J. Inform. Process. Cybernet. **EIK 29** (1994) 6, 333–355
(formerly: Elektron. Inform.verarb. Kybernet.)

Elimination of Cuts in First-order Finite-valued Logics

By Matthias Baaz, Christian G. Fermüller and Richard Zach

Abstract: A uniform construction for sequent calculi for finite-valued first-order logics with distribution quantifiers is exhibited. Completeness, cut-elimination and midsequent theorems are established. As an application, an analog of Herbrand's theorem for the four-valued knowledge-representation logic of Belnap and Ginsberg is presented. It is indicated how this theorem can be used for reasoning about knowledge bases with incomplete and inconsistent information.

1 Introduction

Why elimination of cuts? Cut-elimination procedures play a central rôle in proof theoretic investigations of logical calculi. Their importance lies in the fact that they are *local* procedures. This is in contrast to the results obtained by proofs of cut-free completeness, which are global (and do not take existing proofs into account). For the *analysis* of given proofs—and this makes cut-elimination so important for computer science—a local procedure, which operates on given data, is needed. It allows one to look at parts of proofs independently of others, e.g., making the assumptions of a proof explicit. Cut elimination is also essential for the extraction of programs from proofs. The constructive nature of the cut elimination procedure ensures that the extraction algorithm of a function encoded by a proof is primitive recursive, whereas cut-free completeness properties (as in the case of analytic tableaux) only guarantees termination of extraction algorithms. Cut elimination also provides a bound for the term complexity of the cut-free proof, and consequently a bound on the complexity of the extracted function *relative* to the complexity of the functions represented by primitive function symbols.

Why many-valued logics? Many-valued logics have enjoyed rising interest in recent years in computer science. For instance, many-valued logics can be used to model knowledge bases, or more accurately, epistemic states of knowledge. In the following, we will illustrate our results with an example based on such a logic: Belnap [6] has introduced a logic with the truth values *false* (f), *unknown* (u), *contradictory* ($\bot$) and *true* (t), corresponding to the epistemic status of the given facts. Ginsberg [12] has extended the study of logics of this kind to the general theory of logics over *bilattices* and uses it to model non-monotonic knowledge bases. Belnap's logic has recently been used for a representation of (possibly contradictory) knowledge and the problem of knowledge revision and update in [17]. Bilattice logics also find applications in the study of logic programming, see, e.g., [9, 10]. Another recent application of many-valued logics is in the verification of switch-level circuit designs [16].

The success of a logic for use in the context of knowledge representation and reasoning depends crucially on the availability of computational calculi for this logic. Without such calculi,

the study of a logic in relation to its application in computer science and artificial intelligence is destined to remain purely theoretical. In the case of many-valued logics, computational calculi can be given.

Analytic calculi for many-valued logics have been known for quite some time. Sequent systems similar to those presented here have also been introduced by Schröter [22], Rousseau [21], Takahashi [25], and Carnielli [8]; equivalent tableaux formulations were given by Surma [24] and Carnielli [7] (Hähnle's work is based on the latter [13, 14, 15]). Calculi for automated theorem proving in many-valued logics are also legion. Apart from Hähnle's work on tableaux-based theorem proving, various resolution methods have been proposed, e.g., by O'Hearn and Stachniak [19] or Baaz and Fermüller [1, 2]. For more detailed surveys of the work done in these areas see [15, 27].

We present here sequent calculi for arbitrary finite-valued first-order logics with distribution quantifiers. These calculi are essentially the ones of [21] but differ from those of [25] (our rules are more general, and also more compact) and [8] (which is a calculus *dual* to ours, cf. [4]). First we establish soundness and completeness for this calculus. In Section 4 we give a cut-elimination algorithm *schema* for this calculus modulo a parametric operator for reducing cuts on composite formulas. In the usual proof of the *Hauptsatz* for classical logic, this operator is *fixed* and defined by case distinction. We, however, give (in Section 5) a general method based on the resolution principle for finding such reductions. This method is also applicable to the classical case. It provides a uniform way to show that cut-elimination holds in systems where any propositional connectives or distribution quantifiers (which are, of course, definable in classical logic) are taken as *primitive*.

Our method can be extended so as to reduce the degree of the cut formula not only by one, but to reduce it arbitrarily. This suggests the possibility of *interactively* reducing cuts of higher complexity *in one step*. This may have an application for implementing cut-elimination procedures, an undertaking thwarted until now due to its prohibitive complexity.

Finally, in Section 6, we give an analog of Herbrand's Theorem for a first-order version of Belnap's logic and show how this can be used to extract information from proofs in the sequent calculus.

2 Preliminaries

Definition 2.1. A *language* $\mathcal{L}$ for a logic consists of countably many free variables, bound variables, function symbols (including constants), predicate symbols, as well as propositional connectives, quantifiers, and auxiliary symbols: "(", ")", ",".

We use a, b, c, ... to denote free variables; x, y, z, ... to denote bound variables; P, Q, R, ... to denote predicate symbols; $\Box$ to denote connectives; and Q to denote quantifiers.

Terms, subterms, formulas and subformulas are defined as usual. The *degree* of a formula is its depth, more precisely:

$$d(A) = \begin{cases} 0 & \text{if } A \text{ is atomic} \\ \max\{d(A_1), \ldots, d(A_n)\} + 1 & \text{if } A \equiv \Box(A_1, \ldots, A_n) \\ d(B) + 1 & \text{if } A \equiv (\mathsf{Q}x)A \end{cases}$$

Definition 2.2. A *matrix* for a language $\mathcal{L}$ is given by:

1. a nonempty set of *truth values* $V = \{v_1, \ldots, v_m\}$ of size m,
2. an abstract algebra $\mathbf{V}$ with domain V of appropriate type: For every n-place connective $\Box$ of $\mathcal{L}$ there is an associated *truth function* $\widetilde{\Box}\colon V^n \to V$, and
3. for every quantifier Q, an associated *truth function* $\widetilde{\mathsf{Q}}\colon \wp(V) \setminus \{\emptyset\} \to V$

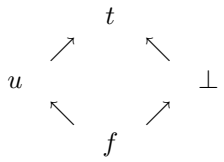
A language and a matrix for it together fully determine a *logic* **L**. **L** is said to be *m-valued*.

The intended meaning of a truth function for a propositional connective is analogous to the two-valued case: Given formulas $A_1, \ldots, A_n$, which take the truth values $w_1, \ldots, w_n$, respectively, the truth value of $\Box(A_1, \ldots, A_n)$ is given by $\widetilde{\Box}(w_1, \ldots, w_n)$.

A truth function for quantifiers is a mapping from nonempty sets of truth values to truth values: Given a quantified formula $(\mathsf{Q}x)F(x)$, such a set of truth values describes the situation where the ground instances of F take exactly the truth values in this set as values under a given interpretation. In other words, for a non-empty set $M \subseteq V$, $(\mathsf{Q}x)F(x)$ takes the truth value $\widetilde{\mathsf{Q}}(M)$ if, for every truth value $v \in V$, it holds that $v \in M$ iff there is a domain element d such that the truth value of $F(d)$ is v. The set M is called the *distribution* of F. This generalization of quantifiers dates back to [18]. Quantifiers of this type have been called *distribution quantifiers* in [7].

Example 2.1. The matrix for Belnap's logic consists of:

1. The set of truth values $V = \{f, u, \bot, t\}$. V carries the structure of a *bilattice* with two partial orders $\leq_k$ and $\leq_t$ as follows: $f <_t u, \bot <_t t$ and $u <_k t, f <_k \bot$.



2. The truth functions for the connectives $\neg$, $\wedge$, $\vee$ ($\wedge$ and $\vee$ are the glb and lub in the order $\leq_t$, respectively):

$\neg$	
t	f
u	u
$\bot$	$\bot$
f	t

$\wedge$	t	u	$\bot$	f
t	t	u	$\bot$	f
u	u	u	f	f
$\bot$	$\bot$	f	$\bot$	f
f	f	f	f	f

$\vee$	t	u	$\bot$	f
t	t	t	t	t
u	t	u	t	u
$\bot$	t	t	$\bot$	$\bot$
f	t	u	$\bot$	f

   Furthermore, we have unary connectives J_w for every $w \in V$ where $\mathsf{J}_w(u) = t$ if $u = w$ and $\mathsf{J}_w(u) = f$ otherwise.
3. In view of the application in Section 6, we introduce the quantifiers $\forall$ (classical for all) and U (uniformity) defined by the following truth functions:

$$\widetilde{\forall}(W) = \begin{cases} t & \text{if } W = \{t\} \\ f & \text{otherwise} \end{cases} \qquad \widetilde{\mathsf{U}}(W) = \mathrm{lub}_{\leq_k}(W)$$

Definition 2.3. A *structure* $\mathcal{M} = \langle D, \Phi_{\mathcal{M}} \rangle$ for a language $\mathcal{L}$ ($\mathcal{L}$-structure) consists of the following:

1. A non-empty set D, called the *domain* (elements of D are called *individuals*.
2. A mapping $\Phi_{\mathcal{M}}$ that satisfies the following:
   (a) Each free variable of $\mathcal{L}$ is mapped to an element of D.
   (b) Each n-ary function symbol f of $\mathcal{L}$ is mapped to a function $f_{\mathcal{M}}\colon D^n \to D$, or to an element of D if $n = 0$. Additionally, $\Phi_{\mathcal{M}}$ maps elements of D to themselves.
   (c) Each n-ary predicate symbol P of $\mathcal{L}$ is mapped to a function $P_{\mathcal{M}}\colon D^n \to V$, or to and element of V if $n = 0$.

Definition 2.4. Let $\mathcal{M}$ be an $\mathcal{L}$-structure. An *assignment* s is a mapping from the free variables of $\mathcal{L}$ to individuals.

Definition 2.5. An *interpretation* $\mathbf{I} = \langle \mathcal{M}, s \rangle$ is an $\mathcal{L}$-structure $\mathcal{M} = \langle D, \Phi \rangle$ together with an assignment s.

