionistic logic (IL) and multiplicative intuitionistic logic (MILL), which results in the presence of two distinct context-formers in its sequent presentation; that is, the two conjunctions ∧ and ∗ are represented at the meta-level by context-formers ; and , in place of the usual commas for IL and MILL, respectively.

Definition 2.1 (Formula). Let A be a denumerable set of propositional letters. The formulas of BI are defined by the following grammar:

\[ \phi ::= \top | \bot | \top^* | A \in A | (\phi \wedge \phi) | (\phi \vee \phi) | (\phi \rightarrow \phi) | (\phi \ast \phi) | (\phi \leftarrow \phi) \]

If o ∈ {∧, ∨, →, ⊤, ⊥}, then it is an additive connective, and if o ∈ {∗, ←, T∗} then it is a multiplicative connective. The set of all formulas is denoted F.

Definition 2.2 (Bunch). A bunch is constructed from the following grammar:

\[ \Delta ::= \phi \in F | \emptyset_+ | \emptyset_- | (\Delta ; \Delta) | (\Delta , \Delta) \]

The symbols \emptyset_+ and \emptyset_- are the additive and multiplicative units, respectively, and the symbols ; and , are the additive and multiplicative context-formers, respectively. The set of all bunches is denoted B. A bunch is basic if it is a formula, \emptyset_+, or \emptyset_-, and complex otherwise. The context-former at the root of a complex bunch is its principal context-former. The set of complex bunches with additive and multiplicative principal context-formers are denoted B+ and B×, respectively.

Definition 2.3 (Sub-bunch). For any bunches Δ and Γ, let Δ ⪯ Γ denote that Δ is a proper sub-tree of Γ, and let Δ ⪯∗ Γ denote that either Δ ⪯ Γ or Δ = Γ. A bunch Δ is a sub-bunch of a bunch Γ when Δ ⪯ Γ.

We may write Γ(Δ) to mean that Δ is a sub-bunch of Γ and define Γ[Δ ↦ Δ′] — abbreviated to Γ(Δ′) where no confusion arises — be the result of replacing the occurrence of Δ by Δ′.

Definition 2.4 (Sequent). A sequent is a pair of a bunch Γ, called the context, and a formula φ, called the exact, and is denoted Γ ⇒ φ.

Since each of the context-formers of a bunch represents a comma in IL or MILL, they have the same properties; in particular, they may each be read as a multiset constructor instead of a list constructor, witnessed in the proof theory by the presence of an exchange rule.

Definition 2.5 (Permutation Equivalence, Coherent Equivalence). Two bunches Γ, Γ′ ∈ B are equivalent up to permutation when Γ ≡ Γ′, where ≡ is the least equivalence relation on bunches satisfying:
Definition 2.6 (Additive Data). Let $\Delta_1, \Delta_2 \leq \Gamma \in \mathbb{B}$ both be multiplicative bunches or atomic bunches. The relation $\Delta_1 \sim \Delta_2$ holds when the principal context-formers of $\Delta_1$ and $\Delta_2$ are not above one another and there is a path in the parse-tree of $\Gamma$ between them passing through only additive context-formers.

The equivalence classes of sub-bunches of $\Gamma$ under $\sim$ are the additive sets of the bunch. The additive component of a sub-bunch in $\Gamma$ is the least sub-bunch in $\Gamma$, with respect to the sub-bunch relation, that contains every member of its additive set.

The elements contained in additive sets are additive data.

Example 2.7. In Figure 1, a bunch together with its additive sets and components is presented. The additive sets are determined by choosing some additive data and finding all other additive data that can be reached when passing through only additive context-formers.

2.2 Proof Theory

In this paper, BI-truth, the predicate defining the logic of bunched implications, is defined by provability in the following sequent calculus:

Definition 2.8 (System LBI). The sequent calculus LBI is composed of the rules in Figure 2.

We may regard a rule $r$ as a relation $r(P_1, ..., P_n, C)$ that holds if and only if the conclusion $C$ may be inferred from the premisses $P_1, ..., P_n$ by the rule.

Though the usual inductive definition of proofs suffices, some extra flexibility in the meta-language is useful, thus we forego the usual definition, doubtlessly familiar, and use a correctness condition of a tress of sequents instead:
Definition 2.9 (Derivation). Let L be a sequent calculus, let \( \mathcal{A} \) be a set of sequents, and let \( S \) be some sequent. A rooted finite tree \( D \) of sequents is a L-derivation of \( S \) from \( \mathcal{A} \) if, for any node \( \zeta \), the following hold:

- if \( \zeta \) is a leaf, then either \( \zeta \in \mathcal{A} \) or there is \( A \in L \) such that \( A(\zeta) \) holds;
- if \( \zeta \) has children \( P_0, ..., P_n \) in \( D \), then there is a rule s.t. \( r(P_0, ..., P_n, \zeta) \); and
- if \( \zeta \) is the root, then \( \zeta = S \).

Remark. Derivation allows us to talk about sections of proofs which may then be replaced by other sections with the same leaves and root but which have a particular desirable structure.

The foregoing provides all the requisite apparatus to define BI:

Definition 2.10 (BI-truth). A sequent \( S \) is a BI-truth iff it admits a LBI-proof.

We may write \( D : \mathcal{A} \vdash_S S \) to denote that \( D \) is a L-derivation of \( S \) from \( \mathcal{A} \), or simply a L-proof of \( S \) when \( \mathcal{A} = \emptyset \); we will suppress the set constructors when there is no confusion. Dropping the reference to the derivation (i.e., writing \( \mathcal{A} \vdash_S S \)) asserts that there is a proof witnessing the judgment (i.e., there is \( D \) such that \( D : \mathcal{A} \vdash S \)). The resulting relation \( \vdash_{LBI} \) is the provability relation (i.e., a sequent \( S \) is a BI-truth iff \( \vdash_{LBI} S \)), this paper shows that the relation is decidable; that is, that its characteristic function is computable.

A crucial simplification to provability as given by LBI is possible: we may restrict attention to analytic proofs, which was shown to be adequate by Brotherston [3], and Gheorghiu and Marin [13].

Lemma 2.11 (cut-admissibility). If \( D : \mathcal{A} \vdash_{LBI} S \), then there is \( D' : \emptyset \vdash_{LBI} S \) without cut.

To expose certain features of the exchange rule \( e \), we will appeal to a variant of LBI:

Definition 2.12 (Sequent Calculus LBI'). Sequent calculus LBI' is composed of all the rules of Figure 2 except cut, replacing \( e \) by the rule in Figure 3.

Remark. The \( e' \) rule uses permutation equivalence, whereas \( e \) uses coherent equivalence.
The occurrences of sub-bunches and formulas in sequents in \( \text{lBI} \) deliver decidability of provability. We introduce the requisite tools for subsequent discussion and then show that these tools permit a problem reduction from the decidability of provability in \( \text{lBI} \), and this follows from the additional rules in the latter system: the relation \( \Gamma \equiv \Gamma' \) holds iff there is a transformation of the premise to the conclusion using the commutative monoid equations, but these equations are given by \( \Delta^+ \), \( \Delta_c \), \( \text{w}_{\phi_x} \), \( \text{w}_{\phi_c} \), and \( e' \). 

Not all of the data (i.e., formulas and sub-bunches) present in a sequent is material in a single inference since the rules act locally on particular structures within the premises and conclusion; it is this local behaviour that will be controlled later. Here the notion of what is being operated on is made precise:

**Definition 2.14 (Active).** Let \( r \) be a rule in a sequent calculus, and consider an inference \( r(P_1, ..., P_n, C) \). The occurrences of sub-bunches and formulas in sequents \( P_i \) and \( C \) that instantiate meta-variables \( \Delta, \Delta', \Delta'', \phi \) or \( \psi \) in the rule figure defining \( r \) are active in the inference.

**Example 2.15.** Consider the following instance of weakening, the \( \phi \) is active in the premiss, and both the \( \phi \) and \( \psi \) are active in the conclusion:

\[
\emptyset_+, \phi \equiv \chi \\
\emptyset_+, (\phi, \psi) \equiv \chi
\]

**Remark.** The definition of active bunch in an inference will be extended later, such as in Figure 5 where it includes instantiations of the metavariable \( \Sigma \).

## 3 THE CONTROL RÉGIME

This section contains the specific metatheory of \( \text{BI} \) delivering decidability of provability. We introduce the requisite tools for subsequent discussion and then show that these tools permit a problem reduction from the decidability of provability in \( \text{BI} \) to the decidability of \textit{regimented provability} in \( \text{dBI} \), the definition contained within. The decidability of the latter is subject of Section 4.

Section 3.1 concerns the first tool required: an analogue of Gentzen’s reduction relation on sequents that can be internalized by the structural rules. The more complex structure of bunched contexts over list or multisets results in a more complex definition; for example, it is not only occurrences of formulas that need to be controlled, but entire sub-bunches. Moreover, reduction must satisfy certain computational conditions such a \textit{confluence} and \textit{normalization}, which are established within.

Unfortunately, Gentzen’s control régime alone is insufficient to yield a primitive recursive subset of the search space for \( \text{BI} \). There are two reasons: firstly, \( \text{BI'} \) lacks the BGP; and, secondly, in the presence of \( c_{\phi_+} \) and \( c_{\phi_-} \) there is no bound on the number of generations in the naive construction given in Section 1. This motivates the introduction of another system, \( \text{dBI} \), which has \textit{small-step} structural rules and in which \( c_{\phi_+} \) and \( c_{\phi_-} \) are no longer present, that is equivalent to \( \text{BI} \). This is
the subject of Section 3.2, and may be regarded as the second tool used to deliver the decidability claim of BI.

Having defined the system in which proof-search will happen, Section 3.3 concerns the third tool required for the decidability problem: Gentzen’s control régime, which manifests as a class of proofs following a particular structure dubbed *regimented* proofs.

Finally, Section 3.4 shows the soundness and completeness of regimented proofs in dLBI. The culmination of this section as a whole is Theorem 3.34 of this subsection, which enables the problem reduction delivery decidability.

### 3.1 Reduction and Normality

We establish here some computational and proof-theoretic requirements that the reduction of sequents, which proceeds by acting on the contexts, must satisfy. However, since the structural rules are subsequently restricted to form dLBI, the *modus operandi* for the control régime delivering proof-search, we must similarly restrict the reduction, resulting in *small-step* reduction.

**Definition 3.1 (Reduction and Normality of Bunches).** Let \( \Gamma, \Gamma' \in \mathbb{B} \) be arbitrary. Big-step reductions are defined as follows:

\[
\Gamma \geq \Gamma' \iff \begin{cases} 
\Gamma \equiv \Gamma' \\
\Gamma = \Gamma(\Delta \vdash \varnothing_x) \text{ and } \Gamma' = \Gamma(\Delta) \\
\Gamma = \Gamma(\Delta \vdash \varnothing_+) \text{ and } \Gamma' = \Gamma(\Delta) \\
\Gamma = \Gamma(\Sigma \vdash \varnothing) \text{ and } \Gamma' = \Gamma(\Sigma) 
\end{cases}
\]

A bunch \( \Gamma \) is normal iff further reductions are all permutations; that is, for any \( \Gamma'' \), if \( \Gamma \geq \Gamma'' \geq \Gamma' \), then \( \Gamma \equiv \Gamma' \). The set of normal bunches is denoted \( \mathbb{B} \).

A big-step reduction is a small-step reduction, denoted \( \Gamma \Rightarrow \Gamma' \), when \( \Sigma \) is normal and multiplicative, or basic. The transitive closure of small-step reduction is denoted \( \Rightarrow^* \), small-step reduction adorned with an asterix.

**Remark.** By definition, for any bunch \( \Gamma \), one has \( \Gamma \Rightarrow \Gamma \), thus the transitive closure of reduction is also reflexively closed.

**Remark.** Each component in the definition of reduction is a contraction law of LBI’.

**Example 3.2.** Let \( \Gamma = (\phi, (\chi \vdash \varnothing_+)) \vdash (\psi \vdash (\varnothing_x \vdash \psi)) \), one can normalize it by first permuting \( \psi \) and \( \varnothing_x \), removing one of the \( \psi \), and then removing the additive unit; that is,

\[
\Gamma \Rightarrow ((\phi, (\chi \vdash \varnothing_+)) \vdash (\varnothing_x \vdash (\psi \vdash \psi))) \Rightarrow ((\phi, (\chi \vdash \varnothing_+)) \vdash (\varnothing_x \vdash \psi)) \Rightarrow ((\phi, (\chi) \vdash (\varnothing_x \vdash \psi))
\]

**Remark.** The concept of a big-step reduction may be dropped. They are introduced in order to define normal bunch for the definition of small-step reductions. Big-step reductions can in many cases be simulated by small-step reductions. This is the moral of Lemma 3.26, which shows the completeness of LBI’ with a *small-step* sequent calculus. Henceforth, therefore, reduction will refer to small-step reduction.

