Under consideration for publication in Math. Struct. in Comp. Science

Proof Search in Lax Logic

Jacob M. Howe[†]

Computing Laboratory, University of Kent

Received 13 June 2000

A Gentzen sequent calculus for Lax Logic is presented, the proofs in which naturally correspond in a 1–1 way to the normal natural deductions for the logic. The propositional fragment of this calculus is used as the basis for another calculus, one which uses a history mechanism in order to give a decision procedure for propositional Lax Logic.

1. Introduction and Background

Proof search can be used with either of two meanings. It can either be used to mean the search for all proofs of a formula (proof enumeration), or to mean the search for a yes/no answer to a query (theorem proving). This paper describes two new sequent calculi for Lax Logic. One calculus is for proof enumeration for quantified Lax Logic, the other calculus is for theorem proving in propositional Lax Logic.

Lax Logic is an intuitionistic modal logic first introduced by Curry (Curry, 1952) to illustrate cut-elimination in the presence of modalities. The logic was rediscovered by Mendler, who developed the logic in the context of hardware verification to enable abstract verification of circuits (Mendler, 1993). The logic has a single modality ($\circ$, *somehow*) axiomatised by

$$S \supset \circ S, \qquad \circ\circ S \supset \circ S, \qquad (S \supset T) \supset (\circ S \supset \circ T)$$

The modality is unusual in having properties of both necessity and possibility. It can be thought of as expressing correctness up to a constraint, abstracting away from the detail (hence the choice of name, Lax Logic). A formula $\circ P$ can be read as "for some constraint c, P holds under c". The proof theory and semantics of Lax Logic, including Gentzen calculi, natural deduction calculi and Kripke semantics, are further developed in (Fairtlough and Mendler, 1997; Fairtlough and Walton, 1997; Benton et al., 1998).

The ability of Lax Logic to give an abstract expression of constraints has been utilised both in hardware verification and to give a proof theoretic semantics for constraint logic programming languages. In hardware verification, the timing constraints that need to be satisfied in a circuit can be abstracted away as instances of the modality and reasoned about separately from the logical analysis of the circuit (Mendler, 1993; Fairtlough and Mendler, 1994). In constraint logic programming, Lax Logic has been used to extend the view of logic programming as backwards proof search in constructive logics (Miller et al.,

[†] This work was partly supported by EPSRC grant GR/MO8769

1991). In essence, this approach takes normal natural deduction as the proof theoretic semantics for logic programming. Constraints can be abstracted away as modalities and the query can be reasoned about logically. The logic is used to give proofs of queries. In turn, these proofs give the constraints to be satisfied. The constraints can then be analysed separately (Fairtlough et al., 1997; Walton, 1998).

Natural deduction has a pragmatic drawback. In searching backwards for a proof of a formula, it is not always obvious which rule to apply. For example, in Intuitionistic Logic it is not obvious from the conclusion that rule ($\supset_\mathcal{E}$) should be applied. Even when the rule has been fixed, it is hard to determine the formulae in the premiss. Cut-free Gentzen sequent calculus systems (Gentzen, 1969) are much better from this point of view. When a principal formula has been chosen, the rules applicable are restricted. The application of logical rules is directed by the syntax of the principal formula. Structural rules can often be built into the sequent system. In such a system, when a principal formula has been chosen, the next rule application is exactly determined by the syntax of that formula.

There are well known translations (Prawitz, 1965) between normal natural deductions and sequent proofs. Therefore, one can search for proofs in sequent calculus systems and then translate the resulting proofs to normal natural deductions. The drawback is that many sequent proofs translate to the same normal natural deduction. Hence when one is trying to enumerate all proofs of a formula, the same proof is found many times. This gives one motivation for 'permutation-free' sequent calculi (introduced in (Herbelin, 1995) for Intuitionistic Logic). These are sequent calculi (enabling syntax directed proof search) whose proofs can be translated in a 1–1 way with the normal natural deductions for the logic. Permutation-free calculi have the advantages of a sequent calculus system, whilst reflecting the structure of normal natural deductions. The first calculus described in this paper, PFLAX, is a proof enumeration calculus for first-order quantified Lax Logic. PFLAX is a permutation-free calculus for Lax Logic – the sequent proofs naturally correspond in a 1–1 way to the normal natural deductions.

Propositional logics are usually decidable and therefore it is desirable to find effective decision procedures for such logics. Here, by studying the nature of non-terminating backwards search to see where one can stop the search, a decision procedure for propositional Lax Logic is given; this theorem proving calculus is called PFLAX^{Hist}. The calculus uses a technique for detecting loops using a history mechanism, building on work of Heuerding *et al* (Heuerding et al., 1996; Heuerding, 1998; Howe, 1997). It uses the propositional fragment of PFLAX as the base calculus to which a history mechanism is added, giving the decision procedure. The technique is general and may be applied to many other propositional logics. We know of no other decision procedure for propositional Lax Logic.

2. Natural Deduction

This section gives the relevant material on natural deduction needed to develop the permutation-free calculus for Lax Logic (the proofs in which correspond in a 1–1 way to normal natural deductions). A natural deduction calculus for Lax Logic (with rules for quantifiers and falsum added) taken from (Benton et al., 1998), can be seen in Figure 1.

$$\frac{}{\Gamma, P \vdash P}\ (ax) \qquad \frac{}{\Gamma \vdash \top}\ (\top_{\mathcal{I}}) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash P}\ (\bot_{\varepsilon})$$

$$\frac{\Gamma, P \vdash Q}{\Gamma \vdash P \supset Q}\ (\supset_{\mathcal{I}}) \qquad \frac{\Gamma \vdash P \supset Q \quad \Gamma \vdash P}{\Gamma \vdash Q}\ (\supset_{\varepsilon})$$

$$\frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\ (\wedge_{\mathcal{I}}) \qquad \frac{\Gamma \vdash P \wedge Q}{\Gamma \vdash P}\ (\wedge_{\varepsilon_1}) \qquad \frac{\Gamma \vdash P \wedge Q}{\Gamma \vdash Q}\ (\wedge_{\varepsilon_2})$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\ (\vee_{\mathcal{I}_1}) \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\ (\vee_{\mathcal{I}_2}) \qquad \frac{\Gamma \vdash P \vee Q \quad \Gamma, P \vdash R \quad \Gamma, Q \vdash R}{\Gamma \vdash R}\ (\vee_{\varepsilon})$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash \circ P}\ (\circ_{\mathcal{I}}) \qquad \frac{\Gamma \vdash \circ P \quad \Gamma, P \vdash \circ Q}{\Gamma \vdash \circ Q}\ (\circ_{\varepsilon})$$

$$\frac{\Gamma \vdash P[u/x]}{\Gamma \vdash \forall x P}\ (\forall_{\mathcal{I}})\dagger \qquad \frac{\Gamma \vdash \forall x P}{\Gamma \vdash P[t/x]}\ (\forall_{\varepsilon}) \qquad \frac{\Gamma \vdash P[t/x]}{\Gamma \vdash \exists x P}\ (\exists_{\mathcal{I}}) \qquad \frac{\Gamma \vdash \exists x P \quad \Gamma, P[u/x] \vdash R}{\Gamma \vdash R}\ (\exists_{\varepsilon})\dagger$$

† u not free in Γ

Fig. 1. Sequent style presentation of natural deduction for Lax Logic.