Definition 2.6. Let $\mathbf{I} = \langle \langle D, \Phi_{\mathcal{M}} \rangle, s \rangle$ be an interpretation. The mapping Φ can be extended in the obvious way to a mapping from terms to individuals:

1. If t is a free variable, then $\Phi_{\mathbf{I}}(t) = s(t)$.
2. If t is of the form $f(t_1, \ldots, t_2)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms, then $\Phi_{\mathbf{I}}(t) = f_{\mathcal{M}}\big(\Phi_{\mathbf{I}}(t_1), \ldots, \Phi_{\mathbf{I}}(t_n)\big)$.

Definition 2.7. Given an interpretation $\mathbf{I} = \langle \mathcal{M}, s \rangle$, we define the *valuation* $\mathrm{val}_{\mathbf{I}}$ for formulas F to be a mapping from formulas to truth values as follows:

1. If F is atomic, i.e., of the form $P(t_1, \ldots, t_n)$, where P is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms, then $\mathrm{val}_{\mathbf{I}}(F) = P_{\mathcal{M}}\big(\Phi_{\mathbf{I}}(t_1), \ldots, \Phi_{\mathbf{I}}(t_n)\big)$.
2. If the outermost logical symbol of F is a propositional connective $\Box$ of arity n, i.e., F is of the form $\Box(F_1, \ldots, F_n)$, where $F_1, \ldots, F_n$ are formulas, then $\mathrm{val}_{\mathbf{I}}(F) = \widetilde{\Box}\big(\mathrm{val}_{\mathbf{I}}(F_1), \ldots, \mathrm{val}_{\mathbf{I}}(F_n)\big)$.
3. If the outermost logical symbol of F is a quantifier Q, i.e., F is of the form $(\mathsf{Q}x)G(x)$, then
$$\mathrm{val}_{\mathbf{I}}(F) = \widetilde{\mathsf{Q}}\big(\{\mathrm{val}_{\mathbf{I}} G(d) \mid d \in D\}\big).$$

3 Sequent calculi

Definition 3.1. An *(m-valued) sequent* Γ is an m-tuple of finite sequences $\Gamma_{(i)}$ of formulas written thus: $\Gamma_{(1)} \mid \ldots \mid \Gamma_{(m)}$. If Γ is a sequent, then $\Gamma_{(i)}$ denotes the i-th component of Γ.

If Δ is a sequence of formulas and $I \subseteq M = \{1, \ldots, m\}$ (or $W \subseteq V$), we denote by $[I\colon \Delta]$ ($[W\colon \Delta]$) the sequent whose i-th component is Δ if $i \in I$ ($v_i \in W$), and is empty otherwise. For $[\{i, j, \ldots\}\colon \Delta]$ we write $[i, j, \ldots\colon \Delta]$. If Γ and Γ' are sequents, then we write Γ, Γ' for the sequent $\Gamma_{(1)}, \Gamma'_{(1)} \mid \ldots \mid \Gamma_{(m)}, \Gamma'_{(m)}$.

Definition 3.2. Let $\mathbf{I}$ be an interpretation. $\mathbf{I}$ satisfies a sequent Γ, iff there is an i, $1 \leq i \leq m$, s.t., for some formula $F \in \Gamma_{(i)}$, $\mathrm{val}_{\mathbf{I}}(F) = v_i$. $\mathbf{I}$ is called a *model* of Γ and we write $\mathbf{I} \models \Gamma$.

Γ is called *satisfiable*, iff there is an interpretation $\mathbf{I}$ s.t. $\mathbf{I} \models \Gamma$, and *valid*, iff for every interpretation $\mathbf{I}$, $\mathbf{I} \models \Gamma$.

The above definition boils down to the interpretation of an m-sided sequent as a *disjunction* of statements saying that a particular formula takes a particular truth value. Specifically, if a formula A occurs in $\Gamma_{(i)}$, the interpretation is "A takes the value v_i." It is instructive to see that this interpretation yields the usual interpretation in the two-valued case: there, a sequent $A_1, \ldots, A_r \to B_1, \ldots, B_s$ is commonly read as being equivalent to $(A_1 \wedge \ldots \wedge A_r) \supset (B_1 \vee \ldots \vee B_s)$. By propositional logic, this is equivalent to $\neg A_1 \vee \ldots \vee \neg A_r \vee B_1 \vee \ldots \vee B_s$, or in plain English: either one of the A_i's is false or one of the B_j's is true.

Remark 3.1. It is also possible to interpret sequents in a dual way, namely: if a formula A occurs in $\Gamma_{(i)}$, the interpretation is "A *does not* take the value v_i." Sequent calculi for this semantics can also be uniformly obtained, they correspond to analytic tableaux [4, 27]

Definition 3.3. An *introduction rule for a connective* $\Box$ *at place* i is a schema of the form:

$$\frac{\langle \Gamma, \Delta_{\Box:i}(j)\rangle_{j\in I}}{\Gamma, [i\colon \Box(A_1,\ldots,A_n)]}\ \Box{:}i$$

where the arity of $\Box$ is n, I is a finite set, $\mathrm{frm}\big(\Delta_{\Box:i}(j)\big) \subseteq \{A_1,\ldots,A_n\}$ (where $\mathrm{frm}(\Delta)$ denotes the set of formulas occurring in Δ) and the following condition holds:

Let $\mathbf{I}$ be an interpretation. Then the following are equivalent:

1. $\Box(A_1,\ldots,A_n)$ takes the truth value v_i under $\mathbf{I}$.
2. For $j \in I$, $\mathcal{M}$ satisfies the sequents $\Delta_{\Box:i}(j)$.

It should be stressed that the introduction rules for a connective at a given place are far from being unique: Let the expression A^{v_l} denote the statement "A takes the truth value v_l". Then every introduction rule for $\Box(A_1,\ldots,A_n)$ at place i corresponds to a conjunction of disjunctions of some A^{v_l} which is true iff $\Box(A_1,\ldots,A_n)$ takes the truth value v_i (namely, $\bigwedge_{j=1}^{k}\bigvee_{l=1}^{m}\bigvee_{A\in\Delta_{\Box:i}(j)_{(l)}} A^{v_l}$). This represents an i-th partial normal form in the sense of [20]. *Any* such partial normal form for $\Box(A_1,\ldots,A_n)^{v_i}$ will do.

In particular, the truth table for $\Box$ immediately yields a *complete* conjunctive normal form, the corresponding rule is as in Definition 3.3, with: $I \subseteq V^n$ is the set of all n-tuples $j = (w_1,\ldots,w_n)$ of truth values such that $\tilde{\Box}(w_1,\ldots,w_n) \neq v_i$; and $\Delta_{\Box:i}(j)_{(l)} = \{A_k \mid 1 \leq k \leq n, v_l \neq w_k\}$.

A rule constructed this way can have up to $m^n - 1$ premises (if $\Box$ takes the value v_i only once in the truth table), but standard methods for minimizing combinational functions, such as the Quine-McCluskey procedure, can be used to find rules that are minimal w.r.t. the number of premises and the number of formulas per premise. This procedure has been implemented in the system MULTLOG [3]. For an analysis of upper bounds for the number of premises for propositional and quantifier rules see [27, Ch. 1].

If a connective never takes a particular truth value v_i, then the introduction rule at place i is actually a weakening rule (see below).

Example 3.1. Consider the disjunction in Belnap's logic: The partial normal forms

$$\begin{array}{lll}
(A\vee B)^f & \text{iff} & A^f \text{ and } B^f\\
(A\vee B)^u & \text{iff} & (A^f \text{ or } A^u) \text{ and } (B^f \text{ or } B^u) \text{ and } (A^u \text{ or } B^u)\\
(A\vee B)^\bot & \text{iff} & (A^f \text{ or } A^\bot) \text{ and } (B^f \text{ or } B^\bot) \text{ and } (A^\bot \text{ or } B^\bot)\\
(A\vee B)^t & \text{iff} & A^t \text{ or } B^t
\end{array}$$

yield the following introduction rules:

$$\frac{\Gamma, [f\colon A] \quad \Gamma, [f\colon B]}{\Gamma, [f\colon A\vee B]}\ \vee{:}f \qquad \frac{\Gamma, [f, u\colon A] \quad \Gamma, [f, u\colon B] \quad \Gamma, [u\colon A, B]}{\Gamma, [u\colon A\vee B]}\ \vee{:}u$$

$$\frac{\Gamma, [t\colon A, B]}{\Gamma, [t\colon A\vee B]}\ \vee{:}t \qquad \frac{\Gamma, [f, \bot\colon A] \quad \Gamma, [f, \bot\colon B] \quad \Gamma, [\bot\colon A, B]}{\Gamma, [\bot\colon A\vee B]}\ \vee{:}\bot$$

The other rules are as follows:

$$\frac{\Gamma, [f\colon A, B]}{\Gamma, [f\colon A\wedge B]}\ \wedge{:}f \qquad \frac{\Gamma, [t, u\colon A] \quad \Gamma, [t, u\colon B] \quad \Gamma, [u\colon A, B]}{\Gamma, [u\colon A\wedge B]}\ \wedge{:}u$$

$$\frac{\Gamma, [t\colon A] \quad \Gamma, [t\colon B]}{\Gamma, [t\colon A\wedge B]}\ \wedge{:}t \qquad \frac{\Gamma, [f, \bot\colon A] \quad \Gamma, [f, \bot\colon B] \quad \Gamma, [\bot\colon A, B]}{\Gamma, [\bot\colon A\wedge B]}\ \wedge{:}\bot$$

$$\frac{\Gamma, [t\colon A]}{\Gamma, [f\colon \neg A]}\ \neg\!:\!f \qquad \frac{\Gamma, [u\colon A]}{\Gamma, [u\colon \neg A]}\ \neg\!:\!u \qquad \frac{\Gamma, [\bot\colon A]}{\Gamma, [\bot\colon \neg A]}\ \neg\!:\!\bot \qquad \frac{\Gamma, [f\colon A]}{\Gamma, [t\colon \neg A]}\ \neg\!:\!t$$

$$\frac{\Gamma, [f\colon A]}{\Gamma, [t\colon \mathsf{J}_f(A)]}\ \mathsf{J}_f\!:\!t \qquad \frac{\Gamma, [u\colon A]}{\Gamma, [t\colon \mathsf{J}_u(A)]}\ \mathsf{J}_u\!:\!t \qquad \frac{\Gamma, [\bot\colon A]}{\Gamma, [t\colon \mathsf{J}_\bot(A)]}\ \mathsf{J}_\bot\!:\!t \qquad \frac{\Gamma, [t\colon A]}{\Gamma, [t\colon \mathsf{J}_t(A)]}\ \mathsf{J}_t\!:\!t$$

$$\frac{\Gamma, [u, \bot, t\colon A]}{\Gamma, [f\colon \mathsf{J}_f(A)]}\ \mathsf{J}_f\!:\!f \qquad \frac{\Gamma, [f, \bot, t\colon A]}{\Gamma, [f\colon \mathsf{J}_u(A)]}\ \mathsf{J}_u\!:\!f \qquad \frac{\Gamma, [f, u, t\colon A]}{\Gamma, [f\colon \mathsf{J}_\bot(A)]}\ \mathsf{J}_\bot\!:\!f \qquad \frac{\Gamma, [f, u, \bot\colon A]}{\Gamma, [f\colon \mathsf{J}_t(A)]}\ \mathsf{J}_t\!:\!f$$

The remaining rules (J_v:w) for $v \neq w$ are weakenings at place w.