A reduction \( \Gamma \geq \Gamma' \) is proper, denoted \( \Gamma \succ \Gamma' \) and \( \Gamma \succ \Gamma' \) for big-step and small-step reductions, respectively, if it is not a permutation (i.e., if \( \Gamma \not\equiv \Gamma' \)).

**Definition 3.3 (Reduction and Normality of Sequents).** The small-step reduction relation extend to sequents by acting on their contexts; that is,

\[
(\Gamma \Rightarrow \phi) \Rightarrow (\Gamma' \Rightarrow \phi') \text{ iff } \Gamma \Rightarrow \Gamma' \text{ and } \phi = \phi'
\]

A sequent is normal when its context is normal.
The reflexive and transitive closure of class reduction is

\[ \Gamma \leadsto^* \Gamma' \]

the following lemmas about bunches are equally valid for sequents.

\[ \Gamma \leadsto^* \Delta \]

such that

\[ \Gamma \leadsto \Delta \]

is normal bunch or sequent.

\[ \Gamma \leadsto \Delta \]

either a permutation or proper, thus

\[ \Gamma \leadsto \Delta \]

induction hypothesis, the reduct has a normal form. Hence, by transitivity, \( \Gamma \) has a normal form.

\[ \Gamma \leadsto \Delta \]

3.1.1 Confluence. We will not fix a reduction pattern, but require that the eventual normal form can be freely chosen; hence, it is essential that reduction is confluent.

\[ \Gamma \leadsto \Delta \]

Def. 3.5 (Local and Global Confluence). A relation \( \Downarrow \) with reflexive and transitive closure \( \Downarrow \) is said to be locally confluent (resp. globally confluent) if whenever \( x \Downarrow u \) and \( x \Downarrow v \) (resp. \( x \Downarrow u \) and \( x \Downarrow v \)) there is \( y \) such that \( u \Downarrow y \) and \( v \Downarrow y \).

Global confluence (henceforth: confluence) follows from local confluence for terminating relations. In this study we will work with equivalence classes of bunches to manage the permutation steps independently from the proper reductions. For any \( \Gamma \in \mathbb{B} \) denote its equivalence class under permutation by \( [\Gamma] \) (i.e., \( [\Gamma] := \{ \Gamma' \in \mathbb{B} \mid \Gamma' \equiv \Gamma \} \)).

Def. 3.6 (Class Reduction). Let \( \mathcal{G}, \mathcal{G}' \in \mathbb{B}/\equiv \) then class reduction is defined as follows:

\[ \mathcal{G} \hookrightarrow \mathcal{G}' \iff \exists \Gamma', \Gamma'' \in \mathbb{B} \text{ such that } (\Gamma \in \mathcal{G}) \text{ and } (\Gamma' \in \mathcal{G}') \text{ and } (\Gamma > \Gamma') \]

The reflexive and transitive closure of class reduction is \( \hookrightarrow^* \).

\[ \Gamma \hookrightarrow^* [\Gamma'] \iff \Gamma \rightarrow^* \Gamma' \]

Proof. The claim follows by unpacking definitions. If \( [\Gamma] \hookrightarrow^* [\Gamma'] \), then there are \( \Delta \) and \( \Delta' \) such that \( \Gamma \equiv \Delta < \Delta' \equiv \Gamma' \), but each step is a small-step reduction, thus \( \Gamma \rightarrow^* \Gamma' \). Similarly, if \( \Gamma \rightarrow^* \Gamma' \), then there is some sequence of small-step reductions witnessing it each of which is either a permutation or proper, thus \( [\Gamma] \hookrightarrow^* [\Gamma'] \).

\[ \Gamma \hookrightarrow^* [\Gamma'] \]

Lemma 3.7. Class reduction is locally confluent.

Proof. The claim may be unpacked as follows: if \( \Gamma \equiv A \equiv B \) and \( A \geq A' \) and \( B \geq B' \), then there is \( C \) such that \( A' \geq A' \geq C \less B' \). Let \( \alpha \) and \( \beta \) be the sub-bunched removed for the reductions of \( A \) and \( B \), respectively; the proof proceeds by case analysis on the relationship of \( \alpha \) and \( \beta \). There are two
such cases, either $\alpha$ and $\beta$ are in sub-bunch relation, or they are independent — they cannot have a common component since they are multiplicative and so their components are separated.

First, the cases where $\alpha$ and $\beta$ are in sub-bunch relation:

- $\alpha = \beta$. It follows that $A' \equiv B'$ and the case is trivial.
- $\alpha \prec \beta$. Since $\beta$ was removed in the reduction of $B$, it must share a context-former with a $\beta'$, which is syntactically the same bunch, in $\Gamma$. Moreover, this $\beta'$ is present in both $A'$ and $B'$ after suitable permutation. Perform the same reduction on $\beta'$ as in $\beta$ in $A'$, then the resulting sub-bunches are once more equivalent and one may thus remove $\beta$ yielding a bunch $C$. This bunch is similarly reached from $B'$ by performing the same reduction in $\beta'$ as was done in $\beta$.
- $\beta \prec \alpha$. Mutatis mutandis — this is the same as the preceding case but interchanging $\alpha$ and $\beta$.

Second, the cases where $\alpha$ and $\beta$ are independent sub-bunches in $A$ and $B$; that is, $\Gamma = \Gamma(\Pi_1 \odot \Pi_2)$ such that, without loss of generality, $A = \Gamma(\Pi_1' \odot \Pi_2')$ and $B = \Gamma(\Pi_1'' \odot \Pi_2'')$, where $\Pi_1 \equiv \Pi_1' \equiv \Pi_1''$ and $\Pi_2 \equiv \Pi_2' \equiv \Pi_2''$, and $\alpha \leq \Pi_1'$ and $\beta \leq \Pi_2''$.

- $\alpha \equiv \Pi_1'$ and $\beta \equiv \Pi_2''$. If either $\alpha$ or $\beta$ is a unit, then the other one must be too; moreover, the unit type must match the type of the context-former. It follows that $A' = B'$ and the case is satisfied. Suppose then that neither $\alpha$ nor $\beta$ is a unit, then it must be that the context-former is additive and that $\Pi_1' \equiv \Pi_1''$, hence the case is satisfied after permutation.
- $\alpha \prec \Pi_1'$ and $\beta \equiv \Pi_2''$. If $\beta$ is a unit, then its type must be the same as that of its connecting context-former. Hence $\Pi_2''$ is equal to $\Pi_2''$, and the bunch may be removed in $A'$ yielding a bunch $C$. This bunch is reached from $B'$ by permuting $\Pi_1'$, which is still present, to $\Pi_1'$, and then removing $\alpha$.
- $\alpha \equiv \Pi_1'$ and $\beta \prec \Pi_2''$. Mutatis mutandis — this is the same as the preceding case but interchanging $\alpha$ and $\beta$.
- $\alpha \prec \Pi_1'$ and $\beta \prec \Pi_2''$. It must be that $A' = \Gamma(\hat{\Pi}_1' \odot \Pi_2')$, where $\Pi_1' \succ \hat{\Pi}_1'$ by removal of $\alpha$, and $B' = \Gamma(\Pi_1'' \odot \hat{\Pi}_2')$, where $\Pi_2' \succ \hat{\Pi}_2'$ by removal of $\beta$. It follows that $A' \simeq \Gamma(\Pi_1' \odot \Pi_2'') \prec B'$.

Lemma 3.9. Class reduction is confluent.

Proof. The size of bunches (i.e., the number of context-formers) is invariant under permutation; so, if $G \prec \prec G'$, then all members of $G''$ have strictly fewer context-formers than the bunches in $G$. It follows that class reduction is terminating; hence, by Newman’s Lemma [20] and Lemma 3.8, class reduction is confluent.

Lemma 3.10. Reduction is confluent.

Proof. Suppose $\Gamma \equiv \Gamma'$ and $\Gamma \equiv \Gamma''$, then, by Lemma 3.7, we have $[\Gamma] \equiv [\Gamma']$ and $[\Gamma] \equiv [\Gamma'']$. By Lemma 3.9, we have $\exists \Gamma''$ satisfying $[\Gamma'] \equiv [\Gamma''] \equiv [\Gamma'']$. Confluence follows by a final use of Lemma 3.7.

3.1.2 Strategies. Reduction is instantiated in the sequent calculus by the structural rules, sequences of reduction are represented by strategies. Here the consideration of LBI’ over LBI becomes pertinent as it extracts the implicit contractions (i.e., $c_{\odot}$ and $c_{\otimes}$) within coherent equivalence.

Definition 3.11 (Strategy). A derivation $D : S \vdash S'$ is said to be a positive (resp. negative) strategy if every inference is either an instance of a contraction (resp. weakening) rule or exchange. A positive strategy is normalizing if and only if $S'$ is normal.

Remark. It is not full weakening that is required to define (negative) strategies but only the inverse of contraction, sometimes called mingle [14].
LEMMA 3.12. There is a sequence \( \sigma = (S_1, \ldots, S_n) \) of sequents such that \( S_1 = S, S_n = S' \), and \( S_i \triangleright S_{i+1} \) if and only if there is a positive LBI’-strategy \( D : S \vdash S' \) in which the \( (i, j) \) parent-child pair is \( (S_i, S_j) \). Furthermore, if there is a positive strategy \( D : S \vdash S' \), then there is a negative strategy \( D : S' \vdash S \) containing the same sequents.

PROOF. The first claim follows from observing that the conditions defining reduction describe instances of the structural rules. The second claim is immediate since a reverse reading of an instance of contraction is an instance of weakening. \( \square \)

Denote \( D_\sigma : S \triangleright^* S' \) for the positive strategy corresponding to the sequence of reductions \( \sigma \), and \( \bar{D}_\sigma : S \triangleright^* S' \) for the associated negative strategy. A reduction sequence \( \sigma \) for \( \Delta \triangleright^* \Delta' \), determines a reduction sequence for \( \Gamma(\Delta) \triangleright^* \Gamma(\Delta') \) since the relation applies on any sub-bunch. Denote \( \bar{D} : S \triangleright^* S' \) if there is \( \sigma \) such that \( \bar{D} = \bar{D}_\sigma \).

The invertibility of reduction allows the proof-search problem to be reduced to considering normal forms:

LEMMA 3.13. \( \vdash_{\text{LBI}} S \iff \vdash_{\text{LBI'}} \bar{S} \).

PROOF. Each direction follows immediately from Lemma 3.12. \( \square \)

3.2 System dLBI

Gentzen’s control régime is not implemented in LBI’, but rather in a variant that has the BGP and in which the number of generations required during the naive construction of the proof-search space in Section 1 can be bounded. The system arises from LBI’ by restricting all the rules to instances in which only a bounded number of sub-bunches that already occur in the context of the sequent can be introduced and by eliminating \( c_{\varnothing_x} \) and \( c_{\varnothing_y} \). This system is dLBI. As we will see in Section 4, the first move is what makes dLBI have the BGP, and the second is what allows the number of generations of the naive construction to be controlled.

Definition 3.14 (System dLBI). System dLBI is composed of the rules in Figure 5, where \( \Sigma \) is a normal and multiplicative bunch. In any instance of a rule except for the \( \perp_{\text{LBI}} \)-rule, any sub-bunch instantiating \( \Sigma \) is deemed active.

Remark. Since \( c_{\varnothing_x} \) and \( c_{\varnothing_y} \) are no longer available in dLBI, a positive strategy in dLBI does not include reductions of the form \( \Gamma(\Delta, \varnothing_x) \triangleright \Gamma(\Delta) \) or \( \Gamma(\Delta, \varnothing_x) \triangleright \Gamma(\Delta) \). Though seemingly limiting, the additional rules in dLBI are designed to make the restriction adequate.