Normal natural deductions are the objects of interest. The β-reduction and commuting conversion steps of normalisation are given in (Benton et al., 1998). The extra cases for $\bot_{\varepsilon}$ and $\exists_{\varepsilon}$ can be added, and are to be found in (Howe, 1998; Howe, 1999).

Definition 1 *A natural deduction is said to be in β, c-**normal form** when no β-reductions and no commuting conversions are applicable.*

We present a restricted version of natural deduction for Lax Logic. In this calculus, the only deductions possible are in β, c-normal form. This calculus has two kinds of 'sequent', differentiated by their consequence relations, $\rhd$ and $\rhd\!\!\rhd$. Rules are applicable only when the premisses have the appropriate consequence relation. The conclusions have a fixed consequence relation. Thus valid deductions are of a restricted form. This calculus, which we call NLAX, is given (with the proof terms given in the next section) in Figure 2.

Proposition 1 *The calculus* NLAX *only allows deductions to which no β-reductions and no commuting conversions are applicable. Moreover, it allows all β, c-normal deductions.*

2.1. *Term Assignment*

We give a proof term system for NLAX. The term system is needed to prove the results given in section 4. In (Moggi, 1989) Moggi gave a λ-calculus, which he called the *computational λ-calculus*. As is shown in (Benton et al., 1998), this calculus naturally matches Lax Logic. More about the computational λ-calculus and Lax Logic (there called computational logic) can be found in (Benton et al., 1998).

Proof terms for unrestricted natural deduction for Lax Logic can be found in (Howe, 1998; Moggi, 1989). We are interested in the 'real' proofs for Lax Logic – the normal natural deductions. We restrict the terms that can be built, in order that they match our restricted natural deduction calculus NLAX, giving proof objects. The proof terms

$$\dfrac{}{\Gamma, x : P \triangleright var(x) : P}\,(ax) \qquad \dfrac{\Gamma \triangleright A : P}{\Gamma \triangleright\!\triangleright an(A) : P}\,(M) \qquad \dfrac{}{\Gamma \triangleright\!\triangleright * : \top}\,(\top_{\mathcal{I}}) \qquad \dfrac{\Gamma \triangleright A : \bot}{\Gamma \triangleright\!\triangleright efq(A) : P}\,(\bot_{\varepsilon})$$

$$\dfrac{\Gamma, x : P \triangleright\!\triangleright N : Q}{\Gamma \triangleright\!\triangleright \lambda x.N : P \supset Q}\,(\supset_{\mathcal{I}}) \qquad \dfrac{\Gamma \triangleright A : P \supset Q \quad \Gamma \triangleright\!\triangleright N : P}{\Gamma \triangleright ap(A, N) : Q}\,(\supset_{\varepsilon})$$

$$\dfrac{\Gamma \triangleright\!\triangleright N_1 : P \quad \Gamma \triangleright\!\triangleright N_2 : Q}{\Gamma \triangleright\!\triangleright pr(N_1, N_2) : P \wedge Q}\,(\wedge_{\mathcal{I}}) \qquad \dfrac{\Gamma \triangleright A : P \wedge Q}{\Gamma \triangleright fst(A) : P}\,(\wedge_{\varepsilon_1}) \qquad \dfrac{\Gamma \triangleright A : P \wedge Q}{\Gamma \triangleright snd(A) : Q}\,(\wedge_{\varepsilon_2})$$

$$\dfrac{\Gamma \triangleright\!\triangleright N : P}{\Gamma \triangleright\!\triangleright i(N) : P \vee Q}\,(\vee_{\mathcal{I}_1}) \qquad \dfrac{\Gamma \triangleright\!\triangleright N : Q}{\Gamma \triangleright\!\triangleright j(N) : P \vee Q}\,(\vee_{\mathcal{I}_2})$$

$$\dfrac{\Gamma \triangleright A : P \vee Q \quad \Gamma, x_1 : P \triangleright\!\triangleright N_1 : R \quad \Gamma, x_2 : Q \triangleright\!\triangleright N_2 : R}{\Gamma \triangleright\!\triangleright wn(A, x_1.N_1, x_2.N_2) : R}\,(\vee_{\varepsilon})$$

$$\dfrac{\Gamma \triangleright\!\triangleright N : P}{\Gamma \triangleright\!\triangleright smhi(N) : \circ P}\,(\circ_{\mathcal{I}}) \qquad \dfrac{\Gamma \triangleright A : \circ P \quad \Gamma, x : P \triangleright\!\triangleright N : \circ Q}{\Gamma \triangleright\!\triangleright smhe(A, x.N) : \circ Q}\,(\circ_{\varepsilon})$$

$$\dfrac{\Gamma \triangleright\!\triangleright N : P[u/x]}{\Gamma \triangleright\!\triangleright \lambda u.N : \forall x P}\,(\forall_{\mathcal{I}})\dagger \qquad \dfrac{\Gamma \triangleright A : \forall x P}{\Gamma \triangleright apn(A, t) : P[t/x]}\,(\forall_{\varepsilon})$$

$$\dfrac{\Gamma \triangleright\!\triangleright N : P[t/x]}{\Gamma \triangleright\!\triangleright prq(t, N) : \exists x P}\,(\exists_{\mathcal{I}}) \qquad \dfrac{\Gamma \triangleright A : \exists x P \quad \Gamma, y : P[u/x] \triangleright\!\triangleright N : R}{\Gamma \triangleright\!\triangleright ee(A, u.y.N) : R}\,(\exists_{\varepsilon})\dagger$$

† u not free in Γ

Fig. 2. NLAX with proof annotations.

come in two syntactic categories, **A** and **N**. V is the category of variables (proofs), U is the category of variables (individuals in formulae), and T the category of terms. The proof terms are given with an abstract syntax, with the notation chosen to be suggestive of the associated proof rules. Hence $smhi(N)$ for the term associated with the somehow introduction rule. The extra constructor $an(A)$ matches the (M) rule of NLAX.

$$\begin{array}{rl} A ::= & var(V) \mid ap(A, N) \mid fst(A) \mid snd(A) \mid apn(A, T) \\ N ::= & * \mid efq(A) \mid an(A) \mid \lambda V.N \mid pr(N, N) \mid i(N) \mid j(N) \mid wn(A, V.N, V.N) \\ & smhi(N) \mid smhe(A, V.N) \mid \lambda U.N \mid prq(T, N) \mid ee(A, U.V.N) \end{array}$$

NLAX together with proof annotations for normal terms can be seen in Figure 2.

3. Sequent Calculus

In this section we present a new Gentzen sequent calculus for Lax Logic, PFLAX. The proofs allowed by PFLAX naturally correspond in a 1–1 way to normal natural deductions for Lax Logic – i.e. the proofs of NLAX. In Figure 3 we remind the reader of the sequent calculus, extending those in (Fairtlough and Mendler, 1997; Benton et al., 1998) to quantifiers.