Definition 3.4. An *introduction rule for a quantifier* Q *at place* i in the logic $\mathbf{L}$ is a schema of the form:

$$\frac{\langle \Gamma, \Delta_{\mathsf{Q}:i}(j)\rangle_{j\in I}}{\Gamma, [i\colon (\mathsf{Q}x)A(x)]}\ \mathsf{Q}\!:\!i$$

where I is a finite set, $\Delta_{\mathsf{Q}:i}(j)_{(k)} \subseteq \{A(\alpha_1), \ldots, A(\alpha_q), A(\tau_1), \ldots, A(\tau_p)\}$ $(1 \leq k \leq m)$, the α_r are metavariables for free variables (the *eigenvariables* of the rule) satisfying the condition that they do not occur in the lower sequent, the τ_s are metavariables for terms, and the following condition holds:

For every interpretation $\mathbf{I}$ it holds that

1. If for all $d_1, \ldots, d_q \in D$ there are $e_1, \ldots, e_p \in D$ s.t.

$$\mathbf{I} \models \Delta_{\mathsf{Q}:i}(j)\{e_1/\tau_1, \ldots, e_p/\tau_p, d_1/\alpha_1, \ldots, d_q/\alpha_q\}$$

   then $\mathrm{val}_{\mathbf{I}}\big((\mathsf{Q}x)A(x)\big) = v_i$.
2. If for all $e_1, \ldots, e_p \in D$ there are $d_1, \ldots, d_q \in D$ s.t.

$$\mathbf{I} \not\models \Delta_{\mathsf{Q}:i}(j)\{e_1/\tau_1, \ldots, e_p/\tau_p, d_1/\alpha_1, \ldots, d_q/\alpha_q\}$$

   then $\mathrm{val}_{\mathbf{I}}\big((\mathsf{Q}x)A(x)\big) \neq v_i$.

In the case of two-valued classical logic, we have the well-known introduction rules (at place "true") for $\forall$ and $\exists$:

$$\frac{\Gamma \to \Delta, A(\alpha)}{\Gamma \to \Delta, (\forall x)A(x)}\ \forall\!:\!t \qquad \frac{\Gamma \to \Delta, A(\tau)}{\Gamma \to \Delta, (\exists x)A(x)}\ \exists\!:\!t$$

Here, as above, α and τ are metavariables standing for free variables and terms, respectively. That is to say, in an actual proof $A(\alpha)$ is replaced by a formula A containing a free variable a (in place of α), and $A(\tau)$ is replaced by a formula A containing a term t (in place of τ). The "eigenvariable condition" imposed on $\forall\!:\!\top$ is that the free variable a must not occur in the lower sequent. In the case of two-valued logic it is common to use a and t instead of α and τ in the first place. In the general case considered here it is necessary to explicitly introduce the metavariables α and τ in order to state the conditions on the rule in the above definition. Intuitively, what the condition means is that the statements "for all domain elements $d_1, \ldots, d_q$ in place of the eigenvariables $\alpha_1, \ldots, \alpha_q$, respectively, there are domain elements $e_1, \ldots, e_p$ in place of the terms $\tau_1, \ldots, \tau_p$, respectively, so that the sequents $\Delta_{\mathsf{Q}:i}(j)$ are all true" implies that "$(\mathsf{Q}x)A(x)$ takes the value v_i," and conversely, that "$(\mathsf{Q}x)A(x)$ takes the value v_i" implies that "there are domain elements $e_1, \ldots, e_p$ in place of the terms $\tau_1, \ldots, \tau_p$, respectively s.t. for all domain elements $d_1, \ldots, d_q$ in place of the eigenvariables $\alpha_1, \ldots, \alpha_q$, respectively, the sequents $\Delta_{\mathsf{Q}:i}(j)$ are all true," hold. In the case of two-valued logic this means that, for any interpretation, $(\forall x)A(x)$ is true iff for all domain elements d, $A(d)$ is true; $(\exists x)A(x)$ is true iff there is some domain element e s.t. $A(e)$ is true. The subtlety here is that it is required

that there is a domain element and not a *term* (which evaluates to that domain element) for $(\exists x)A(x)$ to be true. In an actual proof, however, τ is replaced by a term.

The truth function for a quantifier Q immediately yields introduction rules for place i in a way similar to the method described above for connectives: Let $I = \{j \subseteq \{v_1, \ldots, v_m\} \mid \widetilde{\mathsf{Q}}(j) \neq v_i\}$, then the rule is given as in Definition 3.4, with $\Delta_{\mathsf{Q}:i}(j)_{(l)} = \{A(\alpha_w^j) \mid w \in j, w \neq v_l\} \cup \{A(\tau^j) \mid v_l \in V \setminus \{j\}\}$. Again, we emphasize that in general this is not the only possible rule.

Example 3.2. Consider the universal quantor $\forall$ for Belnap's logic: $(\forall x)A(x)$ takes the value t only if for all $d \in D$ the instance $A(d)$ takes t, and false otherwise, i.e., if there is a $d \in D$ s.t. $A(d)$ takes a value among f, u, $\bot$. We obtain the following rules:

$$\frac{\Gamma, [f, u, \bot\colon A(\tau)]}{\Gamma, [f\colon (\forall x)A(x)]}\ \forall\!:\!f \qquad \frac{\Gamma, [t\colon A(\alpha)]}{\Gamma, [t\colon (\forall x)A(x)]}\ \forall\!:\!t$$

The remaining rules ($\forall$:u) and ($\forall$:$\bot$) are instances of weakening rules.

Similarly, we obtain rules for U:

$$\frac{\Gamma, [f\colon A(\tau)] \quad \Gamma, [f, u\colon A(\alpha)]}{\Gamma, [f\colon (\mathsf{U}x)A(x)]}\ \mathsf{U}\!:\!f \qquad \frac{\Gamma, [u\colon A(\alpha)]}{\Gamma, [u\colon (\mathsf{U}x)A(x)]}\ \mathsf{U}\!:\!u$$

$$\frac{\Gamma, [t\colon A(\tau)] \quad \Gamma, [t, u\colon A(\alpha)]}{\Gamma, [t\colon (\mathsf{U}x)A(x)]}\ \mathsf{U}\!:\!t \qquad \frac{\Gamma, [f, \bot\colon A(\tau)] \quad \Gamma, [t, \bot\colon A(\tau')]}{\Gamma, [\bot\colon (\mathsf{U}x)A(x)]}\ \mathsf{U}\!:\!\bot$$

The sequents denoted by Δ in the introduction rules for connectives and quantifiers are called *auxiliary sequents*.

Definition 3.5. A *sequent calculus* **LM** for a logic **L** is given by:

1. axiom schemas of the form: $[V\colon A]$,
2. for every connective $\Box$ and every truth value v_i an introduction rule $\Box$:i,
3. for every quantifier Q and every truth value v_i an introduction rule Q:i,
4. weakening rules for every place i:
$$\frac{\Gamma}{\Gamma, [i\colon A]}\ \mathrm{w}\!:\!i$$
5. contraction rules
$$\frac{\Gamma, [i\colon A, A]}{\Gamma, [i\colon A]}\ \mathrm{c}\!:\!i$$
6. exchange rules
$$\frac{\Gamma, [i\colon B, A], \Delta}{\Gamma, [i\colon A, B], \Delta}\ \mathrm{x}\!:\!i$$
7. cut rules for every two truth values $v_i \neq v_j$:
$$\frac{\Gamma, [i\colon A] \quad \Delta, [j\colon A]}{\Gamma, \Delta}\ \mathrm{cut}\!:\!ij$$

The rules (4)–(7) are called *structural rules*.

A sequent Γ is *provable* in a given sequent calculus, if there is an upward tree of sequents, rooted in Γ, s.t. every topmost sequent is an axiom and every other sequent is obtained from the ones standing immediately above it by an application of one of the rules.

Theorem 3.1. (Soundness) *If a sequent is provable in* **LM**, *then it is valid.*

P r o o f . By induction on the length of proofs: Axioms are obviously valid, since every formula has to take *some* truth value. Introduction rules preserve validity by definition. The weakening rules are obviously sound. The cut rules are sound, since no formula can take two different truth values. □

T h e o r e m 3.2. (Completeness) *If a sequent is valid, then it is provable in* **LM** *without cuts from atomic axioms.*

P r o o f . We use the method of *reduction trees*, due to Schütte [23] (see also [26, Ch. 1, § 8]). We show that every sequent S is either provable in the sequent calculus or has a counter-model.