An important feature of dLBI distinguishing it from LBI and LBI’ are the additional logical rules \( \rightarrow_i \) and \( \neg_i \) and \( *_{R_j} \) for \( i \in \{1, 2, 3\} \) and \( j \in \{1, 2\} \), and the additional structural rule \( \text{conv} \). These rules capture the need for unit bunches in the formulation of BI as empty bunches ensuring that sequents are well-formed. The remainder of this section discusses how and why the additional rules of dLBI arise from intuitive readings of rules of LBI and LBI’. Consider a putative conclusion \( \Gamma \iff \phi * \psi \). There are three intuitive possibilities for decomposing the extract: either \( \phi \) is a BI-truth, \( \psi \) is a BI-truth, or \( \Gamma = \Delta, \Delta' \) such that \( \phi \) and \( \psi \) follow from \( \Delta \) and \( \Delta' \), respectively. The \( *_{R} \)-rule in LBI and LBI’ only directly captures the third option, decomposition of the context, with the remaining two captured only by first setting-up the bunch using a structural rule (i.e., \( c_{\varnothing_x} \)) in the reductive reading:

\[
\begin{align*}
\Gamma & \rightarrow \phi & \varnothing_x & \rightarrow \psi & *_{R} & \varnothing_x \rightarrow \phi * \psi & c_{\varnothing_x}, e \\
\Gamma & \rightarrow \phi * \psi & c_{\varnothing_x}, e
\end{align*}
\]
Provability in BI’s Sequent Calculus is Decidable

Fig. 5. Sequent Calculus dLBI
In both cases the multiplicative unit is precisely used to represent an empty bunch ensuring that the sequents of both branches are well-formed according to the specification of $*_{R}$, but it is not otherwise significant. The $*_{1}$ and $*_{2}$ rules are designed so that this use of the structural rule $c_{\varnothing_{x}}$ can be avoided without loss of completeness; that is, the above $LBI^{'}$-derivations can be simulated in $dLBI$ by a single inference. Indeed, $\Gamma$ may be a unit too, in which case $*_{1}$ and $*_{2}$ collapse into one rule:

**Example 3.15.** The following is a $dLBI$-proof, with both $i = 1$ or $i = 2$ being a valid inference:

$$\varnothing_{x} \implies T^{*} \quad \varnothing_{x} \implies T^{*} \quad *_{R_{i}}$$

In the $*_{R}$-rule, the $\Delta$ and $\Delta^{'}$ are active, since they identified sub-bunches required for the inference. In $*_{R_{1}}$ and $*_{R_{2}}$, none of the context on any of sequent, premiss or conclusion, is active.

The new implication left rules can be justified similarly to the new multiplicative conjunction right rules. To simplify the discussion we introduce terminology: a resolution is a (reductive) inference of the following shape, where $\circ$ is the additive context-former or multiplicative context-former, and $\implies$ is either the additive or multiplicative implication symbol, respectively:

$$\Delta^{'} \implies \phi \quad \Gamma(\Delta^{''} \circ \phi \implies \psi) \implies \chi \quad *_{R_{i}}$$

In other words, resolution means implication left rules; they are distinguished morally by being read upward rather than downward, reductively rather than deductively. The rôle of $\Delta$ and $\Delta^{'}$ in a resolution is to manage the flow of data in the neighbourhood of the principal $\implies$-formula. The are four cases to consider:

- actual data is required to justify the antecedent but only a strict sub-bunch of the available data — viz. both $\Delta^{'}$ and $\Delta^{''}$ are non-empty bunches
- actual data is required to justify it and it is all available data — viz. $\Delta^{'}$ is a non-empty bunch and $\Delta^{''}$ is an empty bunch
- no data is used to justify it but there is available data — viz. $\Delta^{'}$ is an empty bunch and $\Delta^{''}$ is a non-empty bunch
- no data is used to justify it and there is not available data — viz. both $\Delta^{'}$ and $\Delta^{''}$ are empty.

In $LBI$ and $LBI^{'}$, the resolution in the cases containing an empty bunch (cases two, three, and four) would have to be preceded by an appropriate unit contraction, similar to the case for $*_{R}$ above, simply to make the resolution well-formed according to the specification of the sequent calculus. In $dLBI$, these unit contractions are built-in, so that one can intuitively perform a resolution sending no data to one or more of the branches as required.

**Example 3.16.** An immediate reduction of $\Delta, T^{*} \implies \phi \implies \psi$ simplifying the sequent is to remove the vacuous guard for $\phi$ in the context (i.e., the $T^{*}$ antecedent). Doing that in $LBI$ and $LBI^{'}$ requires a unit-contraction before the resolution is possible, but since the guard is vacuous (i.e., requires no data to be justified), the reduction can be immediately made in $dLBI$:

$$\varnothing_{x} \implies T^{*} \quad \Delta, T^{*} \implies \phi \implies \psi \quad *_{R_{2}}$$

As above, $\Delta^{'}$ and $\Delta^{''}$ in resolutions are active when they are specified sub-bunches in the context of the conclusion of the inference, and since both are specified only in $\implies_{L}$ and $\implies_{L}$, only in these rules are both active; in instances of $\implies_{1}$, $\implies_{2}$, $\implies_{1}$, and $\implies_{3}$, which correspond to cases two and three above, there is only one active sub-bunch in the premisses and the conclusion other than the implicational formula and its components; and, in $\implies_{3}$-rule there are no active sub-bunches. In short, the unit bunches witnessed in the specification of $\implies_{i}$ for $i \in \{1, 2, 3\}$ are not considered.
active in any of the new inferences, but the $\Delta$ in the specification of the rules in Figure 5, if there is one, is active, which is coherent with Definition 2.14 (Active).

The use of substitution (see Section 2) in the specification of the resolution rules means that everything inside the bunch goes to the right-hand premiss unless specifically stated otherwise; the following examples illustrate this reading with the $\neg\rightarrow$-resolution rules:

Example 3.17. The following shows instances of $\neg\rightarrow_L, \neg\rightarrow_L^1, \neg\rightarrow_L^2$, and $\neg\rightarrow_L^3$ on the same sequent, respectively, differentiated by the choice of how the data in the neighbourhood is used to justify the antecedent of the principal implicational formula:

$$\Gamma (\Delta \vdash \varnothing_{\neg\rightarrow}) \Rightarrow \chi'$$

$$\Gamma (\Delta \vdash \varnothing_{\neg\rightarrow}) \Rightarrow \chi'$$

Remark. The $\neg\rightarrow_2$ and $\neg\rightarrow_3$ rules are different only in that the latter attaches a unit to the conclusion of the principal implicational formula in the right premiss; hence, the former can be recovered from the latter by a (reductive) use of $\neg\rightarrow_{\neg\rightarrow_3}$, and thus it is not required for completeness. The purpose of having all the rules in the system is for symmetry and to simplify the cases analysis in the rewriting argument that delivers the completeness of dLBI. The analogous remark holds for $\neg\rightarrow_2$ and $\neg\rightarrow_3$.

The only rule that remains to be discuss is the conv-rule. The elimination of $c_{\varnothing_x}$ and $c_{\varnothing_x}$ from LBI’ is proved by an upward permutation rewriting argument, and their interaction with the other rules yields the additional rules in dLBI. There are two particular interactions with weakening that are of interest:

$$\Gamma (\Delta \vdash \varnothing_{\neg\rightarrow}) \Rightarrow \chi'$$

Read downward, both these cases concern the conversion of an additive unit into a multiplicative unit. In the left case this is without consequence, but in the right one the resulting unit is then removed. This is a problem as it means that during proof-search one has to consider sequents with multiplicative context-formers that have no ancestor in the original sequent, the original problem with $c_{\varnothing_x}$, which, among other things, causes the number of generations in the construction of the proof-search space to be unbounded. The good conversion (left) is generalized to become the conv rule, and the bad conversion (right), dubbed a radicalization, determining the rad-rule,

$$\Gamma (\Delta \vdash \varnothing_{\neg\rightarrow}) \Rightarrow \chi'$$

Fortunately, the rad rule may be eliminated through another permutation argument.
Remark. Since dLBI is a restriction of LBI′ (itself a restriction of LBI) in which certain sub-derivations involving the contraction rules are collapsed into single inferences, it is clear that proofs in the former can be algorithmically transformed into proofs in the latter. More explicitly, given a dLBI-proof, one recovers a LBI proof by reversing the collapsing of sub-derivation in special dLBI-rule; for example, encountering certain instances of conv, one reverse the transformation above. It follows that restricting attention to proofs in dLBI forms a problem reduction for witnessing an effective proof-search procedure for LBI.

3.3 Regimented Proof
In this section Gentzen’s control régime is generalized to BI. The régime determines a class of highly structured but not necessarily goal-directed proofs dubbed regimented. Heuristically, regimented proofs are structured by the following three phases, repeated in succession:

• ACTION: Logical deductions are made; that is, information is introduced, combined, or modified for the sequents so far constructed.

• NORMALIZATION: Any computational effects of the action phase, such as the introduction of extraneous data, are removed.

• LOADING: The extraneous data required for the next action are added.

3.3.1 Action. Informally, information in a context is a sub-bunch; it is extraneous when it can be removed through reduction (i.e., when it can be contracted). Consequently, in dLBI (and the other systems studied above), actions are not merely logical rules because some instances of w′ add information to the sequent. By the definition of reduction, a sub-bunch may be regarded as an extraneous datum when there is another sub-bunch in its additive set that is a permutation of it; when this is the case, the sub-bunch is duplicity.

Definition 3.18 (Duplicity). Let Δ be a sub-bunch of Γ. The duplicity set of Δ, denoted dΔ, is the collection of occurrences of permutation in its neighbourhood; that is,

\[ dΔ = \{ Δ′ ⊒ Γ | Δ ≡ Δ′ \text{ and } Δ \sim Δ′ \} \]

The duplicity of a sub-bunch Δ in Γ is \(|dΔ| − 1\).

The duplicity of an additive in Γ is the maximal duplicity of any of the members; that is, the duplicity of an additive set A is \(dA := \sup_{Δ ∈ A} |dΔ| − 1\).

Duplicity sets and additive sets are trivial if they have zero duplicity, and a bunch is duplicity if its duplicity set is non-trivial.

Returning to the discussion about weakening, suppose the bunch introduced by w′ is duplicity, then no actual information has been added since it can be removed and returned at will using the structural rules, but if it is not duplicity, then the overall information of the context has increased; the first may be regarded as a loading inference and the second as an action.

Definition 3.19 (dLBI-Action). An inference \(r(P_0, ..., P_n, C)\) is an dLBI-action if the rule \(r\) is a logical rule, an axiom, or \(r\) is one of w′ or conv with the constraint that the introduced bunch Σ is not duplicity in the context of C.

3.3.2 Loading. The purpose of this phase is to restrict actions so that the amount of duplicity in the sequence involved is minimal. This is done by permitting only duplicity where it is essential; that is, for active sub-bunches in the inference:

Definition 3.20 (Regimented Equivalent Inference). Let \(r(P_1, ..., P_n, C)\) be an action. Denote \(\hat{P}_i\) for a sequent resulting from \(P_i\) after removing (through reduction) any bunches not active in the inference, and denote \(\hat{C}\) for the result of applying \(r\) to the same active bunches of \(\hat{P}_0, ..., \hat{P}_n\). The
inference \(r(\bar{P}_1, \ldots, \bar{P}_n, \bar{C})\) is a regimented equivalent inference to the original. If the inference was an action, then the regimentation defines a regimented action.

**Example 3.21.** In the following instance of \(\rightarrow\) \(L\) occurrences of formulas are labelled by superscripts that are assigned using the smallest fresh natural number available when reading the sequent from the left:

\[
\begin{align*}
\phi^0 \vdash \phi^1 & \implies \phi^2 & \psi^0 \vdash \psi^1 & \implies \chi^0 \\
\phi^0 \vdash \phi^1 \vdash \psi^0 \vdash \phi \rightarrow \psi^0 & \implies \chi^0 & \phi^0 \vdash \psi^0 \vdash \phi \rightarrow \psi^0 & \implies \chi^0
\end{align*}
\]

The first inference is an unregimented action since in the left antecedent \(\phi^0\) and \(\phi^1\) are duplicate but not active. The second inference is a regimented action since, though its right antecedent also contains duplicate sub-bunches (viz. \(\psi^0\) and \(\psi^1\)), they are both active. \(\square\)

It is essential that no information is lost in the process of regimentation; that is, that the conclusion of the regimented equivalent inference contains the same information as the original one. Here is where confluence of reduction becomes important:

**Lemma 3.22.** For any instance of an action \(\hat{C}\), \(\hat{C}\) reduces to a normal form of \(C\).

**Proof.** Let \(\sigma_i\) be sequences witnessing \(P_i \succ^* \bar{P}_i\) and respectively. The applicable reductions in the concatenation \(\sigma_1 \ldots \sigma_n\) is a reduction sequence witnessing \(C \succ^* \hat{C}\). It follows, by confluence (i.e., by Lemma 3.10), that \(\hat{C} \succ^* \hat{C}\), as required. \(\square\)

It is now possible to define the *loading* phase, which is the introduction of precisely the data required to make the next action, which is then necessarily regimented.