We give a new sequent calculus, PFLAX ('permutation-free' Lax Logic). PFLAX extends the permutation-free calculus MJ for Intuitionistic Logic (Herbelin, 1995; Dyckhoff and Pinto, 1998; Dyckhoff and Pinto, 1999) to a calculus for Lax Logic. Like MJ this calculus has two forms of judgment, $\Gamma \Rightarrow R$ and $\Gamma \xrightarrow{Q} R$. The first looks like the usual kind of sequent; however, only right rules and contraction are applicable to this kind of

$$\frac{}{\Gamma, P \Rightarrow P}\ (ax) \qquad \frac{\Gamma, P, P \Rightarrow R}{\Gamma, P \Rightarrow R}\ (C) \qquad \frac{}{\Gamma \Rightarrow \top}\ (\top_{\mathcal{R}}) \qquad \frac{}{\Gamma, \bot \Rightarrow P}\ (\bot_{\mathcal{L}})$$

$$\frac{\Gamma, P \Rightarrow Q}{\Gamma \Rightarrow P \supset Q}\ (\supset_{\mathcal{R}}) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, Q \Rightarrow R}{\Gamma, P \supset Q \Rightarrow R}\ (\supset_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \Rightarrow Q}{\Gamma \Rightarrow P \wedge Q}\ (\wedge_{\mathcal{R}}) \qquad \frac{\Gamma, P \Rightarrow R}{\Gamma, P \wedge Q \Rightarrow R}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma, Q \Rightarrow R}{\Gamma, P \wedge Q \Rightarrow R}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_1}) \qquad \frac{\Gamma \Rightarrow Q}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_2}) \qquad \frac{\Gamma, P \Rightarrow R \quad \Gamma, Q \Rightarrow R}{\Gamma, P \vee Q \Rightarrow R}\ (\vee_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow \circ P}\ (\circ_{\mathcal{R}}) \qquad \frac{\Gamma, P \Rightarrow \circ R}{\Gamma, \circ P \Rightarrow \circ R}\ (\circ_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P[u/x]}{\Gamma \Rightarrow \forall x P}\ (\forall_{\mathcal{R}})\dagger \qquad \frac{\Gamma, P[t/x] \Rightarrow R}{\Gamma, \forall x P \Rightarrow R}\ (\forall_{\mathcal{L}}) \qquad \frac{\Gamma \Rightarrow P[t/x]}{\Gamma \Rightarrow \exists x P}\ (\exists_{\mathcal{R}}) \qquad \frac{\Gamma, P[u/x] \Rightarrow R}{\Gamma, \exists x P \Rightarrow R}\ (\exists_{\mathcal{L}})\dagger$$

† u not free in Γ

Fig. 3. Sequent calculus for Lax Logic.

sequent in backwards proof search. The second kind of sequent has a formula (on the left) in a privileged position called the *stoup*, following (Girard, 1991). The formula in the stoup is always principal in the conclusion of an inference rule. The stoup is a form of focusing; proof search is restricted so that, whenever possible, the active formulae of an inference are principal in the premiss. In backwards proof search, left rules are only applicable to stoup sequents. PFLAX (together with the proof terms given in the next section) is displayed in Figure 4. We give a simple example of a derivation in PFLAX:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{B, B, \circ B \wedge (B \supset A) \xrightarrow{B} B}\ (ax)}{B, B, \circ B \wedge (B \supset A) \Rightarrow B}\ (C) \qquad \dfrac{}{B, B, \circ B \wedge (B \supset A) \xrightarrow{A} A}\ (ax)}{B, B, \circ B \wedge (B \supset A) \xrightarrow{B \supset A} A}\ (\supset_{\mathcal{L}})}{B, B, \circ B \wedge (B \supset A) \xrightarrow{\circ B \wedge (B \supset A)} A}\ (\wedge_{\mathcal{L}_2})}{B, B, \circ B \wedge (B \supset A) \Rightarrow A}\ (C)}{B, B, \circ B \wedge (B \supset A) \Rightarrow \circ A}\ (\circ_{\mathcal{R}})}{B, \circ B \wedge (B \supset A) \xrightarrow{\circ B} \circ A}\ (\circ_{\mathcal{L}})}{B, \circ B \wedge (B \supset A) \xrightarrow{\circ B \wedge (B \supset A)} \circ A}\ (\wedge_{\mathcal{L}_1})}{B, \circ B \wedge (B \supset A) \Rightarrow \circ A}\ (C)$$

3.1. *Term Assignment for Sequent Calculus*

We give a term assignment system for PFLAX. The term system is a simple extension of that given in (Herbelin, 1995; Dyckhoff and Pinto, 1996; Dyckhoff and Pinto, 1998). The term calculus has two syntactic categories, **M** and **Ms**. V is the category of variables (proofs), U is the category of variables (individuals) and T is the category of terms. The proof terms are given with an abstract syntax suggestive of the associated proof rules.

$$\frac{}{\Gamma \xrightarrow{P} [\,] : P}\ (ax) \qquad \frac{\Gamma, x : P \xrightarrow{P} Ms : R}{\Gamma, x : P \Rightarrow (x; Ms) : R}\ (C) \qquad \frac{}{\Gamma \Rightarrow * : \top}\ (\top_{\mathcal{R}}) \qquad \frac{}{\Gamma \xrightarrow{\perp} ae : \perp}\ (\perp_{\mathcal{L}})$$

$$\frac{\Gamma, x : P \Rightarrow M : Q}{\Gamma \Rightarrow \lambda x.M : P \supset Q}\ (\supset_{\mathcal{R}}) \qquad \frac{\Gamma \Rightarrow M : P \quad \Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \supset Q} (M :: Ms) : R}\ (\supset_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M_1 : P \quad \Gamma \Rightarrow M_2 : Q}{\Gamma \Rightarrow pair(M_1, M_2) : P \wedge Q}\ (\wedge_{\mathcal{R}}) \qquad \frac{\Gamma \xrightarrow{P} Ms : R}{\Gamma \xrightarrow{P \wedge Q} p(Ms) : R}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \wedge Q} q(Ms) : R}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow M : P}{\Gamma \Rightarrow inl(M) : P \vee Q}\ (\vee_{\mathcal{R}_1}) \qquad \frac{\Gamma \Rightarrow M : Q}{\Gamma \Rightarrow inr(M) : P \vee Q}\ (\vee_{\mathcal{R}_2})$$

$$\frac{\Gamma, x_1 : P \Rightarrow M_1 : R \quad \Gamma, x_2 : Q \Rightarrow M_2 : R}{\Gamma \xrightarrow{P \vee Q} when(x_1.M_1, x_2.M_2) : R}\ (\vee_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P}{\Gamma \Rightarrow smhr(M) : \circ P}\ (\circ_{\mathcal{R}}) \qquad \frac{\Gamma, x : P \Rightarrow M : \circ R}{\Gamma \xrightarrow{\circ P} smhl(x.M) : \circ R}\ (\circ_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P[u/x]}{\Gamma \Rightarrow \lambda u.M : \forall x P}\ (\forall_{\mathcal{R}})\dagger \qquad \frac{\Gamma \xrightarrow{P[t/x]} Ms : R}{\Gamma \xrightarrow{\forall x P} apq(t, Ms) : R}\ (\forall_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P[t/x]}{\Gamma \Rightarrow pairq(t, M) : \exists x P}\ (\exists_{\mathcal{R}}) \qquad \frac{\Gamma, y : P[u/x] \Rightarrow M : R}{\Gamma \xrightarrow{\exists x P} spl(u.y.M) : R}\ (\exists_{\mathcal{L}})\dagger$$

† u not free in Γ

Fig. 4. The sequent calculus PFLAX, with proof annotations.

In particular, $[\,]$ is used for the axiom term and $(M :: Ms)$ for the term associated with implication on the left, giving a list of M terms suggestive of the ordering of rules in the calculus.