Let E be an enumeration of all tuples of terms over $\mathcal{L}$. Call a free variable *available* at stage k iff it occurs in the tree constructed before stage k (if there is no such variable, pick any and call it available), and *new* otherwise. Call a p-tuple $\bar{t}$ of terms available for the reduction of F at place i with eigenvariables $a_1, \ldots, a_q$ at stage k on a branch B iff

1. $\bar{t}$ contains only variables which are available at stage k or are among $a_1, \ldots, a_q$, and either

2(a). $\bar{t}$ has not been used for a reduction of F at place i on B in a stage before k, or

2(b). the instance of the premise lying on B of a reduction of F at place i in a stage before k where $\bar{t}$ has been used did not contain any term variables.

A reduction tree is a tree of sequents constructed from Γ in stages as follows:

Stage 0: Write Γ at the root of the tree.

Stage k: If the topmost sequent Γ' of a branch contains an atomic formula A s.t. $A \in \bigcap_{j\in I} \Gamma'_{(j)}$ then stop the reduction for this branch. Call a branch *open* if it does not have this property.

Apply the following reduction steps for every formula F occurring at place i in the topmost sequent Γ' of an open branch, which has neither already been reduced at place i on this branch in this stage, nor is the result of a reduction at this stage:

1. $F \equiv \Box(A'_1, \ldots, A'_n)$: Replace Γ' in the reduction tree by:

$$\frac{\langle \Gamma', \Delta'_{\Box:i}(j)\rangle_{j\in I}}{\Gamma'}$$

   where $\Delta'_{\Box:i}(j)$ is an instance of $\Delta_{\Box:i}(j)$ in the rule $\Box{:}i$ introducing F as in Definition 3.3, obtained by instantiating $A_1, \ldots, A_n$ with $A'_1, \ldots, A'_n$, respectively.

2. $F \equiv (\mathsf{Q}x)A'(x)$: Let $\alpha_1, \ldots, \alpha_p$ be all eigenvariables and $\tau_1, \ldots, \tau_q$ be all term variables in the premises of the rule schema $\mathsf{Q}{:}i$. Replace Γ' in the reduction tree by:

$$\frac{\left\langle \Gamma', \Delta'_{\mathsf{Q}:i}(j)\right\rangle_{j\in I}}{\Gamma'}$$

   where $\Delta'_{\mathsf{Q}:i}(j)$ is an instance of $\Delta_{\mathsf{Q}:i}(j)$ in $\mathsf{Q}{:}i$ introducing F as in Definition 3.4, obtained by instantiating A with A', the eigenvariables $\alpha_1, \ldots, \alpha_p$ with the first p-tuple $a_1, \ldots, a_p$ of new free variables in the enumeration E, and the term variables $\tau_1, \ldots, \tau_q$ with the first available q-tuple $t_1, \ldots, t_q$ of terms in E containing variables which are either available or among $a_1, \ldots, a_p$, Observe that $F \in \Gamma'_{(i)}$ and thus F occurs in all upper sequents.

Now let T_S be the reduction tree constructed in this manner. If T_S is finite, then every topmost sequent contains an atomic formula that occurs at each place in that sequent. A

cut-free proof of Γ from axioms containing these formulas is easily constructed by inserting weakenings and exchanges.

If T_S is infinite it has an infinite branch B by König's Lemma. For every atomic formula $P(t_1, \ldots, t_n)$ in B, there is an $1 \leq l \leq m$ s.t. $P(t_1, \ldots, t_n)$ never occurs at place l in any sequents in this branch. We construct an interpretation $\mathbf{I}$ as follows: the domain is the set of terms, $\Phi_{\mathbf{I}}(t) = t$ (t a term) and $\Phi_{\mathbf{I}}\big(P(t_1, \ldots, t_n)\big) = v_l$, where v_l is the truth value corresponding to the place l.

If F is a formula occurring in B, and F occurs at place i anywhere in B, then $\mathrm{val}_{\mathbf{I}}(F) \neq v_i$. This is easily seen by induction on the structure of F: In particular, no formula in Γ evaluates to the truth value corresponding to the position at which it stands. Hence $\mathbf{I}$ does not satisfy Γ. □

We obtain as a consequence of the reduction tree construction the following

Proposition 3.1. *Let F be any formula. The sequent $[V\colon F]$ is cut-free provable from atomic axioms.*

4 Elimination of Cuts

The completeness theorem above shows that for every valid sequent there is a *cut-free* proof of that sequent. When analyzing given arbitrary proofs—as in our application in Section 6—one would like to have a method of stepwise transformation to a cut-free proof. This corresponds to extracting algorithmic content from proofs.

The proof of the cut-elimination theorem for the family **LM** of many-valued sequent calculi is analogous to the proof of the classical case given in [11]. It proceeds by moving the cuts in a given proof upwards and by reducing cuts to cuts on formulas of smaller complexity. The most important prerequisite for the proof is therefore the possibility to transform a cut on a composite formula to a derivation using only cuts acting on subformulas of the original cut-formula. In this section, we give a cut-elimination procedure modulo such a notion of reduction of cuts in the form of a parametric operator. In Section 5 we will show how to obtain such an operator.

Like in [11], we replace the cut rule by the obviously equivalent *mix* rule:

$$\frac{\Pi \qquad\qquad \Lambda}{\Pi^{i\setminus A}, \Lambda^{j\setminus A}}\ \mathrm{mix}{:}ij(A)$$

where A occurs in $\Pi_{(i)}$ and $\Lambda_{(j)}$, and $\Pi^{i\setminus A}$ ($\Lambda^{j\setminus A}$) is obtained from Π (Λ) by deleting every occurrence of A in $\Pi_{(i)}$ ($\Lambda_{(j)}$). Call the calculus obtained from **LM** by replacing the cut rule with the mix rule $\mathbf{LM}'$.

Definition 4.1. The *degree* of a mix:$ij(A)$ is the depth of the mix formula A. The *degree* of an $\mathbf{LM}'$-proof P having only one mix:$ij(A)$ as its last inference, denoted $d(P)$, is the degree of that mix.

We call a thread (cf. [26, p. 14]) in P containing the left (right) upper sequent of the mix a *left (right) thread*. The *rank* of a left (right) thread is the number of consecutive sequents counting upwards from the left (right) upper sequent of the mix which contain the mix formula at place i (j). The *left (right) rank of* P, denoted $r_l(P)$ ($r_r(P)$) is the maximum of the ranks of its left (right) threads. The *rank* of P, denoted $r(P)$, is the sum of its left and right rank: $r(P) = r_l(P) + r_r(P)$.

Definition 4.2. An $\mathbf{LM}'$-proof P is called *regular*, if every eigenvariable in P is the eigenvariable of only one quantifier introduction and occurs only in the subproof ending in that introduction.

In particular it cannot be the case in a regular proof that a free variable occurs as an eigenvariable in one inference and in a term which is replaced by a bound variable in another inference. It is easy to see that any proof can be transformed into a regular proof by renaming some eigenvariables.

The crucial part of the cut-elimination theorem is the reduction of mixes. From a proof ending in one mix we pass to a proof containing several mixes acting on formulas of lower complexity. The important step is when the principal formula of the mix is introduced immediately above the mix. To deal with this case, we introduce the notion of *mix reduction function* Red below. Using this function we obtain from the premises of the introduction rules immediately preceding the mix a deduction of the conclusion of the mix from these premises, but using only cuts on the formulas occurring in the premises, thus lowering the degree. By *applying* Red *to a mix* we mean the modification of the proof by replacing the two introductions and the mix by this deduction.

Definition 4.3. Let A be a formula with outermost logical symbol f, let $f{:}i$ and $f{:}j$ be the introduction rules for f at places i and j, and let $\Pi, \Delta_{f:i}(k)$ $(k \in I_i)$ and $\Lambda, \Delta_{f:j}(l)$ $(l \in I_j)$ be the premises of instances of $f{:}\,i$ and $f{:}\,j$ introducing A at places i and j, respectively. If f is a quantifier, then $f{:}\,i$ and $f{:}\,j$ have eigenvariables $a_1, \ldots, a_p$ and eliminate (i.e., replace by the bound variable) terms $t_1, \ldots, t_q$. Then $\mathrm{Red}(A; f{:}\,i; f{:}\,j) = R$ is an $\mathbf{LM}'$ derivation with the following properties:

1. The end sequent of R is the empty sequent,
2. Every topmost sequent of R is obtained from $\Delta_{f:i}(s)$ or $\Delta_{f:j}(r)$ by replacing some (including none) eigenvariables among $a_1, \ldots, a_p$ by terms among $t_1, \ldots, t_q$,
3. R contains only mixes of degree $< d(P)$.

The function Red is called a *mix reduction function*.

Remark 4.1. Note that Red eliminates only the outermost logical symbol of A, i.e., by applying Red once the degree of a mix is reduced by exactly one. It is also possible to work with a function Red which decreases the degree by more than one. When considering directly dependent rules, i.e., rules which can be replaced by a number of primitive rules, it is then evident how such an extension can be used to obtain cut-elimination procedures for the resulting calculi. It only requires some inspection of the proofs ending in the two introduction rules $f{:}\,i$ and $f{:}\,j$ and some manipulation to modify this extension so that mixes of degree d can be reduced in one step to mixes of degree $< d - 1$. This will be evident in the next section where we show how to obtain such a function Red.

Definition 4.4. Let R be an $\mathbf{LM}'$-derivation with end-sequent Γ and let Π be a sequent. Then we denote by Π, R the derivation obtained from R by adding Π to the left of every sequent in R, and by appending exchanges and contractions as necessary so that the end-sequent of Π, R is Γ.

We are now ready to state and prove the *Hauptsatz* for **LM**:

Theorem 4.1. *Cuts can be eliminated in* **LM***-proofs.*

For the reader not steeped in proof theory this statement seems curious. It seems that we have accomplished this result already (by the completeness theorem 3.2). A clarification is in order here: The phrase "cuts can be eliminated" has the status of a *terminus technicus*, and means much more than that for every proof there is one not using the cut rule. Roughly, it means that there is a primitive recursive procedure that transforms a given proof into a cut-free one, and moreover, this procedure is "local" in the sense that it eliminates one cut at a time (or in some variants a whole cluster of cuts).