**Definition 3.23 (Loading Strategy).** Let \(r(P_0, \ldots, P_n, C)\) be an action. Any negative strategy witnessing \(\bar{P}_i \succ^* \bar{P}_i\) is a loading strategy.

**3.3.3 Normalization.** The notion of *normalization* is already provided by the definition of reduction and strategy in Section 3.1.

**3.3.4 Regimented Proof.** All the definitions above apply to any of the sequent calculi of BI studied herein; that is, the notion of regimented proof below may be instantiated in any sequent calculus when the definition of action, which takes a proof system as a parameter, has been fixed.

**Definition 3.24 (Regimented Derivations).** A \(L\)-derivation is regimented if read root-to-leaf every \(L\)-action is regimented, is preceded by a normalizing strategy, and is succeeded by a loading strategy.

If \(D : \Sigma \vdash_L S\) is a regimented \(L\)-derivation, then it may be denoted \(D : \Sigma \vdash_L S\). The normalizing and loading phases may be empty if the conclusion of the action is normal or nothing needs to be loaded for the next action.

**Remark.** The restriction to regimented derivation is Gentzen’s control régime, it is established by an inductive transformation of inferences below.

**3.4 Completeness of dLBI**

Attention to the rules of dLBI has so far been with respect to the additional rules and the missing ones, but the structural rules have also been restricted. Hence, for the sake of simplicity, the completeness argument proceeds through an intermediary system of *small-step* inference called sLBI, which has the rules of LBI’ restricted as in dLBI. It is an aid to breaking up the long and tedious permutation arguments that form the proof of completeness of dLBI.
The soundness and completeness argument of normal forms with respect to regimented dLBI-proofs is in four stages only the last two of which remain to be done:

\[ \vdash_{LBI} S \iff \vdash_{LBI'} S \]  
\[ \iff \vdash_{LBI'} \bar{S} \]  
\[ \iff \vdash_{\bar{S}} S \iff \vdash_{LBI+rad} \bar{S} \iff \vdash_{dLBI+rad} \bar{S} \]  
\[ \iff \vdash_{dLBI} \bar{S} \]  

(Section 3.4.1)  

3.4.1 Completeness of dLBI+. To show completeness (i.e., \( \vdash_{LBI} \bar{S} \iff \vdash_{dLBI+rad} \bar{S} \)) one first applies Gentzen’s control regime, and second eliminates any instance of \( c_{\varphi_{\alpha}} \) and \( c_{\varphi_{\alpha}} \) in the resulting proof by permuting upward, adding \( \rightarrow_{L_1}, \rightarrow_{L_2}, \rightarrow_{L_3}, \rightarrow_{R_1}, \rightarrow_{R_2}, \rightarrow_{L_1}, \rightarrow_{L_2}, \rightarrow_{L_3}, \rightarrow_{R_1}, \rightarrow_{R_2}, \rightarrow \) conv, and rad to the system. The result is a regimented (dLBI + rad)-proof. However, the argument is most easily understood by first handling the restriction of the structural rules, which is captured by system sLBI:

**Definition 3.25 (System sLBI).** System sLBI is the variant of LBI’ (i.e., composed of the rules of Figure 2 and Figure 3) in which \( c, w, \) and \( \bot \) replaced by the following, where \( \Sigma \) is a normal bunch:

\[ \frac{\Gamma(\Sigma \uplus \Sigma)}{\Gamma} \Rightarrow \chi \quad c' \quad \frac{\Gamma(\Delta)}{\Gamma(\Delta \uplus \Sigma)} \Rightarrow \chi \quad w' \quad \frac{\Sigma(\bot)}{\chi \bot'} \]

Since sLBI is a restriction of LBI’, soundness is immediate. For completeness, it is sufficient to show admissibility for each rule \( r \) in LBI’; that is

\[ \forall r \in LBI' \exists \mathcal{D} ( r(P_0, ..., P_n, C) \& D : P_0, ..., P_n \vdash_{sLBI} C ) \]

**Lemma 3.26.** \( \vdash_{LBI'} S \iff \vdash_{sLBI} S \).

**Proof.** The \( \vdash_{LBI'} S \iff \vdash_{sLBI} S \) direction is immediate as any instance of a sLBI rule is an instance of a LBI’ rule. For the \( \vdash_{sLBI} S \Rightarrow \vdash_{sLBI} S \) direction, it suffices to show the admissibility of all the rules of LBI’ in sLBI. All the rules except \( w, c, \) and \( \bot \) are already available, so it only remains to shown that any instance \( r(P, C) \) or \( r(C) \) of these rules can be simulated. In each case let \( A \) be the context of the premiss and let \( B \) the context of the conclusion. We show admissibility of each rule independently.

**Absurdity.** The \( \bot \)-rule may be simulated as follows:

\[
\frac{\Gamma(\bot)}{\phi} \quad \vdots \quad D : (\Gamma(\bot) \Rightarrow \phi) \quad \land (\Gamma(\bot) \Rightarrow \phi) \quad \vdots \quad \Gamma(\bot) \Rightarrow \phi.
\]

**Weakening.** We have \( A = \Gamma(\Delta) \) and \( B = \Gamma(\Delta' \uplus \Delta') \), where \( \Delta' \) is some bunch. By Lemma 3.4, there is a normal form \( \Delta' \) of \( \Delta' \), and there is a sequence \( \sigma \) witnessing the reduction \( \Delta' \Rightarrow r \Delta' \). Consider the following derivation:
Provability in BI’s Sequent Calculus is Decidable

\[
\begin{align*}
\Gamma(\Delta) & \Rightarrow \phi \\
\Gamma(\Delta ; \Delta') & \Rightarrow \phi
\end{align*}
\]

\[\vdash_{\sigma} : (\Gamma(\Delta ; \Delta') \Rightarrow \phi) \ll (\Gamma(\Delta ; \Delta') \Rightarrow \phi)\]

This derivation is the desired simulation, it is sLBI-sound since negative strategies are sound and the first step is an instance of w'. It begins and ends with the appropriate sequents.

CONTRACTION. We have A = \Gamma(\Delta ; \Delta) and B = \Gamma(\Delta). By Lemma 3.4 there is a normal form \check{\Delta} of \Delta, and a sequence \sigma witnessing the reduction \Delta \supseteq^* \check{\Delta}. Consider the following proof:

\[
\begin{align*}
\Gamma(\Delta ; \Delta) & \Rightarrow \phi \\
\vdash_{\sigma} : (\Gamma(\Delta ; \Delta) \Rightarrow \phi) \supseteq^* (\Gamma(\Delta ; \check{\Delta}) \Rightarrow \phi) \\
\Gamma(\Delta ; \check{\Delta}) & \Rightarrow \phi
\end{align*}
\]

Since the positive and negative strategies are sLBI-sound and the only other inference is an instance of the c’ rule, the whole proof is sLBI-sound. Since it begins and ends with the appropriate sequents, it gives the desired simulation.

We now proceed to showing the completeness of regimented sLBI-proofs, as they can be transformed into regimented dLBI proofs.

Definition 3.27 (sLBI-action). An inference \(r(P_0, ..., P_n, C)\) is an sLBI-action if the rule \(r\) is a logical sLBI-rule, an axiom; or, \(r\) is w', with the constraint that the introduced bunch \(\Sigma\) is not duplicit in the context of \(C\).

Lemma 3.28. \(\vdash_{sLBI} S \iff \vdash_{sLBI} S\)

Proof. The \(\vdash_{sLBI} S \iff \vdash_{sLBI} S\) direction is immediate since regimented sLBI-proofs are sLBI-proofs. It remains only to prove the \(\vdash_{sLBI} S \implies \vdash_{sLBI} S\) direction.

Given an sLBI-proof of a sequent, denote \(T\) for the tree whose nodes are tuples of actions in the given proof, and call it the action tree. The nodes in the action tree may be expanded into regimented actions preceded by loading strategies and succeeded by normalizing strategies as follows:
Hence, \( \mathcal{T} \) can be recursively transformed into a regimented \(dLBI\)-proof of \( \hat{S} \). The procedure is defined on the maximal number of nodes in a branch in the action tree, and it terminates since each recursion is on a strictly smaller action tree. The recursion is defined as follows:

**Base Case.** If \( \mathcal{T} \) is a single node, then the original proof contained only one action. By definition of a proof, this action was an axiom. By Lemma 3.10 and Lemma 3.12, the sequent may be extended with a normalizing strategy with endsequent \( \hat{S} \). This yields the desired regimented proof.

**Induction Step.** Let \((P_1, ..., P_n, C)\) be the last node of the action tree \( \mathcal{T} \), and let \( r \) be a rule witnessing the inference. Recur on the immediate sub-trees with end-sequents \( P_1, ..., P_n \) to obtain regimented proofs \( \mathcal{D}_l \) for \( \hat{P}_i \), respectively. By Lemma 3.10 and Lemma 3.12, the proofs \( \mathcal{D}_l \) may be extended with loading strategies to yield proofs \( \hat{\mathcal{D}}_l \) for \( \hat{P}_i \), respectively. The proves can be combined using a regimented equivalent inference to the original action, concluding a proof of \( \hat{C} \). It follows from Lemma 3.22 and Lemma 3.12 that this proof can be extended with a normalizing strategy to obtain a proof of \( \hat{C} \). By Lemma 3.10, we can take, without loss of generality, \( \hat{C} = \hat{S} \) as, by definition of \( \mathcal{T} \), there are no actions between \( C \) and \( \hat{S} \) in the input proof.

We now return to \(dLBI\) and relate the two systems through the regimented proofs. Soundness is immediate:

**Lemma 3.29.** \( \vdash_{dLBI} \hat{S} \iff \vdash_{dLBI} \hat{S} \).

**Proof.** All the inferences of \(dLBI\) are derivable in \(sLBI\); the rules not already present (i.e., \(\rightarrow^{L1}, \rightarrow^{L2}, \leftrightarrow^{L3}, \supset^{R1}, \supset^{R2}, \rightarrow^{L1}, \rightarrow^{L2}, \land^{R1}, \land^{R2}, \text{conv.}\)) are shown below:

\[
\begin{align*}
\Delta \supset \phi & \quad \Gamma(\emptyset_x, \psi) \supset \chi \quad \text{L} \quad \frac{\Gamma(\Delta, \emptyset_x, \phi \rightarrow \psi) \quad \chi \quad \text{c_{\emptyset_x}}}{\Gamma(\Delta, \phi \rightarrow \psi) \quad \chi} \\
\emptyset_x \supset \phi & \quad \Delta \supset \psi \quad \text{R} \quad \frac{\emptyset_x \supset \phi \quad \Delta \supset \psi \quad \text{c_{\emptyset_x}}}{\emptyset_x \supset \phi \quad \Delta \supset \psi} \\
\Delta \supset \phi & \quad \Gamma(\emptyset_x, \psi) \supset \chi \quad \text{L} \quad \frac{\Gamma(\Delta, \emptyset_x, \phi \rightarrow \psi) \quad \chi \quad \text{c_{\emptyset_x}}}{\Gamma(\Delta, \phi \rightarrow \psi) \quad \chi} \\
\emptyset_+ \supset \phi & \quad \Gamma(\Delta, \psi) \supset \chi \quad \text{R} \quad \frac{\emptyset_+ \supset \phi \quad \Gamma(\Delta, \psi) \supset \chi \quad \text{c_{\emptyset_+}}}{\emptyset_+ \supset \phi \quad \Gamma(\Delta, \psi) \supset \chi}
\end{align*}
\]
We conclude that unit contractions follow either actions or other unit contractions.

\[
\begin{array}{l}
\varnothing \implies \phi \quad \Gamma(\varnothing \implies \psi) \implies \chi \quad \rightarrow_{L} \\
\Gamma(\varnothing \implies \psi) \implies \chi \quad c_{\varnothing_{x}} \\
\Gamma(\varnothing \implies \psi) \implies \chi \quad c_{\varnothing_{x}} \\
\end{array}
\]

\[
\Delta \implies \phi \quad \psi \quad \wedge_{R} \\
\varnothing_{+} \implies \Delta \implies \phi \wedge \psi \quad c_{\varnothing_{+}} \\
\Delta \implies \phi \wedge \psi \quad c_{\varnothing_{+}} \\
\Gamma \implies \phi \quad \varnothing_{+} \implies \psi \quad \wedge_{R} \\
\Gamma \implies \phi \quad c_{\varnothing_{+}} \\
\Gamma \implies \phi \quad \psi \quad c_{\varnothing_{+}} \quad \square
\]

For the completeness proof, we must eliminate all instances of the \(c_{\varnothing_{+}}\) and \(c_{\varnothing_{x}}\) rules in an arbitrary regimented sLBI-proof, possibly by introducing one of the new rules; to do this, it is first helpful to know where the instances are:

**Definition 3.30 (d-property).** A sLBI \(\cup dLBI \cup \{\{\text{rad}\}\}\)-proof has the d-property if all instances of \(c_{\varnothing_{+}}\) and \(c_{\varnothing_{x}}\) either follow an action or follow an instance of itself following an action.