$M ::= * \mid (V; Ms) \mid \lambda V.M \mid pair(M, M) \mid inl(M) \mid inr(M) \mid smhr(M) \mid \lambda U.M \mid pairq(T, M)$
$Ms ::= [\,] \mid ae \mid (M :: Ms) \mid p(Ms) \mid q(Ms) \mid when(V.M, V.M)$
$\quad smhl(V.M) \mid apq(T, Ms) \mid spl(U.V.M)$

These terms can easily be typed by PFLAX, as seen in Figure 4.

4. Equivalence of the Calculi

We prove the equivalence of the term calculi and the soundness and adequacy of PFLAX. These results prove the desired correspondence. The proofs are extensions of those for the MJ calculus for Intuitionistic Logic (Dyckhoff and Pinto, 1998), hence most detail is omitted. We start by giving pairs of functions that define translations between the term assignment systems for natural deduction (NLAX) and sequent calculus (PFLAX), extending to Lax Logic those of (Herbelin, 1995; Dyckhoff and Pinto, 1998) for Intuitionistic Logic.

Sequent Calculus → Natural Deduction:

$\theta : \mathbf{M} \to \mathbf{N}$	$\theta' : \mathbf{A} \times \mathbf{Ms} \to \mathbf{N}$
$\theta(x; Ms) = \theta'(var(x), Ms)$	$\theta'(A, [\,]) = an(A)$
$\theta(\lambda x.M) = \lambda x.\theta(M)$	$\theta'(A, (M :: Ms)) = \theta'(ap(A, \theta(M)), Ms)$
$\theta(smhr(M)) = smhi(\theta(M))$	$\theta'(A, smhl(x.Ms)) = smhe(A, x.\theta(M))$
$\theta(*) = *$	$\theta'(A, ae) = efq(A)$
$\theta(pair(M_1, M_2)) = pr(\theta(M_1), \theta(M_2))$	$\theta'(A, p(Ms)) = \theta'(fst(A), Ms)$
$\theta(inl(M)) = i(\theta(M))$	$\theta'(A, q(Ms)) = \theta'(snd(A), Ms)$
$\theta(inr(M)) = j(\theta(M))$	$\theta'(A, apq(t, Ms)) = \theta'(apn(A, t), Ms)$
$\theta(\lambda u.M) = \lambda u.\theta(M)$	$\theta'(A, spl(u.y.M)) = ee(A, u.y.\theta(M))$
$\theta(pairq(t, M)) = prq(t, \theta(M))$	

Natural Deduction → Sequent Calculus:

$\psi : \mathbf{N} \to \mathbf{M}$	$\psi' : \mathbf{A} \times \mathbf{Ms} \to \mathbf{M}$
$\psi(an(A)) = \psi'(A, [\,])$	$\psi'(var(x), Ms) = (x; Ms)$
$\psi(\lambda x.N) = \lambda x.\psi(N)$	$\psi'(ap(A, N), Ms) = \psi'(A, (\psi(N) :: Ms))$
$\psi(smhe(A, x.N)) = \psi'(A, smhl(x.\psi(N)))$	$\psi'(fst(A), Ms) = \psi'(A, p(Ms))$
$\psi(smhi(N)) = smhr(\psi(N))$	$\psi'(snd(A), Ms) = \psi'(A, q(Ms))$
$\psi(*) = *$	$\psi'(apn(A, t), Ms) = \psi'(A, apq(t, Ms))$
$\psi(efq(A)) = \psi'(A, ae)$	
$\psi(pr(N_1, N_2)) = pair(\psi(N_1), \psi(N_2))$	
$\psi(i(N)) = inl(\psi(N))$	
$\psi(j(N)) = inr(\psi(N))$	
$\psi(wn(A, x_1.N_1, x_2.N_2)) =$ $\psi'(A, when(x_1.\psi(N_1), x_2.\psi(N_2)))$	
$\psi(\lambda u.N) = \lambda u.\psi(N)$	
$\psi(prq(t, N)) = pairq(t, \psi(N))$	
$\psi(ee(A, u.y.N)) = \psi'(A, spl(u.y.\psi(N)))$	

We give two lemmas demonstrating the equivalence of the term calculi, that is, the translations from one system to the other are 1–1.

Lemma 1 **i)** $\psi(\theta(M)) = M$; **ii)** $\psi(\theta'(A, Ms)) = \psi'(A, Ms)$.

Proof. By simultaneous induction on the structure of M and Ms. For full details see (Howe, 1999). □

Lemma 2 **i)** $\theta(\psi(N)) = N$; **ii)** $\theta(\psi'(A, Ms)) = \theta'(A, Ms)$.

Proof. By simultaneous induction on the structure of N and A. For full details see (Howe, 1999). □

The following two theorems state soundness and adequacy. They show that the translations respect provability, that is, no 'sequent' (and hence its associated term) can be proved in one system, but not its translation in the other.

Theorem 1 (SOUNDNESS) *The following rules are admissible:*

$$\frac{\Gamma \Rightarrow M : R}{\Gamma \rhd\!\!\rhd \theta(M) : R}\ i) \qquad \frac{\Gamma \rhd A : P \quad \Gamma \xrightarrow{P} Ms : R}{\Gamma \rhd\!\!\rhd \theta'(A, Ms) : R}\ ii)$$

Proof. By simultaneous induction on the structure of M and Ms. For full details see (Howe, 1999). □

Theorem 2 (ADEQUACY) *The following rules are admissible:*

$$\frac{\Gamma \rhd\!\!\rhd N : R}{\Gamma \Rightarrow \psi(N) : R}\ i) \qquad \frac{\Gamma \rhd A : P \quad \Gamma \xrightarrow{P} Ms : R}{\Gamma \Rightarrow \psi'(A, Ms) : R}\ ii)$$

Proof. By simultaneous induction on the structure of A and N. For full details see (Howe, 1999). □

Since the term systems are in 1–1 correspondence (lemma 1 and lemma 2) and the translations preserve provability (theorem 1 and theorem 2), the 1–1 correspondence between PFLAX and NLAX has been established. This is stated in the following theorem.

Theorem 3 *The normal natural deductions of Lax Logic (the proofs of* NLAX*) are in 1–1 correspondence to the proofs of* PFLAX.

An immediate corollary of theorem 3 is that PFLAX is sound and complete with respect to natural deduction for Lax Logic. Quantified Lax Logic is demonstrated to be sound and complete with respect to certain classes of Kripke model-structures in (Fairtlough and Walton, 1997).

4.1. *Cut Elimination*

We now briefly discuss cut for PFLAX. In the usual sequent calculus, cut may be formulated as follows:

$$\frac{\Gamma \Rightarrow P \quad \Gamma, P \Rightarrow Q}{\Gamma \Rightarrow Q}\ (cut)$$

In PFLAX, the two judgment forms lead to the following four cut rules (as for Intuitionistic Logic in (Herbelin, 1995; Dyckhoff and Pinto, 1998)):

$$\frac{\Gamma \xrightarrow{Q} P \quad \Gamma \xrightarrow{P} R}{\Gamma \xrightarrow{Q} R}\ (cut_1) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, P \xrightarrow{Q} R}{\Gamma \xrightarrow{Q} R}\ (cut_2)$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \xrightarrow{P} R}{\Gamma \Rightarrow R}\ (cut_3) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, P \Rightarrow R}{\Gamma \Rightarrow R}\ (cut_4)$$

We call PFLAX extended with the four cut rules PFLAX^{cut}. We can give reduction rules for PFLAX^{cut} and prove the weak cut elimination theorem for the logic. We can also prove strong normalisation for the term system associated with the logic, hence strong cut-elimination. Details and proofs (extending those for Intuitionistic Logic in (Herbelin, 1995; Dyckhoff and Pinto, 1998)) can be found in (Howe, 1998).