Lemma 4.1. *Assume that from any proof P' which contains only one mix of degree $< d$ that mix can be eliminated. Then all mixes can be eliminated from any* $\mathbf{LM}'$*-proof P containing n mixes of degrees $< d$.*

Proof. By induction on the number n of mixes in P: For $n = 1$ this is the assumption. If $n > 1$, then let P' be some subproof of P which ends in a mix and contains only that mix. By assumption, the mix can be eliminated from P', yielding a proof P'' of the same end-sequent as P'. By replacing P' in P by P'' we obtain a proof with $n - 1$ mixes, all of degree $< d$, and the induction hypothesis applies. □

Lemma 4.2. *Let P be an* $\mathbf{LM}'$*-proof containing only one mix:$ij(A)$ as its last inference. Then P can be transformed to a mix-free* $\mathbf{LM}'$*-proof P of the same end-sequent.*

Proof. Let P be an $\mathbf{LM}'$-proof containing only one mix:$ij(A)$ which occurs as the last inference in P. W.l.o.g. we assume that P is regular. P is of the form:

$$\dfrac{\begin{array}{c}\vdots\ P_1\\ \Pi\end{array} \qquad \begin{array}{c}\vdots\ P_2\\ \Lambda\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{mix:}ij(A)$$

The proof is by double induction on the rank and degree of P:

1. $r(P) = 2$, i.e., left and right rank of P equal 1. We distinguish cases according to the type of the inferences immediately above the mix:
   (a) Π is an axiom $[V\colon A]$. P is of the form
   $$\dfrac{[V\colon A] \qquad \begin{array}{c}\vdots\ P_2\\ \Lambda\end{array}}{[V \setminus v_i\colon A], \Lambda^{j\backslash A}}\ \text{mix:}ij(A)$$
   We can derive $\Pi^{i\backslash A}, \Lambda jA$ without a mix as follows:
   $$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\ P_2\\ \Lambda\end{array}}{[j\colon A, \ldots, A], \Lambda^{j\backslash A}}\ \text{x}}{[j\colon A], \Lambda^{j\backslash A}}\ \text{c}}{[V \setminus v_i\colon A], \Lambda^{j\backslash A}}\ \text{w}$$
   (b) Λ is an axiom. Similarly.
   (c) Π is the conclusion of a structural inference. Since the left rank is 1, this inference must be a weakening introducing A at place i:
   $$\dfrac{\dfrac{\begin{array}{c}\vdots\ P_1\\ \Pi^{i\backslash A}\end{array}}{\Pi^{i\backslash A}, [i\colon A]}\ \text{w:}\ i \qquad \begin{array}{c}\vdots\ P_2\\ \Lambda\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{mix:}ij(A)$$
   where Π is $\Pi^{i\backslash A}, [i\colon A]$. We obtain $\Pi^{i\backslash A}, \Lambda^{j\backslash A}$ without a mix as follows:
   $$\dfrac{\dfrac{\begin{array}{c}\vdots\ P_1\\ \Pi^{i\backslash A}\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{w}}{}$$

(d) Λ is the conclusion of a structural inference. Similarly.

(e) Both Π and Λ are conclusions of introduction rules $\Box{:}i$ and $\Box{:}j$ for the connective $\Box$:

$$\dfrac{\left\langle \begin{array}{c}\vdots\, P_{ir} \\ \Pi, \Delta_{\Box:i}(r)\end{array}\right\rangle_r \qquad \left\langle \begin{array}{c}\vdots\, P_{js} \\ \Lambda, \Delta_{\Box:j}(s)\end{array}\right\rangle_s}{\dfrac{\Pi, [i\colon \Box(\cdots)] \qquad \Lambda, [j\colon \Box(\cdots)]}{\Pi, \Lambda}\ \mathrm{mix}{:}ij(\Box(\cdots))}$$

$R = \mathrm{Red}(A; \Box\colon i; \Box\colon j)$ is a derivation of the empty sequent with initial sequents among $\Delta_{\Box:i}(r)$ and $\Delta_{\Box:j}(s)$. Let P'_{ir} (P'_{js}) be P_{ir} (P_{js}) with weakenings and exchanges appended so that the end sequent equals $\Pi, \Lambda, \Delta_{\Box:i}(r)$ ($\Pi, \Lambda, \Delta_{\Box:j}(s)$). Let P' be the proof obtained from Π, Λ, R by writing P'_{ir} (P'_{js}) above any topmost sequent of the form $\Pi, \Lambda, \Delta_{\Box:i}(r)$ ($\Pi, \Lambda, \Delta_{\Box:j}(s)$). By the induction hypothesis and Lemma 4.1 the mixes can be eliminated from P'.

(f) Both Π and Λ are conclusions of introduction rules $\mathsf{Q}{:}i$ and $\mathsf{Q}{:}j$ for the quantifier Q:

$$\dfrac{\left\langle \begin{array}{c}\vdots\, P_{ir} \\ \Pi, \Delta_{\mathsf{Q}:i}(r)\end{array}\right\rangle_r \qquad \left\langle \begin{array}{c}\vdots\, P_{js} \\ \Lambda, \Delta_{\mathsf{Q}:j}(s)\end{array}\right\rangle_s}{\dfrac{\Pi, [i\colon (\mathsf{Q}x)A(x)] \qquad \Lambda, [j\colon (\mathsf{Q}x)A(x)]}{\Pi, \Lambda}\ \mathrm{mix}{:}ij(\mathsf{Q}x)A(x))}$$

$R = \mathrm{Red}(A; \mathsf{Q}\colon i; \mathsf{Q}\colon j)$ is a derivation of the empty sequent. An initial sequent in R is of the form $\Delta_{\mathsf{Q}:i}(r)\sigma$ or $\Delta_{\mathsf{Q}:j}(s)\sigma$, where σ is a substitution mapping some eigenvariables to terms. Let P'_{ir} (P'_{js}) be P_{ir} (P_{js}) with weakenings and exchanges appended so that the end sequent equals $\Pi, \Lambda, \Delta_{\mathsf{Q}:i}(r)$ ($\Pi, \Lambda, \Delta_{\mathsf{Q}:j}(s)$). Let P' be the proof obtained from Π, Λ, R by writing $P'_{ir}\sigma$ ($P'_{js}\sigma$) above any topmost sequent of the form $\Pi, \Lambda, \Delta_{\mathsf{Q}:i}(r)\sigma$ ($\Pi, \Lambda, \Delta_{\mathsf{Q}:j}(s)\sigma$). P' is indeed a proof since (a) Π and Λ cannot contain any eigenvariables and (b) the terms in $\mathrm{ran}\sigma$ do not contain eigenvariables of any quantifier introductions in the P_{ir} or P_{js}. By the induction hypothesis and Lemma 4.1 the mixes can be eliminated from P'.

2. $r_r(P) > 1$: Again, we distinguish cases:

(a) $\Lambda_{(i)}$ contains A. We obtain the following mix-free proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots\, P_1}{\Pi}}{\Pi^{i\setminus A}, [i\colon A, \ldots, A]}\,\mathrm{e}}{\Pi^{i\setminus A}, [i\colon A]}\,\mathrm{c}}{\Pi^{i\setminus A}, \Lambda^{j\setminus A}}\,\mathrm{w}$$

(b) $\Pi_{(j)}$ contains A. Similarly.

(c) Λ is the consequence of an inference J_2, which is either structural (but not a mix), or a logical inference (not introducing A at place j). P is of the form

$$\dfrac{\begin{array}{c}\vdots\, \hat{P} \\ \Pi\end{array} \qquad \dfrac{\begin{array}{ccc}\vdots\, P_1 & & \vdots\, P_p \\ \Psi_1 & \ldots & \Psi_p\end{array}}{\Lambda}\,J_2}{\Pi^{i\setminus A}, \Lambda^{j\setminus A}}\,\mathrm{mix}{:}ij(A)$$

Let $j_1, \ldots, j_s$, $1 \leq j_k \leq p$, be all indices s.t. Ψ_{j_k} contains A. (There is at least one such j_k, otherwise the right rank of P would equal 1). Consider the proofs P'_{j_k}:

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Pi & \Psi_{j_k}\end{array}}{\Pi^{i\backslash A}, {\Psi_{j_k}}^{j\backslash A}}\ \text{mix:}ij(A)$$

In P'_{j_k}, $r_l(P'_{j_k}) = r_l(P)$ and $r_r(P'_{j_k}) \leq r_r(P)-1$, and in sum $r(P'_{j_k}) \leq r(P)-1$. Hence the induction hypothesis applies and we have mix-free proofs P''_{j_k} of $\Pi^{i\backslash A}, {\Psi_{j_k}}^{j\backslash A}$. For indices l not occurring in the above list, we have that Ψ_l equals ${\Psi_l}^{j\backslash A}$, and we define P''_l as

$$\dfrac{\begin{array}{c}\vdots\ P_l \\ \Psi_l\end{array}}{\Pi^{i\backslash A}, \Psi_l}\ \text{w}$$

If J_2 is a weakening at place j (and consequently, $p = 1$ and ${\Psi_1}^{j\backslash A} = \Lambda^{j\backslash A}$), then P''_1 serves as our transformed proof. Otherwise, construct a proof as follows:

$$\dfrac{\dfrac{\begin{array}{ccc}\vdots\ P''_1 & & \vdots\ P''_p \\ \Pi^{i\backslash A}, {\Psi_1}^{j\backslash A} & \ldots & \Pi^{i\backslash A}, {\Psi_p}^{j\backslash A}\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ J_2}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{x}$$

(d) Λ is the consequence of a logical inference J_2 introducing A at place j. P is of the form

$$\dfrac{\begin{array}{cc}\vdots\ \hat{P} & \dfrac{\begin{array}{ccc}\vdots\ P_1 & & \vdots\ P_p \\ \Lambda, \Delta_1 & \ldots & \Lambda, \Delta_p\end{array}}{\Lambda, [j\colon A]}\ J_2 \\ \Pi & \end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{mix:}ij(A)$$

Consider the proofs P'_k, $1 \leq k \leq p$ (Note that Δ_k does not contain A—only proper subformulas of A—and hence ${\Delta_k}^{j\backslash A}$ equals Δ_k):

$$\dfrac{\begin{array}{cc}\vdots\ P' & \vdots\ P_k \\ \Pi & \Lambda, \Delta_k\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}, \Delta_k}\ (A, i, j)$$

