The Inverse Method for the Logic of Bunched Implications

Kevin Donnelly[1], Tyler Gibson[2], Neel Krishnaswami[3], Stephen Magill[3], and Sungwoo Park[3]

[1] Department of Computer Science, Boston University, 111 Cummington Street, Boston MA 02215, USA,
kevind@bu.edu
[2] Department of Philosophy, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, USA,
tylerg@andrew.cmu.edu
[3] Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, USA,
{neelk,smagill,gla}@cs.cmu.edu

Abstract. The inverse method, due to Maslov, is a forward theorem proving method for cut-free sequent calculi that relies on the subformula property. The Logic of Bunched Implications (**BI**), due to Pym and O'Hearn, is a logic which freely combines the familiar connectives of intuitionistic logic with multiplicative linear conjunction and its adjoint implication. We present the first formulation of an inverse method for propositional **BI** without units. We adapt the sequent calculus for **BI** into a forward calculus. The soundness and completeness of the calculus are proved, and a canonical form for bunches is given.

1 Introduction

1.1 The Logic of Bunched Implications

The study of substructural logics, beginning with linear logic [10], has shown the usefulness of restricting the structural rules of weakening, contraction, commutativity and associativity. These logics have shown promise in modeling a variety of situations, including reasoning about computations. For example, using the resource interpretation of linear logic, we can reason about availability and use of resources that cannot be regenerated. The example of linear logic also shows the usefulness of making available controlled uses of the eliminated structural rules (which in linear logic comes in the form of the ! and ? modalities).

The Logic of Bunched Implications (**BI**) [13,15] comes from freely combining the additive conjunction of intuitionistic logic with the multiplicative conjunction of linear logic. It is important to note that while the rules for introducing and eliminating multiplicatives are the same as those of linear logic, the use-once resource semantics of the multiplicative connectives no longer holds

in **BI** because of the possibility of nested additives. In the presence of two context forming operations, there naturally arises two different implications with the following adjoint relationships.

$$A * B \vdash C \iff A \vdash B \twoheadrightarrow C \text{ and } A \wedge B \vdash C \iff A \vdash B \supset C$$

The free combination of these conjunctions and implications leads to a logic with tree structured contexts, and a calculus in which the multiplicative conjunction distributes over the additive conjunction, but the inverse, factoring multiplicatives out of additives, does not hold. This leads to a lack of a structural canonical form, which presents a challenge to the standard formulation of the inverse method.

1.2   The Inverse Method

The inverse method [7,14] is a saturation based theorem proving technique for sequent calculi which is related to resolution [4]. First proposed by Maslov [12], the inverse method starts from a collection of axioms in a database and works forward by applying rules to the sequents in the database and adding the results of these rules back into the database until the goal sequent has been derived.

Proof search in the inverse method is of a very different character than tableau search, which must deal with disjunctive non-determinism in searching backward through the proof tree. In the inverse method we are concerned with conjunctive non-determinism, as we work forward our information grows monotonically. So the main challenge of inverse method search is to derive as few 'redundant' sequents as possible while still retaining completeness. The key property that the inverse method uses to limit conjunctive non-determinism is that in all inference rules, each of the formulas of the premises are subformulas of some formula in the conclusion. This implies that even if we restrict ourselves to only apply rules when the conclusion of the rule contains only subformulas of the goal sequent, we will still have a complete search strategy. In addition, this lets us disprove a theorem by exhausting this search space. This makes it also important to restrict rules like weakening, which would otherwise always be applicable in the forward direction. This is dealt with in the intuitionistic inverse method by eliminating weakening and changing the completeness theorem [7], a similar fix works for **BI**.

As an example of the inverse method in **IL**, consider the (intuitionistically true) goal proposition $((p \supset q) \vee (p \supset r)) \supset (p \supset (q \vee r)))$. We begin by enumerating signed subformulas to find possible initial sequents. The subformulas are:

$+ ((p \supset q) \vee (p \supset r)) \supset (p \supset (q \vee r)))$	$- p$
$- ((p \supset q) \vee (p \supset r))$	$- q$
$+ (p \supset (q \vee r)))$	$+ p$
$- (p \supset q)$	$- r$
$- (p \supset r)$	$+ q$
$+ (q \vee r)$	$+ r$

Each pair of a positive proposition p and its negative indicates a possible use of the axiom $p \vdash p$ in the proof of the goal. So we begin with a database including, $p \vdash p$, $q \vdash q$ and $r \vdash r$. We work forward from these axioms by applying rules of the intuitionistic forward calculus whenever the conclusion contains only the signed subformulas above. When we reach a sequent that can be weakened to the goal sequent, we are done. Theorem proving proceeds as follows (some sequents irrelevant to the final proof are not shown):

1. $p \vdash p$	init
2. $q \vdash q$	init
3. $r \vdash r$	init
4. $q \vdash q \vee r$	$\vee R_1$ 2
5. $r \vdash q \vee r$	$\vee R_2$ 3
6. $p \supset q, p \vdash q$	$\supset L$ 2
7. $p \supset r, p \vdash r$	$\supset L$ 3
8. $p \supset q, p \vdash q \vee r$	$\vee R_1$ 6
9. $p \supset r, p \vdash q \vee r$	$\vee R_2$ 7
10. $(p \supset q) \vee (p \supset r), p \vdash q \vee r$	$\vee L$ 8 9
11. $(p \supset q) \vee (p \supset r) \vdash p \supset (q \vee r)$	$\supset R$ 10
12. $\vdash ((p \supset q) \vee (p \supset r)) \supset (p \supset (q \vee r))$	$\supset R$ 11

One distinct advantage of inverse method theorem proving in **BI** is that the resource distribution problem for multiplicative connectives disappears in the forward direction. If we read a rule like

$$\frac{\Gamma \vdash \varphi \quad \Delta \vdash \psi}{\Gamma, \Delta \vdash \varphi * \psi} \; *R$$

in the reverse direction as in tableau proof-search, it is not obvious how to split the resources in the context between Γ and Δ, so this must be calculated during proof search. This can be handled in both linear logic and **BI** by using Boolean constraint methods as in [11]. However, in the forward direction the distribution problem simply disappears.