Theorem 4 *The rules* $(cut_1), (cut_2), (cut_3), (cut_4)$ *are admissible in* PFLAX.

Theorem 5 *The cut reduction system for* PFLAX *strongly normalises.*

5. Deciding Lax Logic

It is useful and interesting to have a decision procedure for any logic. This section describes a decision procedure for propositional Lax Logic. To the best of our knowledge, no decision procedure for propositional Lax Logic has been presented before.

The new calculus uses a history mechanism to ensure termination of backwards proof search. History mechanisms were introduced in (Heuerding et al., 1996; Heuerding, 1998). The refined history mechanism used here can be found in (Howe, 1997; Howe, 1998).

Another approach to deciding propositional logics is by the use of 'contraction-free' sequent calculi, such as the one for propositional Intuitionistic Logic given in (Dyckhoff, 1992; Hudelmaier, 1993). If such a decision procedure for Lax Logic could be found, we would expect it to be faster than one involving a history mechanism. An investigation of contraction-free calculi for Lax Logic can be found in (Avellone and Ferrari, 1996). Unfortunately, this investigation did not succeed in finding a contraction-free calculus. We believe that a contraction-free calculus for Lax Logic cannot be found, as (for arbitrary n) examples can be constructed which require an entire formula in a sequent to be contracted n times in a proof. As an example, consider the sequent $B \supset (\circ A \supset C) \supset \circ A, \circ B, \circ A \supset C \Rightarrow C$, where $\circ A \supset C$ needs to duplicated in its entirety in order to prove the sequent.

5.1. *Deciding Propositional Logics Using History Mechanisms*

One approach to finding a decision procedure for a propositional logic is to place conditions on the sequent calculus to ensure termination of search. It is elegant to be able to build the content of these conditions into the sequent calculus itself. This is how the calculus for theorem proving in this section is developed.

In order to ensure termination of backward proof search, we need to check that the same sequent (modulo number of occurrences of identical formulae) does not appear again on a branch, that is, proof search does not loop. Avoiding loops can also prevent the unnecessary computation arising from a finite number of passes through a loop in a successful derivation. We need a mechanical way to detect such loops. One way to do this is to add a *history* to a sequent. The history is the set of all sequents to have occurred so far on a branch of a proof tree. After each backwards inference the new sequent (without its history) is checked to see whether it is a member of this set. If it is we have looping and backtrack. If not, the new history is the extension of the old history by the old sequent (without the history component), and we try to prove the new sequent, and so on. Unfortunately, this method is space inefficient as it requires long lists of sequents to be stored by the computer, and all of this list has to be checked at each stage. When the sequents are stored, far more information than necessary is kept. Efficiency would be improved by cutting down the amount of storage and checking needed to prevent looping.

The basis of the reduced history is the realisation, as in (Heuerding et al., 1996), that one need only store goal formulae (a goal formula is the succedent of a sequent) in order to loop-check. In the calculi dealt with in this paper, once a formula is in the context it will be in the context of all sequents above it in the proof tree. We say that the calculus has *increasing context*. For two sequents to be the same they need to have the same context (up to multiple occurrences of formulae). Therefore we may empty the history every time the context is (properly) extended. All we need store in the history are goal formulae. If we come to a sequent whose goal is already in the history, then it has the same goal and the same context as another sequent – there is a loop.

There are two slightly different approaches capturing this. There is the straightforward extension of the calculus described in (Heuerding et al., 1996), which we call the 'Swiss history'; more on this loop-checking method can be found in (Heuerding, 1998). There is also related work on histories for Intuitionistic Logic in (Gabbay, 1991). Another approach involves storing slightly more formulae in the history, but which for some calculi detects loops more quickly. This we describe as the 'Scottish history' (Howe, 1997); it can, in many cases, be more efficient than the Swiss method. In this paper we give a calculus for Lax Logic using the Scottish history as we believe this to be the better method for intuitionistic logics (Howe, 1997).

The generality of this approach is attractive. The history mechanism can be attached to many calculi to give decision procedures; applications can be found in (Howe, 1998).

5.2. PFLAX^{Hist}

This section describes a history calculus for propositional Lax Logic. The calculus is an extension of that for Intuitionistic Logic given in (Howe, 1997). The modality is handled similarly to disjunction (disjunction is not covered in (Heuerding et al., 1996), and requires special treatment). It uses the calculus PFLAX as a base on which to build the calculus as this calculus has already reduced the search space to a certain extent. PFLAX has the increasing context required for the application of the history mechanism. However, a more usual formulation could have been used instead. PFLAX^{Hist} can be seen in Figure 5. Observe the two rules for $(\supset_{\mathcal{R}})$. These correspond to the two cases where the new formula is or is not in the context. As noted above, this is very important for history mechanisms. Notice that the number of formulae in the history is at most equal to the length of the formula we check for provability.

A sequent is matched against the conclusions of right rules until the goal formula is either a propositional variable, falsum, disjunction or a $\circ$ formula This has been ensured by the restriction on goal formulae given in the calculus (note that the rules for disjunction on the right and somehow on the right are only possible on backtracking, or with an empty context). A formula from the context is then selected using the rule (C) and matched against the left rules of the calculus. The Scottish calculus keeps (as a set) a complete record of goal formulae between context extensions. At each of the places where the history might be extended, the new goal is checked against the history. If it is in the history, then there is a loop, hence failure and backtracking.

There are other places where the rules are restricted to prevent looping. Where nec-

$$\dfrac{}{\Gamma \xrightarrow{P} P;\mathcal{H}}\ (ax) \qquad \dfrac{\Gamma, P \xrightarrow{P} D;\mathcal{H}}{\Gamma, P \Rightarrow D;\mathcal{H}}\ (C) \qquad \dfrac{}{\Gamma \Rightarrow \top;\mathcal{H}}\ (\top_{\mathcal{R}}) \qquad \dfrac{}{\Gamma \xrightarrow{\bot} D;\mathcal{H}}\ (\bot_{\mathcal{L}})$$

$$\dfrac{\Gamma, P \Rightarrow Q;\{Q\}}{\Gamma \Rightarrow P \supset Q;\mathcal{H}}\ (\supset_{\mathcal{R}1}) \quad \text{if } P \notin \Gamma \qquad \dfrac{\Gamma \Rightarrow Q;(Q,\mathcal{H})}{\Gamma \Rightarrow P \supset Q;\mathcal{H}}\ (\supset_{\mathcal{R}2}) \quad \text{if } P \in \Gamma \text{ and } Q \notin \mathcal{H}$$

$$\dfrac{\Gamma, P \Rightarrow \bot;\{\bot\}}{\Gamma \Rightarrow \neg P;\mathcal{H}}\ (\neg_{\mathcal{R}1}) \quad \text{if } P \notin \Gamma \qquad \dfrac{\Gamma \Rightarrow \bot;(\bot,\mathcal{H})}{\Gamma \Rightarrow \neg P;\mathcal{H}}\ (\neg_{\mathcal{R}2}) \quad \text{if } P \in \Gamma \text{ and } \bot \notin \mathcal{H}$$