In P_k, $r_l(P'_k) = r_l(P)$, $r_r(P'_k) \leq r_r(P) - 1$ and in sum $r(P'_k) \leq r(P) - 1$. Hence, the induction hypothesis applies and we obtain mix-free proofs $P(k)''$ of $\Pi^{i\backslash A}, \Lambda^{j\backslash A}, \Delta_k$. Construct a proof P' as follows:

$$\dfrac{\begin{array}{cc}\vdots\ \hat{P} & \dfrac{\begin{array}{ccc}\vdots\ P''_1 & & \vdots\ P''_p \\ \Pi^{i\backslash A}, \Lambda^{j\backslash A}, \Delta_1 & \ldots & \Pi^{i\backslash A}, \Lambda^{j\backslash A}, \Delta_p\end{array}}{\Pi^{i\backslash A}, \Lambda^{j\backslash A}, [j\colon A]}\ J_2 \\ \Pi & \end{array}}{\Pi^{i\backslash A}, \Pi^{i\backslash A}, \Lambda^{j\backslash A}}\ \text{mix:}ij(A)$$

Note that A does not occur at place j in $\Pi^{i\backslash A}$, since otherwise case 2.b would have applied, hence $r_r(P') = 1$. With $r_l(P') = r_l(P)$ we have that $r(P') < r(P)$ and the

induction hypothesis yields a mix-free proof P'' of $\Pi^{i\setminus A}, \Pi^{i\setminus A}, \Lambda^{j\setminus A}$. We obtain a mix-free proof:

$$\dfrac{\begin{array}{c}\vdots\ P'\\ \Pi^{i\setminus A}, \Pi^{i\setminus A}, \Lambda^{j\setminus A}\end{array}}{\Pi^{i\setminus A}, \Lambda^{j\setminus A}}\ \text{xc}$$

3. $r_r(P) = 1$ and $r_l(P) > 1$: This case is dealt with in the same way as 2. above, *mutatis mutandis.*

This completes the proof of the lemma (and the cut-elimination theorem). □

5 Resolution-based Cut Reduction Operators

In this section we give a definition of one of many possible cut reduction functions Red for the Hauptsatz. Our approach uses many-valued resolution. It is based on the observation that the soundness of the cut rule consists in the fact that a formula A cannot take two different truth values at the same time. Clauses are a convenient metalogical notation for expressing such negative statements, while sequents are not. There are, however, close relationships between clause and sequent calculi and in particular the resolution rule and the cut rule. It is not surprising, then, that resolution deductions can be translated to sequent calculus derivation using only cuts (or, as in our case, mixes).

Many-valued resolution [1] is a straightforward extension of resolution for classical first-order logic. It is based on clause syntax: A clause is a set of literals, a literal is an atomic formula together with a truth value index. In usual resolution, the negation sign plays the rôle of the truth value index: $\neg A$ means "A is false", A without negation sign means "A is true". In the many-valued case we have literals of the form A^w with the interpretation "A takes the value w."

The free variables in a clause C are understood to be "universally quantified". More precisely, a structure $\mathcal{M}$ *universally satisfies* a clause C iff for all assignments s the interpretation $\langle \mathcal{M}, s\rangle$ satisfies C (in the sense of Definition 3.2). $\mathcal{M}$ universally satisfies a set of clauses $\mathcal{C}$ iff $\mathcal{M}$ universally satisfies each clause $C \in \mathcal{C}$. If no $\mathcal{M}$ universally satisfies $\mathcal{C}$, then $\mathcal{C}$ is called *universally unsatisfiable.* The empty clause $\emptyset$ is, of course, universally unsatisfiable.

In the following, we assume familiarity with basic resolution terminology.

Definition 5.1. Let C_1 and C_2 be variable disjoint clauses and let $D_1 = \{A_1^v, \ldots, A_p^v\} \subseteq C_1$ and $D_2 = \{B_1^w, \ldots, B_q^w\} \subseteq C_2$ be subsets of C_1 and C_2 where $v \neq w$. If $\{A_1, \ldots, A_p, B_1, \ldots, B_q\}$ is unifiable with most general unifier (mgu) σ, then $R = (C_1 \setminus D_1)\sigma \cup (C_2 \setminus D_2)\sigma$ is called a *resolvent* of C_1 and C_2.

A (tree-like) resolution deduction from a set of clauses $\mathcal{C}$ is an upward tree of clauses, where every clause either results from a clause in $\mathcal{C}$ by renaming of variables iff it is a topmost clause, or otherwise is a resolvent of the two clauses immediately above it. We write $\mathcal{C} \vdash D$ if there is a resolution deduction from $\mathcal{C}$ whose bottom clause is D.

Theorem 5.1. ([1, 27]) *$\mathcal{C} \vdash \emptyset$ iff $\mathcal{C}$ is universally unsatisfiable.*

Definition 5.2. Let Γ be a sequent consisting of atomic formulas only. Then $\mathrm{cl}(\Gamma)$ is defined as the clause

$$\mathrm{cl}(\Gamma) = \{A^{v_i} \mid A \text{ occurs in } \Gamma_{(i)}\}$$

Conversely, if C is a clause then the sequent $\mathrm{seq}(C)$ is defined by

$$\mathrm{seq}(C)_{(i)} = A_1, \ldots, A_p$$

if $A_1^{v_i}, \ldots, A_p^{v_i}$ are all literals with index v_i in C. (We assume some arbitrary but fixed ordering of atoms.)

Clearly, $\mathrm{seq}(\mathrm{cl}(\Gamma))$ is Γ up to exchanges and contractions.

The mix reduction function Red takes as arguments a formula A with outermost symbol f and two rules $f\colon i$ and $f\colon j$ introducing f at places i and j. We shall now define one possible reduction function by showing how, for any two introduction rules, a schematic derivation of the empty sequent from the premises of the rules can be obtained. For any particular formula and rule applications the corresponding instance of the derivation is immediately obtained by matching formulas to schematic letters, terms to metavariables for terms, and variables to metavariables for eigenvariables.

Consider the rule schemata

$$\frac{\langle \Delta_{\Box:i}(r)\rangle_{r\in I_i}}{[i\colon \Box(A_1,\ldots,A_n)]} \qquad \frac{\langle \Delta_{\Box:j}(s)\rangle_{s\in I_j}}{[j\colon \Box(A_1,\ldots,A_n)]}$$

Let $\Delta'_{\Box:i}(r)$ ($\Delta'_{\Box:j}(s)$) be obtained from $\Delta_{\Box:i}(r)$ ($\Delta_{\Box:j}(s)$) by replacing the schematic formula A_i by an atomic formula P_{A_i}. Let $\mathcal{C}$ be the set of clauses

$$\mathcal{C} = \bigcup_{r\in I_i} \{\mathrm{cl}(\Delta'_{\Box:i}(r)\} \cup \bigcup_{s\in I_j} \{\mathrm{cl}(\Delta'_{\Box:j}(r)\}$$

$\mathcal{C}$ is clearly universally unsatisfiable, since (the "conjunction" of) $\Delta'_{\Box:i}(r)$ is equivalent to $\Box(\cdot)^{v_i}$ and (the "conjunction" of) $\Delta'_{\Box:j}(s)$ is equivalent to $\Box(\cdot)^{v_j}$, where $i \neq j$.

By the completeness of many-valued resolution there is a resolution deduction R of $\emptyset$ from $\mathcal{C}$. Note that the literals in $\mathcal{C}$ are all variable-free, and hence, in every resolution step, there is only one literal that is resolved upon. R can be translated to an $\mathbf{LM}'$ deduction $\mathrm{seq}(R)$ of the empty sequent. We argue by induction:

If R consists only of an initial clause $\mathrm{cl}(\Delta')$, then $\mathrm{seq}(R)$ is the sequent $\mathrm{seq}(\mathrm{cl}(\Delta'))$.

Otherwise, let

$$\dfrac{\begin{array}{cc}\vdots\ R_1 & \vdots\ R_2\\ C_1 \cup \{P_{A_i}^{v_k}\} & C_2 \cup \{P_{A_i}^{v_l}\}\end{array}}{C_1 \cup C_2}$$

be the last resolution step in R. We obtain $\mathrm{seq}(R)$ as follows:

$$\dfrac{\begin{array}{cc}\vdots\ \mathrm{seq}(R_1) & \vdots\ \mathrm{seq}(R_2)\\ \mathrm{seq}(C_1 \cup \{P_{A_i}^{v_k}\}) & \mathrm{seq}(C_2 \cup \{P_{A_i}^{v_l}\})\end{array}}{\mathrm{seq}(C_1), \mathrm{seq}(C_2)}\ \mathrm{mix}{:}ij(P_{A_i})$$

We can now take as $\mathrm{Red}(A; \Box\colon i; \Box\colon j)$ the derivation P where P is obtained from $\mathrm{seq}(R)$ by (a) replacing P_{A_i} by A_i and (b) adding (if necessary) derivations of $\mathrm{seq}(\mathrm{cl}(\Delta))$ from Δ above the initial sequents (these require only exchanges).