We may write \(\mathcal{D} : \varnothing \oplus_{L} S\) to denote that \(\mathcal{D}\) is a regimented \(L\)-proof which has the d-property.

**Lemma 3.31.** \(\vdash_{sLBI} S \implies \vdash_{sLBI+\{\text{rad}\}} S\)

**Proof.** Call a unit a zero when it the child of a matching context-former, then \(c_{\varnothing_{x}}\) and \(c_{\varnothing_{+}}\) are applicable if and only if a zero has been introduced. Without loss of generality, by Lemma 3.10, assume they are applied eagerly; that is, whenever a zero appears after an action or in a normalizing phase of the regimented proof, the appropriate contraction rule is immediately applied.

Consider the first instance, reading leaf-to-root, of \(c_{\varnothing_{x}}\) or \(c_{\varnothing_{+}}\) in the regimented sLBI-proof; it can follow \(c\) only in the following case:

\[
\Gamma(\Delta_{+} (\varnothing_{x} \implies \varnothing_{x})) \implies \chi \quad c_{\varnothing_{+}} \\
\Gamma(\Delta_{+} \varnothing_{x}) \implies \chi \quad c_{\varnothing_{x}} \\
\Gamma(\Delta) \implies \chi \quad c_{\varnothing_{x}}
\]

However, then it must be that the first sequent in the strategy contained an additive set of multiplicative units, and therefore that the action preceding the normalizing phase additively combines data not combined in the premisses. Hence, the action must be one of \(\wedge_{R}, \rightarrow_{L}\), and \(w'\), but for none of these rules can the configuration occur:

- For \(\wedge_{R}\), the additive context-former would be principal in the bunch, but there is a multiplicative context-former above it
- For \(\rightarrow_{L}\), the additive set would contain an implication formula, but it does not
- For \(w'\), the inference would not be an action since it introduces a duplicit bunch.

We conclude that unit contractions follow either actions or other unit contractions. Consider the case of \(c_{\varnothing_{+}}\) following \(c_{\varnothing_{+}}\); that is,

\[
\Gamma(\Delta_{+} \varnothing_{+}) \implies \chi \quad c_{\varnothing_{+}} \\
\Gamma(\Delta_{+} \varnothing_{+}) \implies \chi \quad c_{\varnothing_{+}} \\
\Gamma(\Delta) \implies \chi \quad c_{\varnothing_{+}}
\]

This is also impossible as no action can conclude the first sequent in the derivation.

Consider the case of \(c_{\varnothing_{x}}\) following \(c_{\varnothing_{x}}\), instead. This is possible, but only when preceded by a weakening, in which case the inferences may be collectively replaced by rad:

\[
\Gamma(\Delta_{+} \varnothing_{+}) \implies \chi \quad w' \\
\Gamma(\Delta_{+} (\varnothing_{x} \implies \varnothing_{x})) \implies \chi \quad c_{\varnothing_{x}} \\
\Gamma(\Delta_{+} \varnothing_{x}) \implies \chi \quad c_{\varnothing_{x}} \\
\Gamma(\Delta) \implies \chi \quad \text{rad}
\]
To handle the remaining cases, a sLBI-proof $D$ may be recursively rewritten with the following scheme to yield a proof satisfying the $d$-property; the procedure halts because each recursion occurs on a strictly smaller sub-tree:

**Base Case.** If $D$ has only one action, then the action must be an axiom which all introduce normal bunches, hence it already satisfies the criterion.

**Induction Step.** Consider the last action $r(P_0, ..., P_n, C)$, and recur on the sub-proofs of $P_0, ..., P_n$ to achieve proofs satisfying the criterion. As a result, all instances of $c_{\phi}$ and $c_{\psi}$ in the resulting proof are below the action. Consider then the first instance in the proof contradicting the $d$-property, then from the above we see that the sub-derivations may be replaced by a single instance of rad. 

**Lemma 3.32.** $\vdash_{d \text{-lB} \text{-rad}} \tilde{S} \Longrightarrow \vdash_{d \text{-lB} \text{-rad}} \tilde{S}$.

**Proof.** We can transform any regimented sLBI-proof with the $d$-property into a regimented dLBI-proof by eliminating all the instances of $c_{\phi}$ or $c_{\psi}$. The rules are only applicable after certain instances of $*_{R}$, $*_{L}$, $\wedge_{R}$, $\rightarrow_{L}$, and $w'$, and the respective transformations are as follows:

**Cases of $*_{L}$:**

$$
\begin{align*}
\Delta \Rightarrow \phi &\quad \Gamma(\varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad \Uparrow &\quad \Delta \Rightarrow \phi &\quad \Gamma(\varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad \rightarrow_{L1} \\
\Gamma(\Delta \times \varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad c_{\phi} &\quad \rightarrow_{L1}
\end{align*}
$$

**Cases of $*_{R}$:**

$$
\begin{align*}
\varnothing &\quad \Rightarrow \phi &\quad \Delta \Rightarrow \psi &\quad \Uparrow &\quad \varnothing &\quad \Rightarrow \phi &\quad \Delta \Rightarrow \psi &\quad \rightarrow_{R1} \\
\varnothing \times \Delta &\quad \Rightarrow \phi \times \psi &\quad c_{\phi} &\quad \rightarrow_{R1}
\end{align*}
$$

**Cases of $\rightarrow_{L}$:**

$$
\begin{align*}
\Delta \Rightarrow \phi &\quad \Gamma(\varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad \Uparrow &\quad \Delta \Rightarrow \phi &\quad \Gamma(\varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad \rightarrow_{L1} \\
\Gamma(\Delta \times \varnothing \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad c_{\phi} &\quad \rightarrow_{L1}
\end{align*}
$$

$$
\begin{align*}
\varnothing &\quad \Rightarrow \phi &\quad \Delta \Rightarrow \psi &\quad \Uparrow &\quad \varnothing &\quad \Rightarrow \phi &\quad \Delta \Rightarrow \psi &\quad \rightarrow_{L2} \\
\Gamma(\varnothing \times \Delta \times \phi \Rightarrow \psi) \Rightarrow \chi &\quad c_{\phi} &\quad \rightarrow_{L2}
\end{align*}
$$
there are two possibilities to consider: either the radical $w$ as active in the
by the following transformation:

$\varnothing_+ \iff \phi \Gamma(\varnothing_+; \psi) \implies \chi_{c_\varnothing} \quad \Rightarrow \quad \varnothing_+ \iff \phi \Gamma(\varnothing_+; \psi) \implies \chi_{c_\varnothing}$

$\Gamma(\varnothing_+; \phi \iff \psi) \implies \chi_{c_\varnothing}$

$\Gamma(\phi \iff \psi) \implies \chi_{c_\varnothing}$

**Cases of $\Lambda_R$:**

$\varnothing_+ \iff \phi \Delta \iff \psi_{\Lambda_R}$

$\varnothing_+ \iff \phi \Delta \iff \psi_{\Lambda_R}$

$\varnothing_+ \iff \phi \Delta \iff \psi_{\Lambda_R}$

After transformation, the proof remains regimented since the phase structure is preserved, but there are no instances of $c_\varnothing$ or $c_\varnothing$ remaining, so that the proof is a regimented dLBI-proof.  

**3.4.2 Elimination of $rad$.** What remains is rad-elimination (i.e., $\vdash_{dLBI+rad} \hat{S} \iff \vdash_{dLBI} \hat{S}$), which is also through permutation.

**Lemma 3.33.** $\vdash_{dLBI+rad} \hat{S} \iff \vdash_{dLBI} \hat{S}$.  

**Proof.** The $\vdash_{dLBI} \hat{S} \iff \vdash_{dLBI+rad} \hat{S}$ is immediate since any dLBI-proof is a dLBI + rad-proof. It remains only to consider the $\vdash_{dLBI+rad} \hat{S} \iff \vdash_{dLBI} \hat{S}$ direction.

We call any additive unit that is the child of a multiplicative context-former a radical. Consider an arbitrary instance of rad in a regimented dLBI + rad proof. If the unit on which the rule is applied is not active in the inference preceding it, then the two rules may be trivially permuted; for example,

$\varnothing_+ \iff \Delta \iff \psi_{\Lambda_R}$

$\Delta \iff \phi \varnothing_+ \iff \psi_{\Lambda_R}$

$\Delta \iff \phi \varnothing_+ \iff \psi_{\Lambda_R}$

Suppose this is done, then the loading strategy preceding the rad is empty as it is not possible for the unit be active in the loading inference. Therefore, the rad follows either from another regimented action or from a normalizing strategy.

If rad follows from a normalizing strategy, then the preceding rule was an instance of $c'$, and there are two possibilities to consider: either the radical was active in the $c'$, or it was not. The first case is impossible since it would require the conclusion of the action preceding the normalizing strategy to contain an additive set containing only additive units. The second case can be handled by the following transformation:
It remains to consider the case where the instance of \( \text{rad} \) follows from a regimented action. The only axiom that can introduce a radical is \( \bot' \); however, the instance may simply be replaced by another, which does not have the offending sub-bunch. Similarly, we may disregard \( T_{L_{1}}^{*}, \sim_{R_{1}}, *_{L}, w' \), and \( \text{conv} \), as the conclusion cannot contain a radical not present in the premiss. What remains are \( \sim_{L_{1}}, *_{R_{1}}, w' \) and \( \text{conv} \), which are transformed as follows:

\[
\begin{align*}
\varnothing_{+} &\quad \Rightarrow \quad \phi \quad \Gamma(\Delta, \psi) \quad \Rightarrow \quad \chi \quad \rightarrow_{L} \\
\Gamma(\Delta, \phi \approx \psi) &\quad \Rightarrow \quad \chi \quad \rightarrow_{L} \\
\varnothing_{x} &\quad \Rightarrow \quad \phi \quad \Gamma(\varnothing_{x}, \psi) \quad \Rightarrow \quad \chi \quad \rightarrow_{L_{1}} \\
\Gamma(\varnothing_{x}, \phi \approx \psi) &\quad \Rightarrow \quad \chi \quad \rightarrow_{L_{1}} \\
\varnothing_{+} &\quad \Rightarrow \quad \phi \quad \Delta \quad \Rightarrow \quad \psi \quad \rightarrow_{R} \\
\varnothing_{+}, \Delta &\quad \Rightarrow \quad \phi \approx \psi \quad \rightarrow_{R} \\
\Gamma &\quad \Rightarrow \quad \phi \quad \varnothing_{+} \quad \Rightarrow \quad \psi \quad \rightarrow_{R} \\
\Gamma, \varnothing_{+} &\quad \Rightarrow \quad \phi \approx \psi \quad \rightarrow_{R} \\
\Gamma(\Delta) &\quad \Rightarrow \quad \phi \quad \Gamma(\Delta; \Sigma'(\varnothing_{+}, \varnothing_{x})) \quad \Rightarrow \quad \phi' \quad \rightarrow_{R} \\
\Gamma(\Delta; \Sigma'(\varnothing_{+}, \varnothing_{x})) &\quad \Rightarrow \quad \phi \quad \rightarrow_{R} \\
\Gamma(\varnothing_{+}) &\quad \Rightarrow \quad \phi \quad \Gamma(\varnothing_{+}) \quad \Rightarrow \quad \phi' \quad \rightarrow_{R} \\
\Gamma(\varnothing_{x}) &\quad \Rightarrow \quad \phi \quad \Gamma(\varnothing_{x}) \quad \Rightarrow \quad \phi \quad \rightarrow_{R} \\
\end{align*}
\]

**Theorem 3.34 (Provability Problem Reduction).** A sequent \( S \) is provable iff any normal form admits a regimented dLBI-proof; that is,

\[ \vdash_{LBI} S \quad \iff \quad \not\vdash_{dLBI} \overline{S} \]

\section{4 THE PROOF-SEARCH SPACE}

This section shows that regimented provability in dLBI is decidable, the decidability problem for LBI follows as a corollary. The argument is that restricting attention to those sequents that may
occur in a regimented dLBI-proof is sufficient control régime to have a primitive recursive proof-search space in the style discussed in Section 1. The structure is as follows: Section 4.1 introduces three measures, $\mu$, $\omega$, and $\delta$, corresponding to the dimensions of the naive construction of the proof-search space; Section 4.2 establishes effective bounds for $\omega$ and $\delta$; Section 4.3 establishes effective bounds for $\mu$; and, Section 4.4 gives a putative conclusion; and, Section 4.4 gives a decision procedure for regimented provability in dLBI.

