Intuitionistic Linear Logic and Partial Correctness

Dexter Kozen* Jerzy Tiuryn†

December 30, 2000

Abstract

We formulate a Gentzen-style sequent calculus for partial correctness that subsumes propositional Hoare Logic. The system is a non-commutative Intuitionistic Linear Logic. We prove soundness and completeness over relational and trace-based models. As a corollary we obtain a complete sequent calculus for inclusion and equivalence of regular expressions.

1 Introduction

In formulating logics for program verification such as Hoare Logic (HL), Dynamic Logic (DL), or Kleene Algebra with Tests (KAT), it is tempting to treat tests and correctness assertions as a uniform syntactic category. This temptation is best resisted: although both are classes of assertions, they have quite different characteristics. *Tests* are local assertions whose truth is determined by the current state of execution. They are normally immediately

*Computer Science Department, Cornell University, Ithaca NY 14853-7501, USA. Email: `kozen@cs.cornell.edu`

†Institute of Informatics, Warsaw University, ul. Banacha 2, 02-097 Warsaw, Poland. Email: `tiuryn@mimuw.edu.pl`

decidable. The assertion $x \geq 0$, where x is a program variable, is an example of such a test. Tests occur in all modern programming languages as part of conditional expressions and looping constructs. *Correctness assertions*, on the other hand, are statements about the global behavior of a program, such as partial correctness or halting. They are typically much richer in expressive power than tests and undecidable in general.

DL does not distinguish between these two categories of assertions. The two are freely mixed, and both are treated classically. For this reason, the resulting system is unnecessarily complex for its purposes. The rich-test version of DL, in which one can convert an arbitrary correctness assertion to a test using the operator ?, is Π^1_1-complete. Even with systems that do make the distinction, such as KAT, care must be taken not to inadvertently treat global properties as local; doing so can lead to anomalies such as the Dead Variable Paradox [10].

One major distinguishing factor between tests and correctness assertions that may not be immediately apparent is that the former are classical in nature, whereas the latter are intuitionistic. For example, the DL axiom

$$[p]\,[q]\,b \quad\equiv\quad [p\,;\,q]\,b$$

can be regarded as a noncommutative version of the intuitionistic currying rule

$$p \to q \to b \quad\equiv\quad p \wedge q \to b.$$

Gödel first observed the connection between modal logic and intuitionism, foreshadowing Kripke's formulation of a state-based semantics for modal and intuitionistic logic. Kripke models also form the basis of the semantics of DL (see [7]), although as mentioned, DL does not realize the intuitionistic nature of partial correctness.

In this paper we give a Gentzen-style sequent calculus S that clearly separates partial correctness reasoning into its classical and intuitionistic parts. The system has the flavor of a noncommutative Intuitionistic Linear Logic and is in some ways related to a system of Girard [5, 6]. It is linear because certain expressions cannot be indiscriminately duplicated or eliminated. The linear implication operator takes only programs as left argument, while arbitrary partial correctness formulas can occur on the right. There is a very limited way in which the weakening rule for programs can be used—programs can be inserted only at front of an environment. There is one exception: a program of the form p^* already present in the environment can be duplicated. It is remarked in [15, p. 25] that "the addition of weakening to linear logic has less dramatic consequences than the addition of contraction has." Since programs cannot occur positively on the right hand side of $\vdash$, the system has introduction and elimination rules on the left of $\vdash$.

We give relational and trace-based semantics for this logic and show how it captures partial correctness. We then prove soundness and completeness over both classes of models. As a corollary we obtain a complete sequent calculus for inclusion and equivalence of regular expressions.

2 Syntax

The syntax of S comprises several syntactic categories. These will require some intuitive explanation, which we defer until after the formal definition. In particular we distinguish between two kinds of propositions, which we call *tests* and *formulas*.

tests	$b, c, d \ldots$	$b ::= \langle \text{atomic tests} \rangle \mid \bot \mid b \to c$
programs	$p, q, r, \ldots$	$p ::= \langle \text{atomic programs} \rangle \mid b \mid p \sqcup q \mid p \otimes q \mid p^*$
formulas	$\varphi, \psi, \ldots$	$\varphi ::= b \mid p \to \varphi$
environments	$\Gamma, \Delta, \ldots$	$\Gamma ::= \varepsilon \mid \Gamma, p \mid \Gamma, \varphi$
sequents	$\Gamma \vdash \varphi$	

In the above grammar, $\to$ is called *linear implication*, $\otimes$ is a noncommutative multiplicative connective called *tensor*, $\sqcup$ is a commutative additive connective called *disjunction*, and * is a unary operation called *iteration*. We use brackets where necessary to ensure unique readability. We abbreviate $b \to \bot$ by $\overline{b}$, $\overline{\bot}$ by $\mathbf{1}$, and $p \otimes q$ by pq.