$$\dfrac{\Gamma \Rightarrow P;(P,\mathcal{H}) \quad \Gamma \xrightarrow{Q} D;\mathcal{H}}{\Gamma \xrightarrow{P \supset Q} D : \mathcal{H}}\ (\supset_{\mathcal{L}}) \quad \text{if } P \notin \mathcal{H} \qquad \dfrac{\Gamma \Rightarrow P;(P,\mathcal{H})}{\Gamma \xrightarrow{\neg P} D;\mathcal{H}}\ (\neg_{\mathcal{L}}) \quad \text{if } P \notin \mathcal{H}$$

$$\dfrac{\Gamma \Rightarrow P;(P,\mathcal{H}) \quad \Gamma \Rightarrow Q;(Q,\mathcal{H})}{\Gamma \Rightarrow P \wedge Q;\mathcal{H}}\ (\wedge_{\mathcal{R}}) \quad \text{if } P, Q \notin \mathcal{H} \qquad \dfrac{\Gamma \xrightarrow{P} D;\mathcal{H}}{\Gamma \xrightarrow{P \wedge Q} D;\mathcal{H}}\ (\wedge_{\mathcal{L}_1}) \qquad \dfrac{\Gamma \xrightarrow{Q} D;\mathcal{H}}{\Gamma \xrightarrow{P \wedge Q} D;\mathcal{H}}\ (\wedge_{\mathcal{L}_2})$$

$$\dfrac{\Gamma \Rightarrow P;(P,\mathcal{H})}{\Gamma \Rightarrow P \vee Q;\mathcal{H}}\ (\vee_{\mathcal{R}_1}) \quad \text{if } P \notin \mathcal{H} \qquad \dfrac{\Gamma \Rightarrow Q;(Q,\mathcal{H})}{\Gamma \Rightarrow P \vee Q;\mathcal{H}}\ (\vee_{\mathcal{R}_2}) \quad \text{if } Q \notin \mathcal{H}$$

$$\dfrac{\Gamma, P \Rightarrow D;\{D\} \quad \Gamma, Q \Rightarrow D;\{D\}}{\Gamma \xrightarrow{P \vee Q} D;\mathcal{H}}\ (\vee_{\mathcal{L}}) \quad \text{if } P \notin \Gamma \text{ and } Q \notin \Gamma$$

$$\dfrac{\Gamma \Rightarrow P;(P,\mathcal{H})}{\Gamma \Rightarrow \circ P;\mathcal{H}}\ (\circ_{\mathcal{R}}) \quad \text{if } P \notin \mathcal{H} \qquad \dfrac{\Gamma, P \Rightarrow \circ R;\{\circ R\}}{\Gamma \xrightarrow{\circ P} \circ R;\mathcal{H}}\ (\circ_{\mathcal{L}}) \quad \text{if } P \notin \Gamma$$

D is either an atom, $\bot$, disjunction or a $\circ$ formula. Where the history has been extended we have parenthesised $(P, \mathcal{H})$ for emphasis.

Fig. 5. The calculus PFLAX^{Hist} (Scottish).

essary, the left rules have side conditions to ensure that the context is increasing. For the $(\supset_{\mathcal{R}})$ rule (which attempts to extend the context) there are two cases corresponding to when the context is and when it is not extended. Something similar is happening in the left rules. Take $(\vee_{\mathcal{L}})$ as an example. In both premisses of the rule a formula may be added to context. If both contexts really are extended, then we continue building the proof tree. If one or both contexts are not extended then the sequent, S, with the non-extended context, will be the same as some sequent at a lesser height in the proof tree – there is a loop (which we describe as a trivial loop). This is easy to see: since the context and the goal of S are the same as that of the conclusion, there must be a lower sequent (the conclusion of an instance of (C)) the same as the premiss S. As an example we give a derivation in PFLAX^{Hist} of the sequent in the example in section 3:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{B, \circ B \wedge (B \supset A) \xrightarrow{B} B}\ (ax)}{B, \circ B \wedge (B \supset A) \Rightarrow B}\ (C) \qquad \dfrac{}{B, \circ B \wedge (B \supset A) \xrightarrow{A} A}\ (ax)}{B, \circ B \wedge (B \supset A) \xrightarrow{B \supset A} A}\ (\supset_{\mathcal{L}})}{B, \circ B \wedge (B \supset A) \xrightarrow{\circ B \wedge (B \supset A)} A}\ (\wedge_{\mathcal{L}_2})}{\dfrac{B, \circ B \wedge (B \supset A) \Rightarrow A}{B, \circ B \wedge (B \supset A) \Rightarrow \circ A}\ (\circ_{\mathcal{R}})}\ (C)$$

Note that the derivation of this sequent given in section 3 would be prevented by the history mechanism, as it contains a loop.

It is now demonstrated that PFLAX^{Hist} is equivalent to PFLAX, in terms of provability. The equivalence is proved via an intermediate calculus PFLAX^{D}. The calculus PFLAX^{D} is the calculus PFLAX where the rule (C) is restricted so that it is only applicable when the goal formula is an atom, a disjunction, falsum or a somehow formula.

Proposition 2 *The calculus* PFLAX *is equivalent to the calculus* PFLAX^{D}. *That is, sequent* $\Gamma \Rightarrow G$ *is provable in* PFLAX *iff* $\Gamma \Rightarrow G$ *is provable in* PFLAX^{D}.

The following lemma is needed in the proof of theorem 6.

Lemma 3 (Contraction) *The following rules are admissible in* PFLAX^{Hist}:

$$\frac{\Gamma, P, P \Rightarrow R; \mathcal{H}}{\Gamma, P \Rightarrow R; \mathcal{H}}\ (C') \qquad \frac{\Gamma, P, P \xrightarrow{Q} R; \mathcal{H}}{\Gamma, P \xrightarrow{Q} R; \mathcal{H}}\ (C'')$$

Proof. By simultaneous induction on the heights of derivations of premisses. □

The equivalence proof below, although long, has a simple structure. An algorithm to turn a PFLAX proof tree into a PFLAX^{Hist} proof tree is described in detail. A simple induction argument shows that the algorithm terminates, proving the result.

Theorem 6 *The calculi* PFLAX *and* PFLAX^{Hist} *are equivalent. That is, sequent* $\Gamma \Rightarrow G$ *is provable in* PFLAX *iff sequent* $\Gamma \Rightarrow G; \{G\}$ *is provable in* PFLAX^{Hist}.

Proof. From Proposition 2 we know that it is enough to show that PFLAX^{D} is equivalent to PFLAX^{Hist}. It is trivial that any sequent provable in PFLAX^{Hist} is provable in PFLAX^{D}. (Use contraction (C') above instances of $(\supset_{\mathcal{R}_2})$ and then simply drop the history part of the sequent). We prove the converse.

Take any proof tree for sequent $\Gamma \Rightarrow G$ in PFLAX^{D}. By definition this proof tree is finite, with $n > 0$ nodes. Using this proof tree, we construct (from the root up) a proof tree for the sequent $\Gamma \Rightarrow G; \{G\}$ in PFLAX^{Hist}. The major difference between PFLAX^{D} and PFLAX^{Hist} is that the former permits loops, whereas the latter does not. Essentially we take the PFLAX^{D} proof tree and give a recipe for 'snipping out' the loops: removing the sequents that form the loop.