4.1 Measures

There are two *widths* of bunches that must be studied, additive and multiplicative, which correspond to the size of multisets that are considered in the naive construction; there is also *depth*, which corresponds to the number of generations considered.

Though additive and multiplicative width are intuitively symmetric, their control structures are not. In particular, the work on reduction attends largely to bounding additive width since it is additive combination that permits the use the contraction rule $c'$. The work on eliminating $c\bot yse\emptyset. al\emptyset. el$ is precisely for the control over the multiplicative structures over a bunch. Both measures consider the amount of data — sub-bunches — that are combined using a particular context-former, but compute differently based on this information. It happens that depth and multiplicative width are much more closely related since they are both affected by multiplicative context-formers — though, if one wishes, one may define depth-based on additive context-formers instead, but this would mean accounting for how the measure is affected during the normalization and loading phases of proof, thereby complicating matters rather than simplifying them.

4.1.1 Additive Width. This is the number of occurrences of additive data in the same additive set. The number of occurrences of a given sub-bunch in an environment is captured by *duplicity*,

$$d\Delta = \{\Delta' \sqsubseteq \Delta \mid \Delta \equiv \Delta' \text{ and } \Delta \sim \Delta'\}$$

Gentzen’s bound for intuitionistic logic, whose contexts are identified with additive sets $A$, can be stated formally as the property that $\mu(A) \leq 2$ for every sequent in a certain class of proofs known to be complete. Generalizing this to BI, the requirement becomes a simultaneous bound over all additive sets in the context:

**Definition 4.1 (Multiplicity).** Let $\Gamma \in \mathbb{B}$, and let $\mathcal{A}$ be the set of all additive sets in the bunch. The multiplicity of $\Gamma$, denoted $\mu(\Gamma)$, is defined as follows:

$$\mu(\Gamma) := \sum_{A \in \mathcal{A}} \partial A = \sum_{A \in \mathcal{A}} \left(\sup_{\Delta \in A} |d\Delta| - 1\right)$$

The subtraction is essential as it allows the sum to characterize *extraneous* data, in the sense of data occurrences that are repeated without adding information. In the case of a single additive set (e.g., when working in the additive fragment of BI), the measure recovers Gentzen’s bound of $\mu(\Gamma) \leq 2$.

**Example 4.2.** Let $\Gamma$ be the bunch in Figure 1, then $\psi$ is duplicit as $|d\psi| = 2$. This is the only non-trivial duplicity set, and $\mu(\Gamma) = 1$. \(\square\)

The *bounded growth property* (BGP) can now be formally stated: there is a $k \in \mathbb{N}$ such that, for every rule $r$, and any sequents $P_0, ..., P_1, C$, in the sequent calculus,

$$r(P_0, ...P_n, C) \implies \mu(C) \leq \max\{\mu(P_0), ..., \mu(P_n)\} + k$$
As an example of where the LBI fails to have the BGP, consider \( w(\emptyset_+ \Longrightarrow A, \emptyset_+ \uplus (\emptyset_+)^n \Longrightarrow A) \), where
\[
\begin{align*}
(\emptyset_+)^1 &= \emptyset_+ \\
(\emptyset_+)^n &= \emptyset_+ \uplus (\emptyset_+)^{n-1}
\end{align*}
\]
This family witnesses that LBI does not have the BGP since in a single inference the multiplicity can have increased arbitrarily. This justifies the restriction of the structural rules when forming dLBI.

4.1.2 Multiplicative Width. This is the number of multiplicative context-formers in the same environment; that is, it is an upper bound of the size of the multi-sets used in the naive construction of Section 1 for the construction of the bunch in question to be permitted. The most accurate definition would be a supremum over the cardinality of multiplicative set, analogous to additive sets defined above, but since multiplicative data is characterized by not proliferating during proof-search a simpler definition is available:

**Definition 4.3 (Weight).** The weight of a bunch is defined by the following:

\[
\omega(\Gamma) = \begin{cases} 
0 & \text{if } \Gamma \in A \cup \{\perp, \top, \top^*, \emptyset_+, \emptyset_\times\} \\
\max\{\omega(\psi_1), \omega(\psi_2)\} & \text{if } \Gamma = \psi_1 \circ \psi_2 \text{ for } \circ \in \{\land, \rightarrow, \lor\} \\
\omega(\psi_1) + \omega(\psi_2) + 1 & \text{if } \Gamma = \psi_1 \circ \psi_2 \text{ for } \circ \in \{\ast, \leftrightarrow\} \\
\max\{\omega(\Delta), \omega(\Delta')\} & \text{if } \Gamma = (\Delta \uplus \Delta') \\
\omega(\Delta) + \omega(\Delta') + 1 & \text{if } \Gamma = (\Delta_\uplus \Delta')
\end{cases}
\]

**Example 4.4.** Let \( \Gamma \) be the bunch in Figure 1, then one computed \( \omega(\Gamma) := 1 \) by reading the left branch of the parse-tree, supposing that none of the formulas contain any multiplicative connectives.

4.1.3 Depth. The generations required to produce a particular bunch in the naive construction of the proof-search space in Section 1 manifest as the number layers of additive and multiplicative data it contains. This is its depth. To compute it one may quotient the multisets first, in keeping with the notion of additive set above:

**Definition 4.5 (Topset).** Let \( \Gamma \) be a complex bunch. Denote by \( \Delta \in \Gamma \) that \( \Delta \triangleleft \Gamma \), with the additional constraint that \( \Delta \) is either basic or has a different context-former from \( \Gamma \), and that there are no context-former alternations between the principal context-former of \( \Delta \) and the root of \( \Gamma \). The set of such \( \Delta \) determine the topset of \( \Gamma \).

**Example 4.6.** Let \( \Gamma \) be the bunch in Figure 1, then \((\phi \circ (\chi \circ \emptyset_+)), \psi, \psi, \emptyset_\times \in \Gamma \), since in each case there are no context-former alternations between their principal context-formers and the root of the bunch. However, we do not have \( \psi \circ \emptyset_\times \in \Gamma \), since, although there are no alternations, its principal context-former is the same as the root of the bunch.

**Definition 4.7 (Depth).** The depth of a bunch is defined as follows:

\[
\delta(\Gamma) = \begin{cases} 
\omega(\Gamma) & \text{if } \Gamma \in A \cup \{\emptyset_+, \emptyset_\times\} \\
\max\{\delta(\Pi) \mid \Pi \in \Gamma\} & \text{if } \Gamma \in B^+ \\
\max\{\delta(\Sigma) \mid \Sigma \in \Gamma\} + 1 & \text{if } \Gamma \in B^\times
\end{cases}
\]

**Example 4.8.** Let \( \Gamma \) be the bunch in Figure 1, and suppose the formulas contain no multiplicative connectives, then since there is only one context-former alternation downward, we see \( \delta(\Gamma) = 1 \). □
4.1.4 **Primitive Recursion.** The measures of width and depth extend to sequents by distributing over the projections, and to proofs $\mathcal{D}$ by taking the least uniform bound; that is, for $f \in \{\mu, \omega, \delta\}$,

$$f(\Gamma \implies \phi) := f(\Gamma) + f(\phi) \quad \text{and} \quad f(\mathcal{D}) := \sup_{S \in \mathcal{D}} f(S)$$

**Example 4.9.** The multiplicity of a sequent $\Gamma \implies A$ in which $A \in \mathcal{A}$ has the same multiplicity as its context $\Gamma$.

Bounds in the measures of multiplicity, weight, and depth suffice to limit the proof-search space to a primitive recursive set. Let $\Phi$ be a set of formula and let $\int(\Phi)$ denote the set of sequents that are constructable from those formulas — meaning the context is a combinations of formulas from the set using context-formers and the extract is one of the formulas from the set.

Let $A \sqsubseteq B$ denote that $A$ is a multiset of elements of $B$. We may regard a multiset as a set for which each occurrence of a member is counted as a separate element, in which case we use the notation $|A|$ unambiguously to denote both the cardinality of a set and the number of elements in a multiset.

**Definition 4.10 (Additive and Multiplicative Combinations).** Let $B$ be a set of bunches. The sets additive and multiplicative combinations of elements are given by $\sum B$ and $\prod B$ respectively.

Additive and multiplicative combinations allow us to define the construction of the proof-search space. First we have have following operators,

$$\oplus^n B := \{\Delta \mid \text{there is } A \sqsubseteq B \text{ such that } |A| \leq n \text{ and } \Gamma \in \sum A\}$$

$$\otimes^n B := \{\Delta \mid \text{there is } M \sqsubseteq B \text{ such that } |M| \leq n \text{ and } \Gamma \in \prod M\}$$

The iterative application of taking additive and multiplicative combinations is defined as follows:

$$(\otimes^m \oplus^n)^d B := \begin{cases} \sum \otimes^m B & \text{if } d = 0 \\
(\otimes^m \oplus^n)^{d-1} \otimes^m \oplus^n B & \text{otherwise} \end{cases}$$

Let $B$ be a set of bunches, denote $\mathcal{G}(B)$ for the set of all bunches that can be constructed out of the elements of $B$.

**Lemma 4.11.** If $\Delta \in \mathcal{G}(B)$ satisfies $\mu(\Delta) \leq a$, $\omega(\Delta) \leq m$, and $\delta(\Delta) \leq d$, then

$$\Delta \in \mathcal{V}^d := \bigcup_{i=0}^d ((\otimes^m \oplus^n)^i B) \cup \bigcup_{i=0}^d ((\otimes^n \oplus^m)^i B)$$

**Proof.** We proceed by induction on the $d$ parameter. In the base case $d = 0$, but then $m = 0$, and we may conclude that $\Delta \in \oplus^0 B$, but $\mathcal{B} \subseteq \mathcal{V}$, so we are done. For the inductive step, observe $\delta(\Delta) \leq d + 1$ if and only if $\delta(\Delta) < d$ or $\delta(\Delta) = d + 1$. In the first case, we immediately have $\Delta \in \mathcal{V}^{d+1}$ by the induction hypothesis (since $\mathcal{V}^n \subseteq \mathcal{V}^{n+1}$). In the second case, we note that from the definition of depth, either $\Delta$ is a multiplicative composition of (additive) bunches of depth at most $d$ or it is an additive composition of multiplicative bunches which are combinations of (additive) bunches of depth at most $d$; that is, either $\Delta \in \sum \prod M$ or $\Delta \in \prod \sum M$ where $M \subseteq \mathcal{V}^d$. Hence $\Delta \in \oplus^a \bigcup_{i=0}^d ((\otimes^m \oplus^n)^i B)$ or $\Delta \in \otimes^m \bigcup_{i=0}^d ((\otimes^n \oplus^m)^i B)$, whence $\Delta \in \mathcal{V}^{d+1}$.

**Lemma 4.12.** Let $a, m, d \in \mathbb{N}$ be arbitrary and $\Phi$ be a finite set of formula, then the following set is finite and primitive recursive:

$$\mathcal{E}^{\Phi}_{a, m, d} := \{S \in \int(\Phi) \mid \mu(S) \leq a \text{ and } \omega(S) \leq m \text{ and } \delta(S) \leq d\}$$
Proof. The contexts of the elements in $\Sigma^\Phi_{a,m,d}$ are members of $\mathcal{G}(\Phi)$ satisfying $\mu(\Lambda) \leq a, \omega(\Lambda) \leq m$, and $\delta(\Lambda) \leq d$, thus by Lemma 4.11 they are member of $\mathcal{V}^d$. It follows that $\Sigma^\Phi_{a,m,d} \subseteq \mathcal{V}^d \times \Phi$, and since the right hand side is primitive recursive and finite and the separating condition is decidable, the left hand side is primitive recursive and finite.