Unfortunately reading the rules in the forward direction introduces a new resource problem for the units. In particular, the for the rule

$$\frac{\Gamma(\varnothing_a) \vdash \varphi}{\Gamma(\top) \vdash \varphi} \; \top L$$

and similarly for IL, the rule is always applicable in the forward direction and it is not clear how many times it must be used. The solution for the similar problem in linear logic is given in [6]. It is not clear if a similar fix would work for **BI**, so we omit units from our inverse method.

In our paper we formulate a forward sequent calculus for propositional **BI** without units, which is suitable for inverse method theorem proving. We prove the soundness and completeness of our method relative to the sequent calculus rules given in [15, Ch. 6]. We describe a canonical form for bunches suitable for

use in an implementation, and describe our SML implementation of our inverse method. The main contribution of our paper is in defining for the first time an inverse method for **BI**. In particular, we overcome the lack of a structural canonical form.

2 Propositional BI Without Units

In this section we present a sequent calculus for the propositional fragment of **BI** without propositional units ($\bot, I$, and $\top$). We leave out the units because their inclusion in the inverse method is quite involved, see [6] for the challenges presented by units in the inverse method for linear logic. Similar issues apply to units in **BI**.

Firstly, we have two types of connectives:

$$\begin{array}{ll} \text{Additives} & \wedge \supset \vee \\ \text{Multiplicatives} & * \mathbin{-\!*} \end{array}$$

The additive connectives are the same as those of intuitionistic logic and the multiplicatives come from intuitionistic linear logic. The contexts of **BI**, referred to as bunches, are trees where the leaves are formulas or empty bunches and the inner nodes are multiplicative (,) or additive (;) context forming operations. Note that while we leave out the propositional constants for units ($\bot, I$, and $\top$), contextual units (empty bunches) may still appear in bunches.

$$\begin{array}{llll} \text{Bunches} & \Gamma ::= & \varphi & \text{propositional assumption} \\ & | & \varnothing_m & \text{multiplicative unit} \\ & | & \Gamma, \Gamma & \text{multiplicative combination} \\ & | & \varnothing_a & \text{additive unit} \\ & | & \Gamma; \Gamma & \text{additive combination} \end{array}$$

In the following we write $\Gamma(\Delta)$ to mean a bunch in which Δ is a subbunch and $\Gamma(\Delta')$ to mean the replacement of Δ by Δ' in $\Gamma(\Delta)$. The following equivalence on bunches is used to convert between isomorphic bunches.

Definition 1 (Coherent Equivalence). $\equiv$ *is the least equivalence relation on bunches satisfying:*

1. *Commutative monoid equations for $\varnothing_a$ and ;*
2. *Commutative monoid equations for $\varnothing_m$ and ,*
3. *Congruence: if $\Delta \equiv \Delta'$ then $\Gamma(\Delta) \equiv \Gamma(\Delta')$*

In section 3.3 we give a canonical form for bunches which uses n-ary operations, however all rules are given in terms of the simple binary formulation.

Judgments are of the form $\Gamma \vdash \varphi$ where Γ is a bunch and φ is a formula. The sequent calculus rules for propositional **BI** without units are given in Figure 1. The cut-rule is admissible in this system [15, Ch. 6].

Identity and Structure

$$\frac{\Gamma \vdash \varphi}{\Gamma' \vdash \varphi} \ (\Gamma \equiv \Gamma') \ E \qquad \frac{}{\varphi \vdash \varphi} \ INIT$$

$$\frac{\Gamma(\Delta) \vdash \varphi}{\Gamma(\Delta; \Delta') \vdash \varphi} \ W \qquad \frac{\Gamma(\Delta; \Delta) \vdash \varphi}{\Gamma(\Delta) \vdash \varphi} \ C$$

Additives

$$\frac{\Gamma \vdash \varphi \quad \Delta(\Delta'; \psi) \vdash \chi}{\Delta(\Delta'; \Gamma; \varphi \supset \psi) \vdash \chi} \supset L \qquad \frac{\Gamma; \varphi \vdash \psi}{\Gamma \vdash \varphi \supset \psi} \supset R$$

$$\frac{\Gamma(\varphi; \psi) \vdash \chi}{\Gamma(\varphi \wedge \psi) \vdash \chi} \wedge L \qquad \frac{\Gamma \vdash \varphi \quad \Delta \vdash \psi}{\Gamma; \Delta \vdash \varphi \wedge \psi} \wedge R$$

$$\frac{\Gamma \vdash \varphi_i}{\Gamma \vdash \varphi_1 \vee \varphi_2} \ (i = 1, 2) \vee R_i \qquad \frac{\Gamma(\varphi) \vdash \chi \quad \Gamma(\psi) \vdash \chi}{\Gamma(\varphi \vee \psi) \vdash \chi} \vee L$$

Multiplicatives

$$\frac{\Gamma(\varphi, \psi) \vdash \chi}{\Gamma(\varphi * \psi) \vdash \chi} \ *L \qquad \frac{\Gamma \vdash \varphi \quad \Delta \vdash \psi}{\Gamma, \Delta \vdash \varphi * \psi} \ *R$$

$$\frac{\Gamma \vdash \varphi \quad \Delta(\Delta', \psi) \vdash \chi}{\Delta(\Delta', \Gamma, \varphi \mathbin{-\!*} \psi) \vdash \chi} \ \mathbin{-\!*} L \qquad \frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \mathbin{-\!*} \psi} \ \mathbin{-\!*} R$$

Fig. 1. Sequent calculus rules for core propositional **BI**

3   An Inverse Method for BI

3.1   The Calculus

Following the general method for producing a weakening-free forward calculus from a backward sequent calculus [7], we adapt the sequent calculus for **BI** into a calculus suitable for the inverse method. The rules for our forward sequent calculus for **BI** are as given in Figure 2.