The construction uses 'hybrid trees'. A hybrid tree is a fragment of a PFLAX^{Hist} proof tree with all branches that do not have (ax), $(\top)$ or $(\bot)$ leaves ending with PFLAX^{D} proof trees. These PFLAX^{D} proof trees have roots which can be obtained by backwards application of a PFLAX^{D} rule to the top history sequent (ignoring its history). We analyse each case of a topmost history sequent with non-history premiss(es) resulting from application of rule (R) in the sequent tree. We write $\Gamma \sim \Gamma'$ when the set of formulae in multisets Γ and Γ' are the same (although the number of occurrences may be different). We denote a series of zero or more instances of rule (R) by $(R)^*$. We give the proof for the $\circ, \supset$ fragment.

- The root of the PFLAX^D tree. We change (non-history) sequent $\Gamma \Rightarrow G$ to history sequent $\Gamma \Rightarrow G; \{G\}$.
- (R) is one of (ax), (C), i.e. a rule which in PFLAX^{Hist} has no side conditions. The premiss is changed by adding the appropriate history. It becomes the history sequent obtained by applying (backwards) the PFLAX^{Hist} rule to the original conclusion. For example, if the situation we are analysing is

$$\frac{\Gamma, P \xrightarrow{P} D}{\Gamma, P \Rightarrow D; \mathcal{H}}\ (C) \qquad \text{then it becomes} \qquad \frac{\Gamma, P \xrightarrow{P} D; \mathcal{H}}{\Gamma, P \Rightarrow D; \mathcal{H}}\ (C)$$

  We now have a new hybrid tree.
- (R) is $(\supset_{\mathcal{R}})$. If the context is extended, then add the appropriate history, allowing the replacement of the instance of $(\supset_{\mathcal{R}})$ in PFLAX^D by an instance of $(\supset_{\mathcal{R}_1})$ in PFLAX^{Hist}. If the context is not extended and the new goal is not in the history, then again add the appropriate history, allowing the replacement of the instance of $(\supset_{\mathcal{R}})$ in PFLAX^D by an instance of $(\supset_{\mathcal{R}_2})$ with a (C') in PFLAX^{Hist}. If the new goal is in the history, there is a loop, which the history mechanism prevents. If the history condition is not met, then below the conclusion the hybrid tree has the form:

$$\begin{array}{c}\dfrac{\Gamma, P \Rightarrow G}{\Gamma \Rightarrow P \supset G; \mathcal{H}}\ (\supset_{\mathcal{R}}) \\ \vdots \\ \Gamma' \Rightarrow G; \mathcal{H}'\end{array}$$

  where $G \in \mathcal{H}'$, $\mathcal{H}' \subseteq \mathcal{H}$ and $\Gamma \sim \Gamma'$. The history is not reset at any point in this fragment. This can easily be seen to contain the loop which is the reason for the history condition not being met. It is transformed by removing all sequents above, but not including, $\Gamma' \Rightarrow G; \mathcal{H}'$ (along with any subtrees above excised sequents) up to $\Gamma, P \Rightarrow G$. Adding the appropriate history to this sequent and using (one or more instances of) (C') gives the new hybrid tree:

$$\frac{\Gamma, P \Rightarrow G; \mathcal{H}'}{\Gamma' \Rightarrow G; \mathcal{H}'}\ (C')^*$$

- (R) is $(\supset_{\mathcal{L}})$. If the history condition is satisfied, then add the appropriate history, allowing the replacement of instance of $(\supset_{\mathcal{L}})$ in PFLAX^D by an instance of $(\supset_{\mathcal{L}})$ in PFLAX^{Hist}. If the history condition is not satisfied, then below the conclusion the hybrid tree has the form:

$$\begin{array}{c}\dfrac{\Gamma \Rightarrow P \quad \Gamma \xrightarrow{Q} R}{\Gamma \xrightarrow{P \supset Q} R; \mathcal{H}}\ (\supset_{\mathcal{L}}) \\ \vdots \\ \Gamma' \Rightarrow P; \mathcal{H}'\end{array}$$

  where $P \in \mathcal{H}'$, $\mathcal{H}' \subseteq \mathcal{H}$ and $\Gamma \sim \Gamma'$. The history is not reset at any point in this fragment. It is transformed by removing all the sequents above, but not including, $\Gamma' \Rightarrow P; \mathcal{H}'$ (along with any subtrees above excised sequents) up to $\Gamma \Rightarrow P$. The sequent $\Gamma \xrightarrow{Q} R$ and the subtree above it are also removed. Adding the appropriate history to $\Gamma \Rightarrow P$ and using (zero or more instances of) (C') gives the new hybrid tree:

$$\frac{\Gamma \Rightarrow P; \mathcal{H}'}{\Gamma' \Rightarrow P; \mathcal{H}'}\ (C')^*$$

– (R) is $(\circ_{\mathcal{R}})$. If the history condition is satisfied, then add the appropriate history, allowing the replacement of the instance of $(\circ_{\mathcal{R}})$ in PFLAX^D by an instance of $(\circ_{\mathcal{R}})$ in PFLAX^{Hist}. If the history condition is not satisfied, then below the conclusion the hybrid tree has form:

$$\begin{array}{c}\dfrac{\Gamma \Rightarrow P}{\Gamma \Rightarrow \circ P; \mathcal{H}}\ (\circ_{\mathcal{R}}) \\ \vdots \\ \Gamma' \Rightarrow P; \mathcal{H}'\end{array}$$

where $P \in \mathcal{H}'$, $\mathcal{H}' \subseteq \mathcal{H}$ and $\Gamma \sim \Gamma'$. The history is not reset at any point in this fragment. It is transformed by removing all sequents from, but not including, $\Gamma' \Rightarrow P; \mathcal{H}'$ (along with any subtrees above excised sequents) up to $\Gamma \Rightarrow P$. Adding the appropriate history and using (zero or more instances of) (C') gives the new hybrid tree:

$$\frac{\Gamma \Rightarrow P; \mathcal{H}'}{\Gamma' \Rightarrow P; \mathcal{H}'}\ (C')^*$$

– (R) is $(\circ_{\mathcal{L}})$. If the side condition is satisfied, then add the appropriate history, allowing the replacement of the instance of $(\circ_{\mathcal{L}})$ in PFLAX^D by an instance of $(\circ_{\mathcal{L}})$ in PFLAX^{Hist}. If the side condition is not satisfied, then below the conclusion the hybrid tree has form:

$$\begin{array}{c}\dfrac{\Gamma, P \Rightarrow \circ R}{\Gamma \xrightarrow{\circ P} \circ R; \mathcal{H}}\ (\circ_{\mathcal{L}}) \\ \vdots \\ \Gamma \Rightarrow \circ R; \mathcal{H}\end{array}$$

where $P \in \Gamma$. This is transformed by removing all sequents from, but not including, $\Gamma \Rightarrow \circ R; \mathcal{H}$ (along with any subtrees above excised sequents) up to $\Gamma, P \Rightarrow \circ R$. Adding the appropriate history and using a single instance of (C') gives the new hybrid tree:

$$\frac{\Gamma, P \Rightarrow \circ R; \mathcal{H}}{\Gamma \Rightarrow \circ R; \mathcal{H}}\ (C')$$

Since the number of sequents without a history in a hybrid tree is finite and as every step strictly decreases the number of sequents without a history, this process is terminating. The instances of (C') may be eliminated from the constructed derivation. □

Note that the PFLAX derivation given in section 3 is transformed by the algorithm described in the proof to the derivation of the same sequent in PFLAX^{Hist} given above.

We have shown that PFLAX^{Hist} is sound and complete. To prove that it is a decision procedure, we prove that it is also terminating – backwards proof search in the calculus ends in success or failure after a finite number of steps.