A *test* is either an atomic test, the symbol $\bot$ representing falsity, or an expression $b \to c$ representing classical implication, where b and c are tests. We use the symbols $b, c, d, \ldots$ exclusively to stand for tests. The set of all tests is denoted $\mathcal{B}$. The sequent calculus to be presented in Section 4 will encode classical propositional logic for tests.

A *program* is either an atomic program, a test, or an expression $p \sqcup q$, $p \otimes q$, or p^*, where p and q are programs. We use the symbols $p, q, r, \ldots$ exclusively to stand for programs. The set of all programs is denoted $\mathcal{P}$. As in PDL [4], the program operators can be used to construct conventional procedural programming constructs such as conditional tests and while loops.

A *formula* is either a test or an expression $p \to \varphi$, read "after p, φ," where p is a program and φ is a formula. Intuitively, the meaning is similar to the DL modal construct $[p]\varphi$. The operator $\to$ associates to the right. We use the symbols $\varphi, \psi, \ldots$ to stand for formulas.

Environments are denoted $\Gamma, \Delta, \ldots$. An environment is a (possibly empty) sequence of programs and formulas. The empty environment is denoted ε. Intuitively, an environment describes a previous computation that has led to the current state.

Sequents are of the form $\Gamma \vdash \varphi$, where Γ is an environment and φ is a formula. We write $\vdash \varphi$ for $\varepsilon \vdash \varphi$. Intuitively, the meaning of $\Gamma \vdash \varphi$ is similar to the DL assertion $[\Gamma]\varphi$, where we think of the environment $\Gamma = \ldots, p, \ldots, \psi, \ldots$ as the rich-test program $\cdots; p; \cdots; \psi?; \cdots$ of DL.

The partial correctness assertion $\{b\}\,p\,\{c\}$ of HL is encoded by the formula $b \to p \to c$. The Hoare-style rule

$$\frac{\{b_1\}\,p_1\,\{c_1\},\ \ldots,\ \{b_n\}\,p_n\,\{c_n\}}{\{b\}\,p\,\{c\}}$$

is encoded by the sequent $b_1 \to p_1 \to c_1, \ldots, b_n \to p_n \to c_n \vdash b \to p \to c$. It follows from Theorem 6.1 that all relationally valid rules of this form are derivable; this is false for HL (see [9, 12]).

3 Semantics

3.1 Guarded Strings

Let $\mathsf{B} = \{b_1, \ldots, b_k\}$ and $\mathsf{P} = \{p_1, \ldots, p_m\}$ be finite sets of atomic tests and atomic programs, respectively. The sets B and P will be fixed throughout. The *guarded strings* over P, B were introduced in [11]; we review the definition here.

An *atom* of B is a sequence $\ell_1 \cdots \ell_k$ such that ℓ_i is either b_i or $\overline{b}_i$. We require for technical reasons that the ℓ_i occur in this order. An atom represents a minimal nonzero element of the free Boolean algebra on B. We denote by $\mathcal{A}_{\mathsf{B}}$ the set of all atoms of B. For an atom α and a test b, we write $\alpha \leq b$ if $\alpha \rightarrow b$ is a classical propositional tautology.

A *guarded string* is a sequence $\sigma = \alpha_0 q_1 \alpha_1 \cdots \alpha_{n-1} q_n \alpha_n$, where $n \geq 0$, each $\alpha_i \in \mathcal{A}_{\mathsf{B}}$, and $q_i \in \mathsf{P}$. We define $\mathbf{first}(\sigma) = \alpha_0$ and $\mathbf{last}(\sigma) = \alpha_n$.

If $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$, we can form the *fusion product* $\sigma\tau$ by concatenating σ and τ, omitting the extra copy of $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$ in between. For example, if $\sigma = \alpha p \beta$ and $\tau = \beta q \gamma$, then $\sigma\tau = \alpha p \beta q \gamma$. If $\mathbf{last}(\sigma) \neq \mathbf{first}(\tau)$, then $\sigma\tau$ does not exist.