We annotate the rules in the new system with superscript I to differentiate them from the old rules, the judgment of our new system has the form $\Gamma \vdash^{I} \varphi$.

The first step in generating a forward sequent calculus is to eliminate the weakening rule (or reformulate the rules so weakening is not built into them, if there is no explicit weakening rule). Since our starting point is a sequent calculus with an explicit weakening rule, we remove it. In order to state the completeness theorem for the weakening-free system we need the following.

Identity and Structure

$$\frac{\Gamma(\Delta_1; \Delta_2) \vdash^{\mathtt{I}} \varphi \quad (\Delta \in \mathrm{lubs}_{\sqsubseteq} (\Delta_1) (\Delta_2)) \not\equiv \Delta_1; \Delta_2}{\Gamma(\Delta) \vdash^{\mathtt{I}} \varphi} \; C^{\mathtt{I}}$$

$$\frac{\Gamma \vdash^{\mathtt{I}} \varphi}{\Gamma' \vdash^{\mathtt{I}} \varphi} \; (\Gamma \equiv \Gamma') \, E^{\mathtt{I}} \qquad \frac{}{\varphi \vdash^{\mathtt{I}} \varphi} \; INIT^{\mathtt{I}}$$

Additives

$$\frac{\Gamma \vdash^{\mathtt{I}} \varphi \quad \Delta(\Delta'; \psi) \vdash^{\mathtt{I}} \chi}{\Delta(\Delta'; \Gamma; \varphi \supset \psi) \vdash^{\mathtt{I}} \chi} \supset L^{\mathtt{I}} \qquad \frac{\Gamma; \varphi \vdash^{\mathtt{I}} \psi}{\Gamma \vdash^{\mathtt{I}} \varphi \supset \psi} \supset R_1^{\mathtt{I}}$$

$$\frac{\Gamma \vdash^{\mathtt{I}} \varphi \quad \Delta \vdash^{\mathtt{I}} \psi}{\Gamma; \Delta \vdash^{\mathtt{I}} \varphi \wedge \psi} \wedge R^{\mathtt{I}} \qquad \frac{\Gamma \vdash^{\mathtt{I}} \psi}{\Gamma \vdash^{\mathtt{I}} \varphi \supset \psi} \supset R_2^{\mathtt{I}}$$

$$\frac{\Gamma(\varphi_i) \vdash^{\mathtt{I}} \chi}{\Gamma(\varphi_1 \wedge \varphi_2) \vdash^{\mathtt{I}} \chi} \; (i=1,2) \; \wedge L_i^{\mathtt{I}} \quad \frac{\Gamma \vdash^{\mathtt{I}} \varphi_i}{\Gamma \vdash^{\mathtt{I}} \varphi_1 \vee \varphi_2} \; (i=1,2) \; \vee R_i^{\mathtt{I}}$$

$$\frac{\Gamma(\varphi) \vdash^{\mathtt{I}} \chi \quad \Delta(\psi) \vdash^{\mathtt{I}} \chi \quad \Sigma(p) \in \mathrm{lubs}_{\sqsubseteq} (\Gamma(p)) (\Delta(p))}{\Sigma(\varphi \vee \psi) \vdash^{\mathtt{I}} \chi} \vee L^{\mathtt{I}}$$
$$\text{for new parameter } p, \text{ not appreaing in } \Gamma, \Delta \text{ or } \Sigma$$

Multiplicatives

$$\frac{\Gamma(\varphi, \psi) \vdash^{\mathtt{I}} \chi}{\Gamma(\varphi * \psi) \vdash^{\mathtt{I}} \chi} *L^{\mathtt{I}} \qquad \frac{\Gamma \vdash^{\mathtt{I}} \varphi \quad \Delta \vdash^{\mathtt{I}} \psi}{\Gamma, \Delta \vdash^{\mathtt{I}} \varphi * \psi} *R^{\mathtt{I}}$$

$$\frac{\Gamma \vdash^{\mathtt{I}} \varphi \quad \Delta(\Delta', \psi) \vdash^{\mathtt{I}} \chi}{\Delta(\Delta', \Gamma, \varphi \mathbin{-\!*} \psi) \vdash^{\mathtt{I}} \chi} \mathbin{-\!*} L^{\mathtt{I}} \qquad \frac{\Gamma, \varphi \vdash^{\mathtt{I}} \psi}{\Gamma \vdash^{\mathtt{I}} \varphi \mathbin{-\!*} \psi} \mathbin{-\!*} R^{\mathtt{I}}$$

Fig. 2. Forward sequent calculus rules for core propositional **BI**

Definition 2 (Bunch Ordering). $\sqsubseteq$ *is the transitive, reflexive (with respect to $\equiv$) closure of $\Gamma(\Delta) \sqsubseteq \Gamma(\Delta; \Delta')$*

Note that this is equivalent to saying $\Delta \sqsubseteq \Delta'$ iff there is some derivation of $\Delta' \vdash \varphi$ from $\Delta \vdash \varphi$ using only rules W or E.

We next have to examine each of the rules to make sure they are still complete without weakening. One rule which obviously must be changed is $\supset R$ because the original system with weakening can derive $\varphi \vdash \psi \supset \varphi$ only because we first weaken ψ into the context, then use the $\supset R$ rule. To fix this we split the rule in two: $\supset R_1^{\mathtt{I}}$ which is the same as the old rule and $\supset R_2^{\mathtt{I}}$ which builds in the weakening step. We also split rule $\wedge L$ into $\wedge L_1^{\mathtt{I}}$ and $\wedge L_2^{\mathtt{I}}$ which build in weakening, the weakening-free original rule $\wedge L^{\mathtt{I}}$ is then derivable from $\wedge L_1^{\mathtt{I}}$, $\wedge L_2^{\mathtt{I}}$ and $C^{\mathtt{I}}$.