Theorem 7 *Backwards proof search in the calculus* PFLAX^{Hist} *is terminating.*

Proof. We associate with every sequent a quintuple of natural numbers. With a sequent without a stoup, $\Gamma \Rightarrow R; \mathcal{H}$, we associate: $W = (k-n, k-m, 1, 0, r)$. With a sequent with a stoup, $\Gamma \xrightarrow{P} R; \mathcal{H}$, we associate: $W = (k-n, k-m, 0, s, r)$. Here, k is the number of elements in the *set* of subformulae of (Γ, R); n is the number of elements in the *set* of elements of Γ; m is the number of elements in $\mathcal{H}$; r is the size of goal formula R and s is the size of the stoup formula P. (Notice that although Γ is a multiset, we count its elements as a set). These quintuples are lexicographically ordered from the left. By inspection we see that for every inference rule W for the premisses is lower in the lexicographic order than W for the conclusion. Hence backward proof search is terminating. $\square$

When implementing a theorem prover, knowledge of the invertibility of the inference rules can be useful. This information is given in the following proposition.

Proposition 3 *The following inference rules of* PFLAX^{Hist} *are invertible:* $(\supset_{\mathcal{R}1})$, $(\supset_{\mathcal{R}2})$, $(\neg_{\mathcal{R}1})$, $(\neg_{\mathcal{R}2})$, $(\supset_{\mathcal{L}})$, $(\neg_{\mathcal{L}})$, $(\wedge_{\mathcal{R}})$, $(\vee_{\mathcal{L}})$, $(\circ_{\mathcal{L}})$. *The following inference rules of* PFLAX^{Hist} *are not invertible:* (C), $(\wedge_{\mathcal{L}_1})$, $(\wedge_{\mathcal{L}_2})$, $(\vee_{\mathcal{R}_1})$, $(\vee_{\mathcal{R}_2})$, $(\circ_{\mathcal{R}})$.

6. Conclusion

This paper has presented two proof search calculi for Lax Logic. The first, PFLAX, is a sequent calculus for first-order quantified Lax Logic. The proofs allowed by this calculus naturally correspond in a 1–1 way to the normal natural deductions for first-order quantified Lax Logic. The calculus is well suited for enumerating, without redundancy, all proofs in the logic. This makes the calculus useful in contexts where proof search is for normal natural deductions, such as in (constraint) logic programming.

The second calculus, PFLAX^{Hist}, builds on the propositional fragment of the first calculus to give a decision procedure for propositional Lax Logic. Propositional Lax Logic has been used in hardware verification and PFLAX^{Hist} could be of use in this area. The calculus works by adding a history mechanism to the propositional calculus to prevent looping. This technique is general and may be applied to a wide range of sequent calculi for propositional logics to yield decision procedures. We believe that, to date, PFLAX^{Hist} is the only effective decision procedure for propositional Lax Logic.

Acknowledgements I would like to thank Roy Dyckhoff for his helpful advice during many useful and interesting discussions.

References

Avellone, A. and Ferrari, M. (1996). Almost Duplication-free Tableau Calculi for Propositional Lax Logics. In *TABLEAUX'96*, volume 1071 of *Lecture Notes in Artificial Intelligence*, pages 48–64. Springer-Verlag.

Benton, P. N., Bierman, G. M., and de Paiva, V. (1998). Computational Types from a Logical Perspective. *Journal of Functional Programming*, 8(2):177–193.

Curry, H. B. (1952). The Elimination Theorem When Modality is Present. *Journal of Symbolic Logic*, 17(4):249–65.

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

Dyckhoff, R. and Pinto, L. (1996). A Permutation-free Sequent Calculus for Intuitionistic Logic. Technical Report CS/96/9, University of St Andrews.

Dyckhoff, R. and Pinto, L. (1998). Cut-Elimination and a Permutation-free Sequent Calculus for Intuitionistic Logic. *Studia Logic*, 60:107–118.

Dyckhoff, R. and Pinto, L. (1999). Permutability of Proofs in Intuitionistic Sequent Calculi. *Theoretical Computer Science*, 212(1-2):141–155.

Fairtlough, M. and Mendler, M. (1994). An Intuitionistic Modal Logic with Applications to the Formal Verification of Hardware. In *Computer Science Logic*, pages 354–68. Springer-Verlag.

Fairtlough, M. and Mendler, M. (1997). Propositional Lax Logic. *Information and Computation*, 137(1):1–33.

Fairtlough, M., Mendler, M., and Walton, M. (1997). First-order Lax Logic as a Framework for Constraint Logic Programming. Technical Report MIPS-9714, University of Passau.

Fairtlough, M. and Walton, M. (1997). Quantified Lax Logic. Technical Report CS-97-11, University of Sheffield.

Gabbay, D. (1991). Algorithmic Proof with Diminishing Resources, part 1. In *Computer Science Logic 1990*, volume 533 of *Lecture Notes in Computer Science*, pages 156–173. Springer-Verlag.

Gentzen, G. (1969). *The Collected Papers of Gerhard Gentzen*. North-Holland, Amsterdam. Edited M. E. Szabo.

Girard, J.-Y. (1991). A New Constructive Logic: Classical Logic. *Mathematical Structures in Computer Science*, 1:255–296.

Herbelin, H. (1995). A λ-calculus Structure Isomorphic to Gentzen-style Sequent Calculus Structure. In Pacholski, L. and Tiuryn, J., editors, *Computer Science Logic 1994*, volume 933 of *Lecture Notes in Computer Science*, pages 61–75. Springer-Verlag.

Heuerding, A. (1998). *Sequent Calculi for Proof Search in Some Modal Logics*. PhD thesis, Universität Bern.

Heuerding, A., Seyfried, M., and Zimmermann, H. (1996). Efficient Loop-Check for Backward Proof Search in Some Non-classical Propositional Logics. In *TABLEAUX'96*, volume 933, pages 61–75. Springer-Verlag.

Howe, J. M. (1997). Two Loop Detection Mechanisms: a Comparison. *Lecture Notes in Artificial Intelligence*, 1227:188–200.

Howe, J. M. (1998). *Proof Search Issues in Some Non-Classical Logics*. PhD thesis, University of St Andrews. Available as Technical Report CS/99/1 and electronically from `http://www.cs.ukc.ac.uk/people/staff/jmh1`.

Howe, J. M. (1999). Proof Search in Lax Logic. Technical Report 14-99, University of Kent.

Hudelmaier, J. (1993). An O(n log n)-space Decision Procedure for Intuitionistic Propositional Logic. *Journal of Logic and Computation*, 3(1):63–75.

Mendler, M. (1993). *A Modal Logic for Handling Behavioural Constraints in Formal Hardware Verification*. PhD thesis, University of Edinburgh. ECS-LFCS-93-255.

Miller, D., Nadathur, G., Pfenning, F., and Scedrov, A. (1991). Uniform Proofs as a Foundation for Logic Programming. *Annals of Pure and Applied Logic*, 51(1-2):125–157.

Moggi, E. (1989). Computational Lambda-Calculus and Monads. In *Logic in Computer Science '89*, pages 14–23.

Prawitz, D. (1965). *Natural Deduction*, volume 3 of *Stockholm Studies in Philosophy*. Almqvist & Wiksell, Stockholm.

Walton, M. (1998). *First-Order Lax Logic: A Framework for Abstraction, Constraints and Refinements*. PhD thesis, University of Sheffield.