For sets X, Y of guarded strings, define

$$\begin{aligned}
X \circ Y &\stackrel{\text{def}}{=} \{\sigma\tau \mid \sigma \in X,\ \tau \in Y,\ \sigma\tau \text{ exists}\} \\
X^0 &\stackrel{\text{def}}{=} \mathcal{A}_{\mathsf{B}} \\
X^{n+1} &\stackrel{\text{def}}{=} X \circ X^n.
\end{aligned}$$

Although fusion product is a partial operation on guarded strings, the operation $\circ$ is a total operation on sets of guarded strings. If there is no existing fusion product between an element of X and an element of Y, then $X \circ Y = \varnothing$.

Each program p denotes a set $GS(p)$ of guarded strings as follows:

$$\begin{aligned}
GS(p) &\stackrel{\text{def}}{=} \{\alpha p\beta \mid \alpha, \beta \in \mathcal{A}_{\mathsf{B}}\}, \quad p \text{ an atomic program} \\
GS(b) &\stackrel{\text{def}}{=} \{\alpha \in \mathcal{A}_{\mathsf{B}} \mid \alpha \leq b\}, \quad b \text{ a test} \\
GS(p \sqcup q) &\stackrel{\text{def}}{=} GS(p) \cup GS(q) \\
GS(p \otimes q) &\stackrel{\text{def}}{=} GS(p) \circ GS(q) \\
GS(p^*) &\stackrel{\text{def}}{=} \bigcup_{n \geq 0} GS(p)^n.
\end{aligned}$$

A guarded string σ is itself a program, and $GS(\sigma) = \{\sigma\}$.

A set of guarded strings over P, B is *regular* if it is $GS(p)$ for some program p. The regular sets of guarded strings form the free Kleene algebra with tests on generators P, B [11]; in other words, $GS(p) = GS(q)$ iff $p = q$ is a theorem of KAT.

Lemma 3.1 *The regular sets of guarded strings are closed under the Boolean operations.*

Proof. Closure under $\varnothing$ and union are explicit by means of the constructs $\perp$ and $\sqcup$. It was shown in [11] that for any program p, there is an equivalent program $\widehat{p}$ such that $GS(p) = GS(\widehat{p}) = R(\widehat{p})$, where $R(\widehat{p})$ is the regular set of strings over the alphabet $\mathsf{P} \cup \mathsf{B} \cup \{\overline{b} \mid b \in \mathsf{B}\}$ denoted by $\widehat{p}$ under the usual interpretation of regular expressions. For example, if $w = (p_1 \sqcup \cdots \sqcup p_m)^*$, we might take

$$\widehat{w} \quad = \quad ((b_1 \sqcup \overline{b}_1) \cdots (b_k \sqcup \overline{b}_k)(p_1 \sqcup \cdots \sqcup p_m))^*(b_1 \sqcup \overline{b}_1) \cdots (b_k \sqcup \overline{b}_k).$$

The set $GS(w) = GS(\widehat{w}) = R(\widehat{w})$ is the set of all guarded strings.

It remains to show closure under complement; closure under intersection follows by the De Morgan laws. Let p' be an expression such that

$$R(p') \quad = \quad R(\widehat{w}) - R(\widehat{p}).$$

The expression p' exists since the regular sets of strings over $\mathsf{P} \cup \mathsf{B} \cup \{\overline{b} \mid b \in \mathsf{B}\}$ are closed under the Boolean operations. Then $R(p')$ is a set of guarded strings since $R(\widehat{w})$ is, and

$$GS(p') = R(p') = R(\widehat{w}) - R(\widehat{p}) = GS(w) - GS(p).$$

$\blacksquare$

3.2 Trace Models

Traces are similar to guarded strings but more general. They are defined in terms of Kripke frames. A *Kripke frame* over P, B is a structure $(K,\ \mathfrak{m}_K)$, where

$$\mathfrak{m}_K : \mathsf{P} \to 2^{K\times K} \qquad \mathfrak{m}_K : \mathsf{B} \to 2^K.$$

Elements of K are called *states*. A *trace* in K is a sequence of the form $s_0q_1s_1\cdots s_{n-1}q_ns_n$, where $n \geq 0$, $s_i \in K$, $q_i \in \mathsf{P}$, and $(s_i, s_{i+1}) \in \mathfrak{m}_K(q_{i+1})$ for $0 \leq i \leq n-1$. The first and last states of σ are denoted $\mathbf{first}(\sigma)$ and $\mathbf{last}(\sigma)$, respectively. If $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$, we can fuse σ and τ to get the trace $\sigma\tau$, omitting the extra copy of $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$ in between. If $\mathbf{last}(\sigma) \neq \mathbf{first}(\tau)$ then $\sigma\tau$ does not exist. A trace $s_0q_1s_1\cdots s_{n-1}q_ns_n$ is *linear* if the s_i are distinct. The model K is a *linear trace model* if all traces are linear. By unwinding if necessary, we will always be able to assume without loss of generality that the model is linear.