More interesting complications arise with rules C and $\vee L$. In the intuitionistic inverse method we simply remove C and build contraction into the rules by unioning the contexts that would otherwise be additively combined. It is fine to union together additively combined sequents in our **BI** inverse method, but this does not remove the need for rule C. The problem is that we may be able to use rule C only after weakening two additively joined subbunches to be the same. An example is the derivation:

$$\frac{\dfrac{\dfrac{(\varphi, \psi); (\varphi, \chi) \vdash \eta}{(\varphi, (\psi; \chi)); (\varphi, \chi) \vdash \eta} \; W}{(\varphi, (\psi; \chi)); (\varphi, (\psi; \chi)) \vdash \eta} \; W}{\varphi, (\psi; \chi) \vdash \eta} \; C$$

In fact we cannot eliminate rule C, because each pair of bunches does not have a unique least upper bound with respect to $\sqsubseteq$. For example the bunches (φ, ψ) and (φ, χ) have the minimal upper bounds $\varphi, (\psi; \chi)$ and $(\varphi, \psi); (\varphi, \chi)$ and neither can be obtained from the other using just weakening and equivalence. However each pair of bunches does have a finite minimal upper bound set, defined as follows.

Definition 3 (Minimal upper bound set). *S is a minimal upper bound set for Δ and Γ iff the following hold:*

$$\forall \Sigma \in S. \; \Delta \sqsubseteq \Sigma \wedge \Gamma \sqsubseteq \Sigma, \; and$$

$$\forall \Sigma. \; (\Delta \sqsubseteq \Sigma \wedge \Gamma \sqsubseteq \Sigma) \Rightarrow (\exists \Sigma' \in S. \; (\Sigma' \sqsubseteq \Sigma))$$

We write $\text{lubs}_{\sqsubseteq} (\Delta) (\Gamma)$ for the minimal set of upper bounds of Δ and Γ. We write it in curried notation to avoid the confusion of overloading "," to separate arguments as well as multiplicatively join bunches. Given the minimal upper bound set, we build weakening into the contraction rule ($C^{\mathcal{I}}$) by replacing two additively joined bunches with a common upper bound.

Rule $\vee L$ is affected in a similar way to rule C because the rule

$$\frac{\Gamma(\varphi) \vdash \chi \quad \Gamma(\psi) \vdash \chi}{\Gamma(\varphi \vee \psi) \vdash \chi} \; \vee L$$

requires that a single bunch with two different formulas plugged in prove a particular formula. However, in the inverse method we will generally have $\Delta(\varphi) \vdash \chi$ and $\Delta'(\psi) \vdash \chi$ in the database, and if $\Delta(\varphi) \sqsubseteq \Gamma(\varphi)$ and $\Delta'(\psi) \sqsubseteq \Gamma(\psi)$ we want to be able to apply the rule. So we make a new $\vee L^{\mathcal{I}}$ which uses the minimal upper bound set to achieve this.

In order to state the new $\vee L^{\mathcal{I}}$ we need to either extend the definition of bunches to allow for parameters or we can just think of this parameter as a new, unique atomic proposition. The reason that we need a parameter is easy to see if we think of $\Delta(-)$ and $\Delta'(-)$ as two bunches with holes, and we want to find a common upper bound with only a single hole, $\Gamma(-)$. The parameters are just place holders for the holes.

We first prove the soundness of our calculus by a fairly straightforward induction on the derivations.

Theorem 1 (Soundness). *If $\Gamma \vdash^{\mathcal{I}} \varphi$ then $\Gamma \vdash \varphi$.*

Proof (By structural induction).

case : *Derivation is* $\quad \dfrac{}{\varphi \vdash^{\mathcal{I}} \varphi} \; INIT^{\mathcal{I}}$
We immediately have $\varphi \vdash \varphi$ by rule INIT.

case : *Last rule is* $\quad \dfrac{\Gamma \vdash^{\mathcal{I}} \varphi \quad \Delta(\Delta'; \psi) \vdash^{\mathcal{I}} \chi}{\Delta(\Delta'; \Gamma; \varphi \supset \psi) \vdash^{\mathcal{I}} \chi} \supset L^{\mathcal{I}}$
By IH, have $\Gamma \vdash \varphi$ and $\Delta(\Delta'; \psi) \vdash \chi$.
We can use rule $\supset L$ to get $\Delta(\Delta'; \Gamma; \varphi \supset \psi) \vdash \chi$.

case : *Last rule is* $\quad \dfrac{\Gamma; \varphi \vdash^{\mathcal{I}} \psi}{\Gamma \vdash^{\mathcal{I}} \varphi \supset \psi} \supset R_1^{\mathcal{I}}$
By IH, have $\Gamma; \varphi \vdash \psi$.
By rule $\supset R$, we have $\Gamma \vdash \varphi \supset \psi$.

case : *Last rule is* $\quad \dfrac{\Gamma \vdash^{\mathcal{I}} \psi}{\Gamma \vdash^{\mathcal{I}} \varphi \supset \psi} \supset R_2^{\mathcal{I}}$
By IH, have $\Gamma \vdash \psi$.
by rule W, we have $\Gamma; \varphi \vdash \psi$.
by rule $\supset R$, we have $\Gamma \vdash \varphi \supset \psi$.

case : *Last rule is* $\quad \dfrac{\Gamma(\Delta_1; \Delta_2) \vdash^{\mathcal{I}} \varphi \quad (\Delta \in lubs_{\sqsubseteq}(\Delta_1)(\Delta_2)) \not\equiv \Delta_1; \Delta_2}{\Gamma(\Delta) \vdash^{\mathcal{I}} \varphi} \; C^{\mathcal{I}}$
By IH, we have $\Gamma(\Delta_1; \Delta_2) \vdash \varphi$
Since $\Delta_1 \sqsubseteq \Delta$ we can derive $\Gamma(\Delta; \Delta_2) \vdash \varphi$
Similarly, we can derive $\Gamma(\Delta; \Delta) \vdash \varphi$ then use rule C to get $\Gamma(\Delta) \vdash \varphi$.