4.2 Bounding Weight and Depth.

It remains to show that regimented dLBI-proofs have computable bounds for multiplicity, weight, and depth. The task list can be contracted by realising that weight dominates depth since they both count multiplicative context-formers:

Lemma 4.13. Let $S$ be a sequent, then $\delta(S) \leq \omega(S)$

Proof. The proof is by induction on the complexity of the context $\Gamma$ of $S$:

Base Case. If $\Gamma$ is basic, then $\omega(S) = \delta(S)$ by definition.

Induction Step. Consider the parse tree of $\Gamma$, the computation of depth only increments by one for each connected sub-tree of multiplicative connectives that it encounters, whereas the computation of weight increments by the number of multiplicative context-formers in the sub-tree. By the induction hypothesis the weight of the sub-tree chosen during maximization in the computation of weight is greater than the depth of the sub-tree chosen during maximization in the computation of depth.

The remainder of this section concerns the bounding of weight. Even in the case of regimented proofs, due to the interaction between the resolution rules (i.e., the implication left rules) and contraction, weight can increase during proof-search:

Example 4.14. Consider the following proof in which neither $\phi$ nor $\psi$ contains a multiplicative connective:

\[
\frac{\varnothing \quad \Rightarrow \quad T}{(\phi \quad \Rightarrow \quad T) \quad \Rightarrow \quad \psi \quad \Rightarrow \quad \phi \quad \Rightarrow \quad T} \quad \text{conv}
\]

\[
\frac{\varnothing \quad \Rightarrow \quad T}{(\phi \quad \Rightarrow \quad T) \quad \Rightarrow \quad \psi \quad \Rightarrow \quad T} \quad \text{conv}
\]

Let $A$ be the left premiss of the $\Rightarrow_L$ rule, and $B$ be the premiss of the $c'$ rule, then observe that weight has indeed increased, since $\omega(A) = 2$ and $\omega(B) = 1$.

Fortunately, this example witnesses the only way in which weight can increase; and, moreover, the phenomenon does not compound, so can be bounded a priori. To capture this formally we implement a track-and-trace scheme such that we can effectively analyse where and how the undesirable growth happens and the bound it; the scheme uses a labelling of sequents in proofs that annotates each multiplicative context-former and connective with a label that is copied only in uses of contraction. Monitoring how the labels spread will show that the interaction between contraction as in Example 4.14 is indeed the only time weight increases.

Definition 4.15 (Labelling). A label is an integer. A formula is labelled if every multiplicative connective occurring it carries a label, and a bunch is labelled if every formula in it is labelled and every multiplicative context-former in it carries a label. A sequent is labelled when both the context and the formula are labelled.

A proof is well-labelled if every sequent in it is labelled according to the following scheme:

Base Case. In the end-sequent every multiplicative connective and context-former is given a unique integer label.
The maximal size of a scan of $\Gamma$ of nodes between it and the root of the proof. \[\Gamma\] is the same and uses increasing functions. The maximal scans of a bunch provide an upper-bound for the weight of the bunch: \[\lambda(\Gamma) \geq \omega(\Gamma).\]

**Definition 4.17 (Scan).** Let $\Gamma$ be a labelled bunch, a scan $\ell$ of $\Gamma$ is a multi-set of labels satisfying the following:
- if $\Gamma$ is basic, then $\ell$ is the multi-set of all labels in $\Gamma$.
- if $\Gamma = \Gamma' \circ \Gamma''$, then $\ell = \ell' \cup [n] \cup \ell''$, where $\ell'$ and $\ell''$ are scans of $\Gamma'$ and $\Gamma''$, respectively.
- if $\Gamma = \Gamma' \circ \Gamma''$, then $\ell$ is a scan of either $\Gamma'$ or $\Gamma''$.

The maximal size of a scan of $\Gamma$ is denoted $\lambda(\Gamma)$.

**Example 4.18.** There is only one scan of $\Gamma = (\phi \rightarrow^0 T) \rightarrow \psi \rightarrow^0 \phi$, the context of a bunch in Example 4.16, assuming the formulas contain no multiplicative connectives, and it is $[0,0]$.

The maximal scans of a bunch provide an upper-bound for the weight of the bunch:

**Lemma 4.19.** $\lambda(\Gamma) \geq \omega(\Gamma)$.

**Proof.** We proceed by induction on the structure of $\Gamma$:

**Base Case.** If $\Gamma$ is basic then $\lambda(\Gamma)$ is the size of a scan that contains a label for every multiplicative connective and context-former in $\Gamma$, and the weight of $\Gamma$ cannot be more since it only counts some connectives.

**Induction Step.** There are two cases to consider, either $\Gamma$ is multiplicative or $\Gamma$ is additive. Both cases follow from the induction hypothesis since the form of the computation of a scan and weight is the same and uses increasing functions.

The only task that remains for bounding weight is then to bound the size of scans, which is done by witnessing the assertion that the only way for a connective to be counted twice is through the resolution rules:

**Lemma 4.20.** Let $D$ be a well-labelled proof and let $s = (\Gamma \equiv \phi) \in D$ be arbitrary. For any scans $\ell$ and $\ell'$ of distinct bunches $\Delta$ and $\Delta'$, respectively, satisfying either that $\Delta$ and $\Delta'$ are in the same additive or multiplicative set in $\Gamma$ or $\Delta = \Gamma$ and $\Delta' = \phi$, a label occurs at most twice $\ell \cup \ell'$.

**Proof.** We proceed by induction on the height of the occurrence of $(\Gamma \equiv \phi)$ (i.e., the number of nodes between it and the root of the proof).
We present two detailed examples. which is done by finding bounds for the kind of reductions that appear in them; that is, we observe where

\[ \ell \]

ACM Trans. Comput. Logic, Vol. 1, No. 1, Article 1. Publication date: January 1.

A. Gheorghiu, S. Docherty and D. Pym

.. \[ \ell \]

labels in \[ \pi \]

set (i.e., the one containing the active bunches) as otherwise the action was not regimented.

It remains only to bound multiplicity, which is done by witnessing the effect of Gentzen’s control régime on it.

Remark. System LBI does not satisfy an analogous statement to Lemma 4.20 because \( c_{\phi, \pi} \) may introduce a multiplicative context-formers that would not be justified by any label in the end-sequent, same if \( \text{rad} \) was present, which is why the appeal to systems without these rules (i.e., system dLBI) is important.

A bound on weight follows from the above lemmas:

**Lemma 4.21.** \( \mathcal{D} : \emptyset \vdash \bar{S} \Rightarrow \omega(\mathcal{D}) \leq 2\lambda(\bar{S}) \).

Proof. Let \( \mathcal{D} \) be well-labelled, let \( s = (\Gamma \Rightarrow \phi) \in \mathcal{D} \), and let \( n \) be the number of distinct labels in \( s \). By definition \( \omega(s) = \omega(\Gamma) + \omega(\phi) \), and (by Lemma 4.19 and Lemma 4.20) the bound \( \omega(s) \leq 2n \) holds since a label cannot be counted more than twice in the multi-set union of their respective scans. Since \( n \leq \lambda(\bar{S}) \) since no new labels can have been created, the result follows.

**4.3 Bounding Multiplicity.**

It remains only to bound multiplicity, which is done by witnessing the effect of Gentzen’s control régime on it.

We begin by establishing bounds for the normalization and loading phases of regimented proofs, which is done by finding bounds for the kind of reductions that appear in them; that is, we observe that the conclusion of an action in a regimented proof contains at most one non-trivial additive set (i.e., the one containing the active bunches) as otherwise the action was not regimented.

...
LEMMA 4.22. Let $\Gamma \triangleright^* \Gamma'$, and suppose there is a unique non-trivial additive set in $\Gamma$, then any non-trivial additive set in $\Gamma'$ is also unique.

Proof. If the reduction is a permutation, then we are done. If all the additive sets in $\Gamma'$ are trivial, then we are done. Otherwise, let $A$ be the unique non-trivial additive set of $\Gamma$, and note that it contains the bunch that is removed by the reduction. If no bunch has become duplicit after the reduction, then $A$ is still unique in $\Gamma'$ and we are done. Otherwise, any bunch that becomes duplicit after reduction must be in the duplicity set of a bunch containing the additive component corresponding to $A$ as a sub-bunch, as otherwise it would not have been affected by reduction. Furthermore, the additive set $A$ must have become trivial as otherwise $\Gamma$ contained two non-trivial and distinct additive sets contradicting the hypothesis. The new non-trivial additive set must be unique for the same reason: otherwise there would be a non-trivial additive set containing it contradicting the hypothesis. □

LEMMA 4.23. If $D$ is a normalizing strategy witnessing $S \triangleright^* S'$, and there is unique non-trivial additive set in $S$, then $\mu(D) \leq \mu(S)$

Proof. It follows from Lemma 4.22 that every sequent in the normalizing strategy has at most one non-trivial additive set. Let $\Gamma$ and $\Gamma'$ be the contexts of a redex and reduct respectively, and suppose $\Gamma$ has a non-trivial additive set $A$. If the reduction does not increase the duplicity of any sub-bunch, then trivially $\mu(\Gamma') \leq \mu(\Gamma)$ as required. Otherwise there is $\Delta \leq \Gamma$ such that after reduction its duplicity increases, but then there must be a bunch $\Delta'$ in its duplicity set after reduction, not there before the reduction, containing the additive component corresponding to $A$ as a sub-bunch. However, by uniqueness of $A$, there can be at most one such-bunch, hence the duplicity set has increased by at most one. At the same time, it must be that the duplicity of the additive set decreases, as otherwise $A$ was not unique in $\Gamma$. Hence, at most $\mu(\Gamma') = \mu(\Gamma)$, as required. □

LEMMA 4.24. In a regimented action $R(P_1, ..., P_n, C)$, the premisses $P_i$ and the conclusion $C$ have a unique non-trivial additive set.

Proof. In every rule in dLBI, every premiss $P_i$ has at most one additive-set that is active, and (by the definition of regimented action) if it is non-trivial then it is uniquely so. Similarly, for the conclusion $C$ there is at most one non-trivial additive set as otherwise the non-trivial additive set is contained in some bunch that is contained in some other non-trivial additive set in the context, but then another element of the duplicity contains a sub-bunch that was not active in the inference but that is duplicit, a contradiction. □

Since loading strategies are negative strategies whose corresponding positive strategy satisfies the hypothesis of the above lemma, the same result applies; that is, one can bound the multiplicity of the phase by the multiplicity of the end-sequent.

The following two technical results will establish the bounds on the actions — the bounded growth property.

LEMMA 4.25. Suppose all duplicit sub-bunches of $\Gamma(\Delta)$ are contained in $\Delta$, then for any $\Delta'$,

$$\mu(\Gamma(\Delta')) \leq 1 + \mu(\Delta')$$

Proof. In $\Gamma(\Delta')$ there is at most one duplicity set outside $\Delta'$ containing more than one member, as otherwise there were duplicit sub-bunches outside $\Delta$. Furthermore, for the same reason, this duplicity set has at most two members. Hence, $\mu(\Gamma(\Delta')) \leq 1 + \mu(\Delta')$ as required. □

LEMMA 4.26. Let $\Gamma = \Pi_1 \circlof \Pi_2$ where $\circlof$ is a context-former and $\Pi_1, \Pi_2$ are normal, then $\mu(\Gamma) \leq 1$
Proof. For any additive set $A$ in $\Gamma$ of one the following hold: either $A$ is an additive set of $\Pi_1$, $A$ is an additive set of $\Pi_2$, or $A$ is the topset of $\Gamma$. By normality hypothesis for $\Pi_1$ and $\Pi_2$, in the first two cases $\partial A = 0$, and in the last $\partial A \leq 1$ because $\Pi_1$ and $\Pi_2$ may contain some permutation equivalent bunches. Hence, $\mu(\Gamma) \leq 1$ as required.

**Lemma 4.27 (Bounded Growth Property).** Let $r(P_1, ..., P_n, C)$ be a regimented action. Then

$$\mu(P_1), ..., \mu(P_n), \mu(C) \leq 3$$

Proof. Let $\Gamma_i$ be the contexts of $P_i$, and let $\Lambda_i$ be a minimal sub-bunch of $\Gamma_i$ containing all the active sub-bunches in the inference. If $\Sigma$ is any normal bunch not appearing in $\Gamma_i$ then, by hypothesis, $\Gamma_i(\Sigma)$ is normal. Hence, by Lemma 4.25,

$$\mu(\Gamma_i(\Lambda_i)) \leq \mu(\Lambda_i) + 1$$

By case analysis $\mu(\Lambda_i) \leq 1$, since in every rule there are at most two sub-bunches that are active in the same additive set. Hence, $\mu(\Gamma_i) \leq 2$, whence $\mu(P_i) \leq 2$, as required.