Example 5.1. Consider the case of a mix on two formulas with outermost logical symbol $\vee$ at places f and $\bot$. The clause translation of the premises is

$$\mathcal{C} = \{\{P_A^f\}, \{P_B^f\}, \{P_A^f, P_A^\bot\}, \{P_B^f, P_B^\bot\}, \{P_A^\bot, P_B^\bot\}\}$$

A possible resolution refutation is

$$\dfrac{\{P_B^f\} \qquad \dfrac{\{P_A^f\} \quad \{P_A^\bot, P_B^\bot\}}{\{P_B^\bot\}}}{\emptyset}$$

From this we obtain the following deduction schema:

$$\dfrac{[f\colon B] \qquad \dfrac{[f\colon A] \quad [\bot\colon A, B]}{[\bot\colon B]}\ \mathrm{mix}{:}f\bot(A)}{\emptyset}\ \mathrm{mix}{:}f\bot(B)$$

For the quantifiers, consider the rule schemata

$$\frac{\langle \Delta_{\mathsf{Q}:i}(r)\rangle_{r\in I_i}}{[i\colon (\mathsf{Q}x)A(x)]} \qquad \frac{\langle \Delta_{\mathsf{Q}:j}(s)\rangle_{s\in I_j}}{[j\colon (\mathsf{Q}x)A(x)]}$$

Let $\Delta'_{\mathsf{Q}:i}(r)$ ($\Delta'_{\mathsf{Q}:j}(s)$) be obtained from $\Delta_{\mathsf{Q}:i}(r)$ ($\Delta_{\mathsf{Q}:j}(s)$) by replacing the schematic formula $A(\alpha)$ by an atomic formula $P(x_\alpha)$, and every formula $A(\tau)$ by $P(c_\tau)$. Let $\mathcal{C}$ be the set of clauses

$$\mathcal{C} = \bigcup_{r\in I_i} \{\mathrm{cl}(\Delta'_{\mathsf{Q}:i}(r)\} \cup \bigcup_{s\in I_j} \{\mathrm{cl}(\Delta'_{\mathsf{Q}:j}(r)\}$$

Again, $\mathcal{C}$ is universally unsatisfiable. Let R be a resolution deduction of $\emptyset$ from $\mathcal{C}$. Since R is in tree form, the cumulative substitution ρ of R can be defined as usual. The initial clauses of R may be renamings of clauses in $\mathcal{C}$. We choose these renamings so that a variable x_α is renamed as x^h_α. We recursively translate $R\rho$ into an $\mathbf{LM}'$-derivation as follows:

If $R\rho$ only consists of an initial clause $\mathrm{cl}(\Delta')$, then $\mathrm{seq}(R\rho)$ is the sequent $\mathrm{seq}(\mathrm{cl}(\Delta'\lambda\rho))$ (λ is some renaming substitution).

Otherwise, let

$$\dfrac{\begin{array}{cc}\vdots\ R_1 & \vdots\ R_2\\ C_1 \cup \{L^{v_l}\} & C_2 \cup \{L^{v_k}\}\end{array}}{C_1 \cup C_2}$$

be the last resolution step in $R\rho$, where L is an atom of the form $P(x)$ or $P(c)$ (x is a variable and c a constant symbol). Recall that the cumulative substitution has been applied to R, so resolution steps actually do take this special form with only one literal resolved upon. We obtain $\mathrm{seq}(R\rho)$ as follows:

$$\dfrac{\begin{array}{cc}\vdots\ \mathrm{seq}(R_1) & \vdots\ \mathrm{seq}(R_2)\\ \mathrm{seq}(C_1 \cup \{L^{v_k}\}) & \mathrm{seq}(C_2 \cup \{L^{v_l}\})\end{array}}{\mathrm{seq}(C_1), \mathrm{seq}(C_2)}\ \mathrm{mix}{:}ij(L)$$

The substitution ρ replaces variables of the form x^h_α by constants c_τ. take as $\mathrm{Red}(A; \mathsf{Q}\colon i; \mathsf{Q}\colon j)$ the derivation P where P is obtained from $\mathrm{seq}(R\rho)$ by (a) replacing $P(x^h_\alpha)$ resp. $P(c_\tau)$ by $A(\alpha)$ resp. $A(\tau)$ and (b) adding (if necessary) derivations of $\mathrm{seq}(\mathrm{cl}(\Delta))\lambda\pi$ from $\Delta\lambda\pi$ above the initial sequents (these require only exchanges and contractions).

Example 5.2. Consider the case of a mix on two formulas with outermost logical symbol U at places u and $\bot$. The clause translation of the premises is

$$\mathcal{C} = \big\{\{P(x_\alpha)^u\}, \{P(c_\tau)^f, P(c_\tau)^\bot\}, \{P(c_{\tau'})^t, P(c_{\tau'})^\bot\}\big\}$$

One of the possible resolution refutations is

$$\dfrac{\{P(x'_\alpha)^u\} \qquad \dfrac{\{P(x_\alpha)^u\} \quad \{P(c_\tau)^f, P(c_\tau)^\bot\}}{\{P(c_\tau)^\bot\}}\ c_\tau/x_\alpha}{\emptyset}\ c_\tau/x'_\alpha$$

Note that two copies of the clause $\{P(x_\alpha)^u\}$ have been used. From this we obtain the following deduction schema:

$$\dfrac{[u\colon A(\tau/\alpha)] \qquad \dfrac{[u\colon A(\tau/\alpha)] \quad [f,\bot\colon A(\tau)]}{[\bot\colon A(\tau)]}\ \text{mix:}uf(A(\tau))}{\emptyset}\ \text{mix:}u\bot(A(\tau))$$

Similar considerations could also be used to obtain "macro-reductions" of composite formulas of depth > 1 in one step. This may lead to an algorithmic simplification of cut-elimination if the structure of the cut formula is of a specific type, e.g., monadic.

6 Reasoning about Knowledge in Belnap's Logic

A consequence of the cut-elimination theorem is Gentzen's "Verschärfter Hauptsatz", also known as

T h e o r e m 6.1. (Midsequent Theorem) *Let Π be a sequent consisting only of prenex formulas. Any proof of Π can be effectively transformed to a cut-free proof of Π containing sequents $\Sigma_1, \ldots, \Sigma_p$, s.t., for all $1 \leq j \leq p$,*

1. *Σ_j is quantifier-free,*
2. *every inference above Σ_j is either structural or propositional, and*
3. *every inference below Σ_j is either structural or a quantifier inference.*

P r o o f. By the cut-elimination theorem and Proposition 3.1, the given proof can be transformed to a cut-free proof P of Π from atomic axiom sequents. The *order* of a quantifier introduction $(\mathsf{Q}{:}i)$ in P is defined as the number of propositional inferences below $(\mathsf{Q}{:}i)$. The *order* $o(P)$ of P is the sum of the orders of all quantifier inferences occurring in P.

We prove the theorem by induction on the order of P:

1. $o(P) = 0$: There is no propositional inference occurring below any quantifier inferences. Let B be a branch in P. Let Σ_B be the conclusion of the lowermost propositional inference on B, or the (atomic) axiom sequent on B if B does not contain any propositional inferences. If Σ_B contains a quantified formula F, then F is introduced by weakenings. To see this, recall that Π contains only prenex formulas: By the subformula property, F is a subformula of a formula in Π, hence no propositional inferences apply to it. Eliminate F and all inferences applying to it from the part of B above Σ_B, and add appropriate weakenings and exchanges directly below Σ_B. After the above procedure has been applied to all branches B in P, the (finite) set of all Σ_B serves as the set of Σ_j in the statement of the theorem.
2. $o(P) > 0$: Then there is a quantifier inference $(\mathsf{Q}{:}i)$ with the following property: The topmost logical inference below $(\mathsf{Q}{:}i)$ is a propositional inference, say $(\Box{:}j)$. The part of P between $(\mathsf{Q}{:}i)$ and $(\Box{:}j)$ takes the following form:

$$\dfrac{\dfrac{\left\langle \begin{array}{c}\vdots\\ \Gamma', \Delta_{\mathsf{Q}:i}(k)\end{array}\right\rangle_k}{\Gamma', [i\colon (\mathsf{Q}x)A(x)]}\ \mathsf{Q}{:}i \qquad \begin{array}{c}\vdots\\ \Gamma_1\end{array} \quad \ldots \quad \begin{array}{c}\vdots\\ \Gamma''\end{array} \quad \ldots \quad \begin{array}{c}\vdots\\ \Gamma_n\end{array}}{\Gamma}\ \Box{:}j$$

where the part denoted by $*$ contains only structural inferences, and Γ and Γ'' contain $(\mathsf{Q}x)A(x)$ as a sequent-formula. We can now lower the order by exchanging the positions of $(\mathsf{Q}{:}i)$ and $(\Box{:}j)$:

$$\dfrac{\left\langle \dfrac{\dfrac{\vdots}{\Gamma_1}}{\dfrac{\Delta_{\mathsf{Q}:i}(k),\Gamma_1}{}} \quad \dots \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Gamma',\Delta_{\mathsf{Q}:i}(k)}}{\Gamma',\Delta_{\mathsf{Q}:i}(k),[i{:}\,(\mathsf{Q}x)A(x)]}\,\mathrm{w}}{\Delta_{\mathsf{Q}:i}(k),\Gamma',[i{:}\,(\mathsf{Q}x)A(x)]}\,\mathrm{e}}{\dfrac{\vdots\,*}{\Delta_{\mathsf{Q}:i}(k),\Gamma''}} \quad \dots \quad \dfrac{\dfrac{\vdots}{\Gamma_n}}{\Delta_{\mathsf{Q}:i}(k),\Gamma_n}}{\dfrac{\Delta,\Gamma}{\Gamma,\Delta_{\mathsf{Q}:i}(k)}\,\mathrm{e}}\,\Box{:}j \right\rangle_k}{\dfrac{\Gamma,[i{:}\,(\mathsf{Q}x)A(x)]}{\Gamma}\,e,c}\,\mathsf{Q}{:}i$$

□

Similar considerations can also be used for analyzing natural deduction proofs in many-valued logic as introduced in [5].

In the following example, we consider a representation of knowledge about (possibly inconsistent and incomplete) sets E of facts, using Belnap's logic enriched by the quantifiers $\forall$ and U. The facts are simply Boolean ground literals. These facts may be contradictory in the sense that both P and $\neg P$ are present. Furthermore, it may be the case that for some ground atom A of the language, neither A nor $\neg A$ is contained in the facts base. The facts may be thought of as *evidences* for certain state of affairs. They might, e.g., be protocols of certain experiments.

The *epistemic state* s of a set of facts is the four-valued interpretation of *all* ground atoms induced by the facts base E: if both A and $\neg A$ are facts, then $s(A) = \bot$, if neither A nor $\neg A$ is a fact then $s(A) = u$, if only A ($\neg A$) is a fact then $s(A) = t$ (f).