case : *Last rule is* $\quad \dfrac{\Gamma(\varphi) \vdash^{\mathcal{I}} \chi \quad \Delta(\psi) \vdash^{\mathcal{I}} \chi \quad \Sigma(p) \in lubs_{\sqsubseteq}(\Gamma(p))(\Delta(p))}{\Sigma(\varphi \vee \psi) \vdash^{\mathcal{I}} \chi} \vee L^{\mathcal{I}}$
By IH, we have $\Gamma(\varphi) \vdash \chi$ and $\Delta(\psi) \vdash \chi$
Since we have $\Delta(p) \sqsubseteq \Sigma(p)$ and $\Gamma(p) \sqsubseteq \Sigma(p)$ we know $\Sigma(\varphi) \vdash \chi$ and $\Sigma(\psi) \vdash \chi$ so we can use rule $\vee L$ to get $\Sigma(\varphi \vee \psi) \vdash \chi$

case : *Last rule is* $\quad \dfrac{\Gamma(\varphi_1) \vdash^{\mathcal{I}} \chi}{\Gamma(\varphi_1 \wedge \varphi_2) \vdash^{\mathcal{I}} \chi} \wedge L_1^{\mathcal{I}}$
By IH, we have $\Gamma(\varphi_1) \vdash \chi$.
We can use rule W to get $\Gamma(\varphi_1; \varphi_2) \vdash \chi$ then rule $\wedge L$ to get $\Gamma(\varphi_1 \wedge \varphi_2) \vdash \chi$

We conclude the proof by observing that $\wedge L_2$ is parallel to $\wedge L_1$ and the rest of the rules are identical to the corresponding backward sequent calculus rules (E^I, $*$, and $-\!*$ rules)

Because our completeness proof will say that if $\Gamma \vdash \varphi$ then $\Gamma' \vdash^I \varphi$ such that $\Gamma' \sqsubseteq \Gamma$, we need a lemma about bunches that weaken to a split bunch (i.e. we need to be able to say something about the form of Γ when we know $\Gamma \sqsubseteq \Sigma(\Delta)$).

Lemma 1 (Weakening Split). *If $\Gamma \sqsubseteq \Sigma(\Delta)$ then either $\Gamma \sqsubseteq \Sigma(p)$ or else $\Gamma \equiv \Sigma'(\Delta')$ such that, $\Sigma'(p) \sqsubseteq \Sigma(p)$ and $\Delta' \sqsubseteq \Delta$ (where p is a new parameter).*

Proof (By Structural induction).

We will do induction on the derivation of $\Sigma(\Delta) \vdash \varphi$ from $\Gamma \vdash \varphi$. We may assume WLOG that the first rule is E. In the base case, $\Gamma \equiv \Sigma(\Delta)$ so the second case of the lemma holds.

If the last rule is rule E, then for any $\Sigma'(\Delta') \equiv \Sigma(\Delta)$ either case obviously carries through.

If the last rule is rule W, then we must consider the location of the use of rule W. The bunch weakened on (or removed, if we look at the reverse direction) must be either entirely within $\Sigma(p)$ and disjoint from Δ or else it must entirely contain Δ or be entirely contained in Δ (this can easily be seen by considering the tree structure of bunches). In the first case, either case of the IH carries through. In the second case, the entirety of Δ was simply weakened on and we could have just as easily weakened on p in its place, so the first case of the lemma holds. In the last case, the final step looks like:

$$\frac{\Sigma(\Sigma''(\Delta'')) \vdash \varphi}{\Sigma(\Delta) = \Sigma(\Sigma''(\Delta''; \Delta''')) \vdash \varphi} \; W$$

and since $\Sigma''(\Delta'') \sqsubseteq \Delta$, either case of the IH carries through.

Now we can prove the completeness of our calculus. We use a fairly straightforward structural induction on the derivation, the main complication is that we have to distinguish cases for the possible forms of $\Gamma \sqsubseteq \Sigma(\Delta)$ as given in the previous lemma.

Theorem 2 (Completeness). *If $\Gamma \vdash \varphi$ then $\Gamma^\circ \vdash^I \varphi$ such that, $\Gamma^\circ \sqsubseteq \Gamma$.*

Intuitively we think of Γ° as a bunch that we can weaken to get Γ (these are the types of sequents our inverse method will prove).

Proof (By structural induction).

case : *Derivation is* $\dfrac{}{\varphi \vdash \varphi}\ INIT$

We have immediately $\dfrac{}{\varphi \vdash^{I} \varphi}\ INIT^{I}$.

case : *Last rule is* $\dfrac{\Gamma(\Delta) \vdash \varphi}{\Gamma(\Delta; \Delta') \vdash \varphi}\ W$

By IH, have $(\Gamma(\Delta))^{\circ} \vdash^{I} \varphi$ *with* $(\Gamma(\Delta))^{\circ} \sqsubseteq \Gamma(\Delta)$ *and since* $\Delta \sqsubseteq \Delta; \Delta'$,
$(\Gamma(\Delta))^{\circ} \sqsubseteq \Gamma(\Delta; \Delta')$ *as required.*

case : *Last rule is* $\dfrac{\Gamma(\Delta; \Delta) \vdash \varphi}{\Gamma(\Delta) \vdash \varphi}\ C$