It remains to establish the bound for $\hat{C}$, which we do by cases analysis on $r$ with the following partition:

$$\mathcal{R}_1 := \{\lor R\} \quad \mathcal{R}_2 := \{\to_R, \to_R, \land_R, \land_R\} \quad \mathcal{R}_3 := \{T_L, T_L, \land_L, \land_L, \land\} \quad \mathcal{R}_4 := \{\to_L, \to_L, w', \lor_L\}$$

The variants on $\to_L, \to_R, \land_R$ are identified with the rule itself. The cases are handled as follows:

**Case of $\mathcal{R}_1$.** There are no active sub-bunches, so $\mu(\hat{C}) = 0$, by hypothesis.

**Case of $\mathcal{R}_2$.** The conclusion of the rules takes the form $\Pi_1 \circ \Pi_2$, where $\circ$ is a context-former, and $\Pi_1$ and $\Pi_2$ are normal, by hypothesis. Thus, by Lemma 4.26, $\mu(\hat{C}) \leq 1$.

**Case of $\mathcal{R}_3$.** All the rules are substitutions in a bunch $\Gamma(\Delta)$ by a bunch $\Delta'$, hence, by Lemma 4.25, $\mu(\hat{C}) \leq \mu(\Delta') + 1$. However, $\mu(\Delta') = 0$ since $\Delta'$ is normal, thus $\mu(\hat{C}) \leq 1$.

**Case of $\mathcal{R}_4$.** All the rules are substitution in $\Gamma(\Delta)$ by a bunch $\Delta'$, hence, by Lemma 4.25, $\mu(\hat{C}) \leq \mu(\Delta') + 1$. For the $w'$ observe $\Delta' = \Sigma \circ \Sigma'$ where $\Sigma$ and $\Sigma'$ are normal by hypothesis, so by Lemma 4.26 we have $\mu(\Delta') \leq 1$. Similarly, for $\to L$ and $\to R$ we have $\Delta' = \Sigma \circ \Sigma' \circ \Sigma''$ and each of $\Sigma, \Sigma'$ and $\Sigma$ is normal. Thus by Lemma 4.26 twice $\mu(\Delta') \leq 2$. In both cases we have the bound $\mu(\hat{C}) \leq 3$ as required.

The bound for regimented proofs follows immediately.

**Lemma 4.28.** $\mathcal{D} : \emptyset + \hat{S} \Longrightarrow \mu(\mathcal{D}) \leq 3$.

Proof. We proceed by induction on the number of actions in $\mathcal{D}$; that is, the maximum number of regimented actions in any branch from root to leaf.

**Base Case.** If the number of actions is one, then $\mathcal{D}$ consists of one node which is an axiom, and therefore normal. Thus $\mu(\mathcal{D}) = \mu(\hat{S}) = 0$.

**Induction Step.** Assume the property holds for proofs with at most $n$ actions, and suppose $\mathcal{D}$ has $n + 1$. Let $(\hat{P}_1, ..., \hat{P}_n, \hat{C})$ be the last action, then there are sub-trees $\mathcal{D}_i$ that are proofs of $\hat{P}_i$ respectively. The proofs $\mathcal{D}_i$ have at most $n$ actions, thus by the induction hypothesis $\mu(\mathcal{D}_i) \leq 3$. By Lemma 4.23, Lemma 4.24, and Lemma 3.12, extending the $\mathcal{D}_i$ with loading strategies for $\hat{P}_i$ proofs $\mathcal{D}_i'$ satisfying $\mu(\mathcal{D}_i') \leq 3$. Moreover, applying $r$ gives a proof $\mathcal{D}_{\hat{C}}$ of $\hat{C}$, which by Lemma 4.27 satisfies $\mu(\mathcal{D}_{\hat{C}}) \leq 3$. Observe that $\hat{C}$ has all duplicit bunches in the same additive set, as otherwise there were duplicit bunches not active in the premises. Hence, extending with the normalizing strategy gives $\mathcal{D}$ that, by Lemma 4.23 and Lemma 4.24, satisfies $\mu(\mathcal{D}) \leq 3$ as required. \[\Box\]
4.4 Decidability
Since the measures have all been bounded, a proof-search procedure follows:

**Theorem 4.29.** There is an effective proof-search procedure for LBI.

**Proof.** For any putative goal $S$, it follows from Theorem 3.34 that it suffices to witness the existence or non- of a regimented dLBI of some normal form $\tilde{S}$ of the given sequent. We may restrict attention further by considering minimal regimented dLBI-proof; that is, call a proof **concise** if no sequent appears twice in the same branch, then we may restrict attention to concise regimented proofs by removing any section of the branch of a given proof where a repetition occurs.

Suppose $\tilde{S}$ has a concise regimented dLBI-proof $D$, then by Lemma 4.21, Lemma 4.28, and Lemma 4.13, there is a computable $b$ such that $\omega(D) \leq b$, $\mu(D) \leq b$, and $\delta(D) \leq b$. Let $\Sigma$ be the set of formulas in $\tilde{S}$ closed under subformula relation, which is finite since $S$ is finite and the subformula relation is a well-order. It follows from Lemma 4.12 that $S = \subseteq_{b,b,b}^\Sigma$ is primitive recursive. Observe that all the sequents of $D$ must appear in this set.

The rules of dLBI require at most two premisses, and there are no more than $2^{|S|}$ concise binary tree with labels from $S$ ending with $\tilde{S}$. Denote the set of such trees by $\mathcal{T}$ and note that it is also primitive recursive. If $D$ exists then $D \in \mathcal{T}$, thus checking correctness of the proof candidates completes the proof-search procedure. $\square$

5 Conclusion
We have shown, using traditional and exclusively syntactic techniques from proof theory, that one can restrict the proof-search space for BI, with respect to its sequent calculus, to a primitive recursive set. Analysing Gentzen’s decidability argument for classical (and intuitionistic logic) gave a partial solution, but BI combines additional multiplicative structures, and the latter are poorly behaved in the sense that they admit a limited form of contraction in the unit laws. The problem is solved using the ideas of Dyckhoff’s contraction-elimination argument for intuitionistic logic; that is, a sufficient amount of the offending contraction rule is embedded into the remaining rules so that it becomes impotent. Effective proof-search is a corollary.

The analysis of Gentzen’s methodology yields the notion of a regimented proof, and future work includes exploring the significance of these proofs within the study of proof-search in particular, and proof theory in general. Moreover, since the bounding of regimented proofs largely entails an analysis of data and information of structures (formulas and bunches) within a sequent, one might also consider the semantic significance of such proofs. The measures of weight, depth, and multiplicity also have some significance; indeed, similar measures have been studied elsewhere in the presence of multiplicative and additive structures [10, 15].

Curiously, the neighbourhood of BI is largely undecidable; that is, there cannot be a decision procedure for Boolean BI, Separation Logic, or Classical BI [4, 5, 17, 18]. The analogous treatment of regimented proofs in their respective sequent calculi (of the same general form) fails because cut-free proofs are not complete [3]. However, there is a uniform proof theory using hypersequent calculi for a class of logics related to BI for which the problem remains open [7]. A parallel treatment of substructural logics has been completed successfully [1, 6, 23]. However, there are uniform decidability results for a family of weaker bunched logics: layered graph logics [8].

Finally, one may consider the relationship between the combinatorial and proof theoretic techniques presented here with Kripke’s *tour de force* decidability argument for the full Lambek calculus with contraction [16, 25], which has also seen successful generalization [2, 19, 23]. In particular, one might consider to what extent the combinatorics are related.
ACKNOWLEDGMENTS

We thank Didier Galmiche, Daniel Méry, James Brotherston, Sonia Marin, and Timo Lang, as well as the referees of an earlier version, for their corrections and other comments.

REFERENCES

[1] Balasubramanian, A.R., Lang, T., Ramanayake, R.: Decidability and Complexity in Weakening and Contraction Hypersequent Substructural Logics. In: Symposium on Logic in Computer Science. pp. 1–13. IEEE (2021). https://doi.org/10.1109/LICS52264.2021.9470733

[2] Belnap Jr, N.D., Wallace, J.R.: A Decision Procedure for the System E of Entailment with Negation. Mathematical Logic Quarterly 11(4), 277–289 (1965)

[3] Brotherston, J.: Bunched Logics Displayed. Studia Logica 100(6), 1223–1254 (2012)

[4] Brotherston, J., Kanovich, M.: Undecidability of Propositional Separation Logic and its Neighbours. In: 2010 25th Annual IEEE Symposium on Logic in Computer Science. pp. 130–139. IEEE (2010)

[5] Brotherston, J., Kanovich, M.: Undecidability of Propositional Separation Logic and its Neighbours. Journal of the ACM (JACM) 61(2), 1–43 (2014)

[6] Ciabattoni, A., Galatos, N., Terui, K.: Algebraic Proof Theory for Substructural Logics: Cut-elimination and Completion. Annals of Pure and Applied Logic 163(3), 266 – 290 (2012)

[7] Ciabattoni, A., Ramanayake, R.: Bunched Hypersequent Calculi for Distributive Substructural Logics. In: Eiter, T., Sands, D. (eds.) 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning. EPiC Series in Computing, vol. 46, pp. 417–434 (2017). https://doi.org/10.29007/ngp3, https://easychair.org/publications/paper/sr2D

[8] Docherty, S.: Bunched Logics: A Uniform Approach. Ph.D. thesis, University College London (2019)

[9] Dyckhoff, R.: Contraction-Free Sequent Calculi for Intuitionistic Logic. The Journal of Symbolic Logic 57(3), 795–807 (1992)

[10] Galatos, N., Jipsen, P.: Distributive Residuated Frames and Generalized Bunched Implication Algebras. Algebra universalis 78(3), 303–336 (2017)

[11] Galmiche, D., Méry, D., Pym, D.: The Semantics of BI and Resource Tableaux. Mathematical Structures in Computer Science 15(6) (2005). https://doi.org/10.1017/S0960129505004858

[12] Gentzen, G.: The Collected Papers of Gerhard Gentzen. North-Holland Publishing Company (1969)

[13] Gheorghiu, A., Marin, S.: Focused Proof-search in the Logic of Bunched Implications. FOSSACS 2021: Foundations of Software Science and Computation Structures 24th International Conference p. 247–267 (March 2021)

[14] Kamide, N.: Substructural Logics with Mingle. Journal of Logic, Language, and Information 11(2), 227–249 (2002)

[15] Kaminski, M., Francez, N.: The Lambek Calculus Extended with Intuitionistic Propositional Logic. Studia Logica pp. 1051–1082 (2016)

[16] Kripke, S.: The Problem of Entailment. Journal of Symbolic Logic p. 324 (1959)

[17] Larchey-Wendling, D., Galmiche, D.: The Undecidability of Boolean BI Through Phase Semantics. In: 2010 25th Annual IEEE Symposium on Logic in Computer Science. pp. 140–149. IEEE (2010)

[18] Larchey-Wendling, D., Galmiche, D.: Nondeterministic Phase Semantics and the Undecidability of Boolean BI. ACM Transactions on Computational Logic (TOCL) 14(1), 1–41 (2013)

[19] Meyer, R.K.: Topics in Modal and Many-valued Logic. Ph.D. thesis, University of Pittsburgh (1966)

[20] Newman, M.H.A.: On Theories with a Combinatorial Definition of "Equivalence.". Journal of Symbolic Logic 7(3), 123–123 (1942)

[21] O’Hearn, P.W., Pym, D.J.: The Logic of Bunched Implications. Bulletin of Symbolic Logic 5(2), 215–244 (1999)

[22] Pym, D.J.: The Semantics and Proof Theory of the Logic of Bunched Implications, Applied Logic Series, vol. 26. Springer Netherlands (2002)

[23] Ramanayake, R.: Extended Kripke Lemma and Decidability for Hypersequent Substructural Logics. In: Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science. p. 795–806. LICS ’20, Association for Computing Machinery (2020)

[24] Read, S.: Relevant logic : A Philosophical Examination of Inference. Basil Blackwell, Oxford (1988)

[25] Urquhart, A.: The Complexity of Decision Procedures in Relevance Logic. In: Truth or Consequences, pp. 61–76. Springer (1990)