The *meta-knowledge* is a set K of universal true sentences $(\forall\bar{x})B_1(\bar{x})$, ..., $(\forall\bar{x})B_q(\bar{x})$ of Belnap's logic. The meta-knowledge is assumed to be consistent with all possible epistemic states. In cases where the facts base is too large or unwieldy to work with, one might use instead the meta-knowledge to reason about the facts. For instance, we might deduce from K that the sentence $(\mathsf{U}x)A(x)$ takes the truth value $\bot$, expressing that the instances of $A(x)$ are neither uniformly true nor uniformly false. This can be formalized in the sequent calculus by proving the sequent $[f{:}\,(\forall\bar{x})B_1(\bar{x}),\dots,(\forall\bar{x})B_q(\bar{x})],[\bot{:}\,(\mathsf{U}x)A(x)]$. A proof of this sequent can, by virtue of an analog of Herbrand's theorem, be analyzed to yield a finite sequence of critical pairs $\langle A(t_1),A(t'_1)\rangle$, ..., $\langle A(t_p),A(t'_p)\rangle$ s.t. for every epistemic state under consideration, $A(t_i)$ and $A(t'_i)$ will take different truth values for some $1 \le i \le p$.

Theorem 6.2. *Let P be a proof of*

$$[f{:}\,(\forall x)B_1(x),\dots,(\forall x)B_q(x)],[\bot{:}\,(\mathsf{U}x)A(x)]$$

Then P can be transformed to proofs of

$$[f{:}\,(\forall x)B_1(x),\dots,(\forall x)B_q(x)],\quad [w_1{:}\,A(t_1^{w_1})],[\bot{:}\,A(t_1^{w_1})],\dots,[w_r{:}\,A(t_r^{w_r})],[\bot{:}\,A(t_r^{w_r})]$$

for $\langle w_1,\dots,w_r\rangle \in \{t,f\}^r$. (The critical pairs then are $\langle A(t_1^t),A(t_1^f)\rangle$, ..., $\langle A(t_r^t),A(t_r^f)\rangle$.)

P r o o f. Apply the midsequent theorem to P. None of the quantifier inferences in the proof have eigenvariable conditions (cf. Example 3.2). Let $\Sigma_1, \ldots, \Sigma_p$ be the midsequents of this transformed proof. From each of these midsequents a sequent of the above form can be derived, possibly using weakenings. Observe that $(\forall{:}f)$ takes only one premise, hence one midsequent is sufficient in each case. □

7 Conclusion

We have introduced a general approach to obtaining sequent calculi for finite-valued first-order logics. These sequent calculi enjoy cut-elimination and midsequent theorems. In special cases, the proof-theoretic properties can be exploited to obtain more specific results about the particular logics at hand. We illustrated this by exhibiting a version of Herbrand's theorem for a first-order extension of Belnap's four-valued logic suitable for reasoning about possibly inconsistent and incomplete knowledge bases. Similar results by proof-theoretic analysis such as interpolants constructed in Maehara's lemma (see [26, Ch. 1]) may be used for further classification of inconsistencies.

References

[1] M. Baaz and C. G. Fermüller. Resolution for many-valued logics. In A. Voronkov, editor, *Logic Programming and Automated Reasoning. Proceedings LPAR'92*, LNAI 624, pp. 107–118, Berlin, 1992. Springer.

[2] M. Baaz and C. G. Fermüller. Resolution-based theorem proving for many-valued logics. submitted, 1993.

[3] M. Baaz, C. G. Fermüller, A. Ovrutcki, and R. Zach. MULTLOG: A system for axiomatizing many-valued logics. In A. Voronkov, editor, *Logic Programming and Automated Reasoning. Proceedings LPAR'93*, LNAI 698, pp. 345–347, Berlin, 1993. Springer.

[4] M. Baaz, C. G. Fermüller, and R. Zach. Dual systems of sequents and tableaux for many-valued logics. *Bull. EATCS*, 51:192–197, 1993. paper read at *2nd Workshop on Tableau-based Deduction*, Marseille, April 1993.

[5] M. Baaz, C. G. Fermüller, and R. Zach. Systematic construction of natural deduction systems for many-valued logics. In *Proc. 23rd International Symposium on Multiple-valued Logic*, pp. 208–213, Sacramento, CA, May 24–27, 1993. IEEE Press.

[6] N. D. Belnap. A useful four-valued logic. In J. M. Dunn and G. Epstein, editors, *Modern Uses of Multiple-valued Logic*, pp. 9–37. Reidel, 1975.

[7] W. A. Carnielli. Systematization of finite many-valued logics through the method of tableaux. *J. Symbolic Logic*, 52(2):473–493, 1987.

[8] W. A. Carnielli. On sequents and tableaux for many-valued logics. *J. Non-Classical Logic*, 8(1):59–76, 1991.

[9] M. Fitting. Bilattices in logic programming. In *Proc. 20th International Symposium on Multiple-valued Logic*, pp. 238–246, Charlotte, NC, May 23–25, 1990. IEEE Press.

[10] M. Fitting. Bilattices and the semantics of logic programming. *J. Logic Programming*, 11(1 & 2):91–116, July 1991.

[11] G. Gentzen. Untersuchungen über das logische Schließen I–II. *Math. Z.*, 39:176–210, 405–431, 1934.

[12] M. L. Ginsberg. Multivalued logics: A uniform approach to reasoning in artificial intelligence. *Comput. Intell.*, 4:265–316, 1988.

[13] R. Hähnle. Uniform notation of tableaux rules for multiple-valued logics. In *Proc. 21st International Symposium on Multiple-valued Logic*, pp. 238–245, 1991.

[14] R. Hähnle. *Tableaux-Based Theorem Proving in Multiple-Valued Logics*. PhD thesis, Universität Karlsruhe, Institut für Logik, Komplexität und Deduktionssysteme, May 1992.
[15] R. Hähnle. *Automated Deduction in Multiple-Valued Logics*. Oxford University Press, Oxford, 1993.
[16] R. Hähnle and W. Kernig. Verification of switch level designs with many-valued logic. In A. Voronkov, editor, *Logic Programming and Automated Reasoning. Proceedings LPAR'93*, LNAI 698, pp. 158–169, Berlin, 1993. Springer.
[17] Y. Kaluzhny and A. Y. Muravitsky. A knowledge representation based on the Belnap's four-valued logic. *J. Applied Non-Classical Logics*, 1993. to appear.
[18] A. Mostowski. The Hilbert epsilon function in many-valued logics. *Acta Philos. Fenn.*, 16:169–188, 1963.
[19] P. O'Hearn and Z. Stachniak. A resolution framework for finitely-valued first-order logics. *J. Symbolic Computation*, 13:235–254, 1992.
[20] J. B. Rosser and A. R. Turquette. *Many-Valued Logics*. Studies in Logic. North-Holland, Amsterdam, 1952.
[21] G. Rousseau. Sequents in many valued logic I. *Fund. Math.*, 60:23–33, 1967.
[22] K. Schröter. Methoden zur Axiomatisierung beliebiger Aussagen- und Prädikatenkalküle. *Z. Math. Logik Grundlag. Math.*, 1:241–251, 1955.
[23] K. Schütte. Ein System des verknüpfenden Schließens. *Arch. Math. Logik Grundlag.*, 2:55–67, 1956.
[24] S. J. Surma. An algorithm for axiomatizing every finite logic. In D. C. Rine, editor, *Computer Science and Multiple-valued Logic: Theory and Applications*, pages 137–143. North-Holland, Amsterdam, 1977.
[25] M. Takahashi. Many-valued logics of extended Gentzen style I. *Sci. Rep. Tokyo Kyoiku Daigaku Sect A*, 9:271, 1967.
[26] G. Takeuti. *Proof Theory*. Studies in Logic 81. North-Holland, Amsterdam, 2nd edition, 1987.
[27] R. Zach. *Proof Theory of Finite-valued Logics*. Diplomarbeit, Technische Universität Wien, 1993.

Zusammenfassung: Es wird eine uniforme Konstruktion von Sequentialkalkülen für endlichwertige Logiken erster Stufe mit sog. Distributionsquantoren vorgestellt. Für diese Kalküle werden Vollständigkeit, Schnittelimination und Mittelsequenzsätze gezeigt. Als eine Anwendung wird ein Analogon zum Herbrand'schen Satz für die vierwertige Wissensrepräsentationslogik von Belnap und Ginsberg abgeleitet. Dieser Satz kann zum Schließen mit unvollständigen und inkonsistenten Daten verwendet werden.

Authors' addresses:
Matthias Baaz
Technische Universität Wien
Institut für Algebra und diskrete Mathematik E118.2
Wiedner Hauptstraße 8–10, A-1040 Wien, Austria
baaz@logic.tuwien.ac.at

Christian G. Fermüller
Richard Zach
Technische Universität Wien
Institut für Computersprachen E185.2
Resselgasse 3/1, A-1040 Wien, Austria
chrisf@logic.tuwien.ac.at
zach@logic.tuwien.ac.at

Erratum

The introduction rules for $\wedge{:}f$ and $\vee{:}t$ given in Example 3.1 on page 338 are erroneous. The correct partial normal form for $\vee$ is:

$(A \vee B)^t$ iff (A^t or A^u or $A^\perp$ or B^t) and (A^t or A^u or B^t or B^u) and (A^t or $A^\perp$ or B^t or $B^\perp$) and (A^t or B^t or B^u or $B^\perp$)

The correct rules are:

$$\frac{\Gamma, [u, \perp, t{:}\, A], [t{:}\, B] \quad \Gamma, [\perp, t{:}\, A, B] \quad \Gamma, [u, t{:}\, A, B] \quad \Gamma, [u, \perp, t{:}\, B], [t{:}\, A]}{\Gamma, [t{:}\, A \vee B]} \vee{:}t$$

$$\frac{\Gamma, [u, \perp, f{:}\, A], [f{:}\, B] \quad \Gamma, [\perp, f{:}\, A, B] \quad \Gamma, [u, f{:}\, A, B] \quad \Gamma, [u, \perp, f{:}\, B], [f{:}\, A]}{\Gamma, [f{:}\, A \vee B]} \wedge{:}f$$

We are indebted to E. Reznik and M. Ultlog for drawing our attention to this mistake.