By IH, we have $(\Gamma(\Delta; \Delta))^{\circ} \vdash^{I} \varphi$
Either $(\Gamma(\Delta; \Delta))^{\circ} \sqsubseteq \Gamma(p)$ *or* $(\Gamma(\Delta; \Delta))^{\circ} \equiv \Gamma^{\circ}((\Delta; \Delta)^{\circ})$ *with* $(\Delta; \Delta)^{\circ} \sqsubseteq \Delta; \Delta$ *by previous lemma.*
In the first case, clearly if $(\Gamma(\Delta; \Delta))^{\circ} \sqsubseteq \Gamma(p)$ *then* $(\Gamma(\Delta; \Delta))^{\circ} \sqsubseteq \Gamma(\Delta)$, *so we are done.*
In the second case, either $(\Delta; \Delta)^{\circ} \sqsubseteq \Delta$ *or* $(\Delta; \Delta)^{\circ} \equiv \Delta_1; \Delta_2$ *such that* $\Delta_i \sqsubseteq \Delta(i = 1, 2)$.
In the first case we are done.
In the second case we have $\Delta_i \sqsubseteq \Delta(i = 1, 2)$ *so we have some* $\Sigma \in lubs_{\sqsubseteq} (\Delta_1)$ (Δ_2) *such that* $\Sigma \sqsubseteq \Delta$, *so we use rule* C^{I} *to get* $\Gamma^{\circ}(\Sigma) \vdash^{I} \varphi$.

case : *Last rule is* $\dfrac{\Gamma \vdash \varphi}{\Delta \vdash \varphi}\ (\Delta \equiv \Gamma)E$

Again, this is immediate from IH because $\Delta \equiv \Gamma$.

case : *Last rule is* $\dfrac{\Gamma \vdash \varphi \quad \Delta(\Delta'; \psi) \vdash \chi}{\Delta(\Gamma; \Delta'; \varphi \supset \psi) \vdash \chi}\ \supset L$

By IH have $\Gamma^{\circ} \vdash^{I} \varphi$ *and* $(\Delta(\Delta'; \psi))^{\circ} \vdash^{I} \chi$
Either $(\Delta(\Delta'; \psi))^{\circ} \sqsubseteq \Delta(p)$ *or* $(\Delta(\Delta'; \psi))^{\circ} \equiv \Delta^{\circ}((\Delta'; \psi)^{\circ})$ *such that* $\Delta^{\circ}(p) \sqsubseteq \Delta(p)$ *and* $(\Delta'; \psi)^{\circ} \sqsubseteq \Delta'; \psi$.
In the first case we are done.
In the second case, either $(\Delta'; \psi)^{\circ} \sqsubseteq \Delta'$ *or* $(\Delta'; \psi)^{\circ} \equiv \Delta'^{\circ}; \psi$ *such that* $\Delta'^{\circ} \sqsubseteq \Delta'$.
In the first case we are done.
In the second case we can apply rule $\supset L^{I}$ *to get* $\Delta^{\circ}(\Gamma^{\circ}; \Delta'^{\circ}; \varphi \supset \psi) \vdash^{I} \chi$.

case : *Last rule is* $\dfrac{\Gamma; \varphi \vdash \psi}{\Gamma \vdash \varphi \supset \psi}\ \supset R$

By IH have $(\Gamma; \varphi)^{\circ} \vdash \psi$.
Either $(\Gamma; \varphi)^{\circ} \equiv \Gamma^{\circ} \sqsubseteq \Gamma$ *or* $(\Gamma; \varphi)^{\circ} \equiv \Gamma^{\circ}; \varphi$ *such that* $\Gamma^{\circ} \sqsubseteq \Gamma$.
In the first case we use rule $\supset R_2^{I}$ *and in the second* $\supset R_1^{I}$ *to get* $\Gamma^{\circ} \vdash^{I} \varphi \supset \psi$

case : *Last rule is* $\dfrac{\Gamma(\varphi) \vdash \chi \quad \Gamma(\psi) \vdash \chi}{\Gamma(\varphi \vee \psi) \vdash \chi}\ \vee L$

By IH have $(\Gamma(\varphi))^{\circ} \vdash \chi$ *and* $(\Gamma(\psi))^{\circ}) \vdash \chi$.
Either $(\Gamma(\varphi))^{\circ} \sqsubseteq \Gamma(p)$ *or* $(\Gamma(\varphi))^{\circ} \equiv \Gamma_1^{\circ}(\varphi)$ *such that* $\Gamma_1^{\circ}(p) \sqsubseteq \Gamma(p)$.

In the first case we are done.
In the second case, either $(\Gamma(\psi))^\circ \sqsubseteq \Gamma(p)$ *or* $(\Gamma(\psi))^\circ \equiv \Gamma_2^\circ(\psi)$ *such that*
$\Gamma_2^\circ(p) \sqsubseteq \Gamma(p)$.
In the first case we are done.
In the second case, since $\Gamma_i^\circ(p) \sqsubseteq \Gamma(p)(i = 1, 2)$, *we have* $\Sigma^\circ(p) \in lubs_\sqsubseteq$
$(\Gamma_1^\circ(p)) (\Gamma_2^\circ(p))$ *such that* $\Sigma^\circ(\varphi \vee \psi) \sqsubseteq \Gamma(\varphi \vee \psi)$. *So we apply rule* $\vee L^I$ *and*
we are done.

case : *Last rule is*
$$\dfrac{\Gamma(\varphi; \psi) \vdash \chi}{\Gamma(\varphi \wedge \psi) \vdash \chi} \wedge L$$

By IH, we have $(\Gamma(\varphi; \psi))^\circ \vdash^I \chi$ *such that* $(\Gamma(\varphi; \psi))^\circ \sqsubseteq \Gamma(\varphi; \psi)$.
By the lemma, either $(\Gamma(\varphi; \psi))^\circ \sqsubseteq \Gamma(p)$ *or* $(\Gamma(\varphi; \psi))^\circ \equiv \Gamma^\circ((\varphi; \psi)^\circ)$ *with*
$\Gamma^\circ(p) \sqsubseteq \Gamma(p)$ *and* $(\varphi; \psi)^\circ \sqsubseteq (\varphi; \psi)$.
In the first case we are done.
In the second case we need to consider $(\varphi; \psi)^\circ$. *It cannot be empty or we would*
be in the first case.
If $(\varphi; \psi)^\circ \equiv \varphi$ *then we use rule* $\wedge L_1^I$ *to get* $\Gamma^\circ(\varphi \wedge \psi) \vdash^I \chi$.
If $(\varphi; \psi)^\circ \equiv \psi$ *then we use rule* $\wedge L_2^I$ *to get* $\Gamma^\circ(\varphi \wedge \psi) \vdash^I \chi$.
If $(\varphi; \psi)^\circ \equiv (\varphi; \psi)$ *then we use rule* $\wedge L_1^I$ *to get* $\Gamma^\circ(\varphi \wedge \psi; \psi) \vdash^I \chi$ *then rule*
$\wedge L_2^I$ *to get* $\Gamma^\circ(\varphi \wedge \psi; \varphi \wedge \psi) \vdash^I \chi$ *then rule* C^I *to get* $\Gamma^\circ(\varphi \wedge \psi) \vdash^I \chi$.

note : *The remaining cases are similar.*

3.2   An Example

Inverse method theorem proving in **BI** proceeds in the same way as in intuitionistic logic. Consider the (true) goal sequent $ø_m \vdash^I (p * (q \wedge r)) \twoheadrightarrow ((p \wedge q) * (p \wedge r))$. We start by enumerating the signed subformulas and identifying the initial sequents.

+ $(p * (q \wedge r)) \twoheadrightarrow ((p \wedge q) * (p \wedge r))$	+ $(p \wedge r)$
- $(p * (q \wedge r))$	- q
+ $((p \wedge q) * (p \wedge r))$	- r
- $(q \wedge r)$	+ p
- p	+ q
+ $(p \wedge q)$	+ r

From this we can see that the initial sequents we need are $p \vdash^I p$, $q \vdash^I q$ and $r \vdash^I r$. Theorem proving proceeds in rounds as follows (some unnecessary sequents are omitted).

$$
\begin{array}{lll}
1.\ p \vdash^{\mathtt{I}} p & \text{init} \\
2.\ q \vdash^{\mathtt{I}} q & \text{init} \\
3.\ r \vdash^{\mathtt{I}} r & \text{init} \\
\hline
4.\ p, q \vdash^{\mathtt{I}} p * q & *R^{\mathtt{I}}\ 1\ 2 \\
5.\ p, r \vdash^{\mathtt{I}} p * r & *R^{\mathtt{I}}\ 1\ 3 \\
6.\ q \wedge r \vdash^{\mathtt{I}} q & \wedge L_1^{\mathtt{I}}\ 2 \\
7.\ q \wedge r \vdash^{\mathtt{I}} r & \wedge L_2^{\mathtt{I}}\ 3 \\
\hline
8.\ (p, q); (p, r) \vdash^{\mathtt{I}} (p * q) \wedge (p * r) & \wedge R^{\mathtt{I}}\ 4\ 5 \\
9.\ p, (q \wedge r) \vdash^{\mathtt{I}} p * q & *R^{\mathtt{I}}\ 1\ 6 \\
\hline
10.\ p, (q; r) \vdash^{\mathtt{I}} (p * q) \wedge (p * r) & C^{\mathtt{I}}\ 8 \\
\hline
11.\ p, (q \wedge r) \vdash^{\mathtt{I}} (p * q) \wedge (p * r) & \wedge L^{\mathtt{I}}\ 10 \\
\hline
12.\ p * (q \wedge r) \vdash^{\mathtt{I}} (p * q) \wedge (p * r) & *R^{\mathtt{I}}\ 11 \\
\hline
13.\ \varnothing_m, (p * (q \wedge r)) \vdash^{\mathtt{I}} (p * q) \wedge (p * r) & E^{\mathtt{I}}\ 12 \\
\hline
14.\ \varnothing_m \vdash^{\mathtt{I}} (p * (q \wedge r)) \mathbin{-\!\!*} ((p * q) \wedge (p * r)) & \mathbin{-\!\!*} R^{\mathtt{I}}\ 13 \\
\end{array}
$$

3.3 An $\equiv$ Canonical Form for Bunches

While there is no canonical form which equates bunches modulo $\equiv$, weakening and contraction, there is a canonical form modulo $\equiv$ alone. Although this is fairly obvious, we have not seen it published anywhere. In [2], Armelín gives a similar canonical form which does not equate bunches modulo units as ours does. Use of this canonical form during proof-search lets us drop rule $E^{\mathtt{I}}$ altogether.

It is helpful, both in guiding an actual implementation and in understanding the structure of bunches, to have a canonical representative of $[\Gamma]_{\equiv}$ for any bunch Γ. To do so we define the following grammar.

$$
\begin{array}{rl}
\text{Bunches } \Gamma ::= & \varphi \mid \Pi \mid \Sigma \\
\text{Multiplicative Bunches } \Pi ::= & \varphi \mid \{\Sigma^*\}_m \\
\text{Additive Bunches } \Sigma ::= & \varphi \mid \{\Pi^*\}_a
\end{array}
$$

where $\{A^*\}$ denotes a multiset with elements from A.

We maintain the invariant that the multisets $\{A^*\}$ are never singletons (empty sets are fine, they are the units). If a subbunch which is supposed to be a multiset is a singleton, then we simply promote it in the tree and union it on to its parent. It is easy to see that φ's can always be promoted. Since the levels of the tree alternate between $\{\Sigma^*\}_m$ and $\{\Pi^*\}_a$, if e.g. $\{\Sigma^*\}_m = \{\Sigma\}_m$ and $\Sigma \neq \varphi$ then $\Sigma = \{\Pi^*\}_a$ so we can union that into the context of which $\{\Sigma^*\}_m$ was a member. We also maintain the invariant that a subbunch only appears once in any additive context, so we treat $\{\Pi^*\}_a$ as a set rather than a multiset.

This gives us an $\equiv$-canonical form. To see if two bunches are equivalent we convert to canonical form by these steps: first, flatten binary connectives into n-ary connectives (justified by associativity of , and ;), then forget about ordering by making them multiset operators (justified by commutativity of , and ;) then eliminate singletons by propagating them upwards. This last step is justified by the unit laws (we think of $\{\Sigma\}$ as $\Sigma, \varnothing_m$) which let us promote and the

associativity and commutativity laws which let us fold in (union) multisets that we promote. Lastly, we forget about the number of occurrences in the additive levels of the tree (justified by contraction for additive conjunction).

3.4   Implementation

We have implemented our inverse method for propositional **BI** without units in SML. We use the above canonical form for our bunches and generate proof-terms which are checked at the end.

Proof-terms for **BI** are terms of the $\alpha\lambda$-calculus [15]. We store proof-terms with each derived sequent in the database, so when we finish with a positive answer we have also a proof that the theorem is in fact true. We then check the proof in a straightforward way. This gives us a *certifying* theorem prover. Since the proof-checking code is much shorter and simpler than the proof-search code, we can have much higher confidence in a certified result than one that lacks proof-terms and checking.

In order to accommodate proof-terms in the inverse method, it is helpful to define a new intermediate proof-term `let` $x = e_1$ `in` e_2 which we convert to $[e_1/x]e_2$ before type-checking. This is used in elimination rules so we do not have to do proof substitution on-line.

At present, we only have a very simple prototype implementation without any of the customary optimizations applied in the inverse method. Nonetheless, we have found it useful for validating our ideas. In the conclusion we mention some planned improvements.

4   Related Work and Conclusions

Separation Logic [16] is a logic for reasoning about programs similar to Hoare Logic. Instead of standard intuitionistic logic, Separation Logic uses the connectives of **BI**, along with some other primitives, to express properties about data structures with shared mutable state. Therefore, automated theorem provers for **BI** are likely to eventually have practical uses in reasoning about programs. In particular, it is quite tedious to write out proofs of each inference step in Separation Logic and a good theorem prover for **BI** could go a long way towards automating the process of checking Separation Logic assertions.

There has been some work on a semantic tableau proof-search by Galmiche and Méry in **BI** [9,5] which has so far produced the BILL theorem prover for propositional **BI** without $\perp$, a later paper [8] extends this work to include $\perp$. Our work presents an alternative method for theorem proving in **BI**. We believe it is useful to investigate thoroughly both backward and forward search procedures for **BI** as work in other logics has shown that these methods have different properties and find different theorems easily.

There is also work on the inverse method in intuitionistic logic [18] and linear logic [17] which have resulted in inverse method provers for full first-order

intuitionistic and linear logics. Most of the improved strategies and optimizations used in these works (some described in the previous section) would most probably be applicable to inverse method theorem proving in **BI**.

Work by Armelín and Pym [3] develops a logic programming language, BLP, based on the hereditary Harrop fragment of **BI** with additive predication. Their work devlops a bottom-up proof search as its basis. By extending our inverse method to this fragment, it should be possible to develop an alternative, top-down basis for bunched logic programming.

In this paper, we have demonstrated that the inverse method is applicable to the core of propositional **BI**. Standard efficiency improvements [7] and the addition of units [6] should be relatively straightforward and lead to a theorem prover for full propositional **BI**. Additionally, formulation of a full first-order focusing prover for **BI** is likely to be fruitful for a number of reasons. Firstly, many investigations in proof-search, particularly an analysis of focusing [1] in **BI**, may lead to deeper understandings of its proof theory. And secondly, efficient provers for **BI** will likely become practically useful for program analysis as this is the logic that underlies Separation Logic.

5   Acknowledgments

References

1. Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):197–347, 1992.
2. Pablo Armelín. *Logic programming with bunched logic*. PhD thesis, University of London, 2002.
3. Pablo A. Armelín and David J. Pym. Bunched logic programming. In *IJCAR '01: Proceedings of the First International Joint Conference on Automated Reasoning*, pages 289–304. Springer-Verlag, 2001.
4. L. Bachmair and H. Ganzinger. Resolution theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 2, pages 19–100. North Holland, 2001.
5. Frederic Beal, Daniel Méry, and Didier Galmiche. Bill: A theorem prover for propositional bi logic.
6. Kaustuv Chaudhuri and Frank Pfenning. Resource management for the inverse method in linear logic. Carnegie Mellon University, Unpublished Maniscript, January 2003.
7. Anatoli Degtyarev and Andrei Voronkov. The inverse method. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, pages 179–272. Elsevier Science and MIT Press, 2001.
8. Didier Galmiche, Daniel Méry, and David J. Pym. Resource tableaux. In *CSL '02: Proceedings of the 16th International Workshop and 11th Annual Conference of the EACSL on Computer Science Logic*, pages 183–199. Springer-Verlag, 2002.

9. Didier Galmiche and Daniel Méry. Semantic labelled tableaux for propositional bi (without bottom). *Journal of Logic and Computation*, 13(5), October 2003.
10. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
11. James Harland and David Pym. Resource-distribution via boolean constraints. *ACM Trans. Comput. Logic*, 4(1):56–90, 2003.
12. S. Maslov. The inverse method of establishing deducibility in classical predicate calculus. *Soviet Mathematical Doklady*, 5:1420–1424, 1964.
13. P.W. O'Hearn and D. J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, June 1999.
14. Frank Pfenning. The inverse method. Carnegie Mellon University, Lecture Notes, Ch. 5, February 2004.
15. D.J. Pym. *The Semantics and Proof Theory of the Logic of the Logic of Bunched Implications*, volume 26 of *Applied Logic Series*. Kluwer Academic Publishers, 2002. Errata and Remarks maintained at:
    `http://www.cs.bath.ac.uk/~pym/BI-monograph-errata.pdf`.
16. John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science*, pages 55–74. IEEE Computer Society, 2002.
17. T. Tammet. Proof strategies in linear logic. *Journal of Automated Reasoning*, 12(3):273–304, 1994.
18. Tanel Tammet. A resolution theorem prover for intuitionistic logic. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings 13th Intl. Conf. on Automated Deduction, CADE'96, New Brunswick, NJ, USA, 30 July – 3 Aug 1996*, volume 1104, pages 2–16. Springer-Verlag, Berlin, 1996.