If X and Y are sets of traces, define

$$\begin{aligned} X \circ Y &\stackrel{\text{def}}{=} \{\sigma\tau \mid \sigma \in X,\ \tau \in Y,\ \sigma\tau \text{ exists}\} \\ X^0 &\stackrel{\text{def}}{=} K \\ X^{n+1} &\stackrel{\text{def}}{=} X \circ X^n. \end{aligned}$$

Tests, programs, formulas, and environments are interpreted as sets of traces according to the following inductive definition:

$$
\begin{aligned}
[\![p]\!]_K &\stackrel{\mathrm{def}}{=} \{spt \mid (s,t) \in \mathfrak{m}_K(p)\}, \quad p \text{ an atomic program} \\
[\![b]\!]_K &\stackrel{\mathrm{def}}{=} \mathfrak{m}_K(b), \quad b \text{ an atomic test} \\
[\![\bot]\!]_K &\stackrel{\mathrm{def}}{=} \varnothing \\
[\![p \sqcup q]\!]_K &\stackrel{\mathrm{def}}{=} [\![p]\!]_K \cup [\![q]\!]_K \\
[\![p \otimes q]\!]_K &\stackrel{\mathrm{def}}{=} [\![p]\!]_K \circ [\![q]\!]_K \\
[\![p^*]\!]_K &\stackrel{\mathrm{def}}{=} \bigcup_{n \geq 0} [\![p]\!]_K^n \\
[\![p \to \varphi]\!]_K &\stackrel{\mathrm{def}}{=} \{s \mid \forall \tau \ \mathbf{first}(\tau) = s \text{ and } \tau \in [\![p]\!]_K \Rightarrow \mathbf{last}(\tau) \in [\![\varphi]\!]_K\} \\
[\![\varepsilon]\!]_K &\stackrel{\mathrm{def}}{=} K \\
[\![\Gamma, \Delta]\!]_K &\stackrel{\mathrm{def}}{=} [\![\Gamma]\!]_K \circ [\![\Delta]\!]_K.
\end{aligned}
$$

It follows that

$$
\begin{aligned}
[\![\bar{b}]\!]_K &= K - [\![b]\!]_K \\
[\![\mathbf{1}]\!]_K &= K.
\end{aligned}
$$

Every trace σ has an associated guarded string $\mathrm{gs}(\sigma)$ defined by

$$
\mathrm{gs}(s_0 q_1 s_1 \cdots s_{n-1} q_n s_n) \stackrel{\mathrm{def}}{=} \alpha_0 q_1 \alpha_1 \cdots \alpha_{n-1} q_n \alpha_n,
$$

where α_i is the unique atom of B such that $s_i \in [\![\alpha_i]\!]_K$. Thus $\mathrm{gs}(\sigma)$ is the unique guarded string over P, B such that $\sigma \in [\![\mathrm{gs}(\sigma)]\!]_K$.

The sequent $\Gamma \vdash \varphi$ is *valid* in the trace model K if for all traces $\sigma \in [\![\Gamma]\!]_K$, $\mathbf{last}(\sigma) \in [\![\varphi]\!]_K$; equivalently, if $[\![\Gamma]\!]_K \subseteq \mathbf{last}^{-1}([\![\varphi]\!]_K)$, or if $[\![\Gamma]\!]_K \subseteq [\![\Gamma, \varphi]\!]_K$.

The relationship between trace semantics and guarded strings is given by the following lemma.

Lemma 3.2 *In any trace model K, for any program p and trace τ, $\tau \in [\![p]\!]_K$ iff $\mathrm{gs}(\tau) \in GS(p)$. In other words, $[\![p]\!]_K = \mathrm{gs}^{-1}(GS(p))$. The map $X \mapsto \mathrm{gs}^{-1}(X)$ is a* KAT *homomorphism from the algebra of regular sets of guarded strings to the algebra of regular sets of traces over K.*

Proof. Induction on the structure of p. ∎

3.3 Relational Models

Kripke frames $(K, \mathfrak{m}_K)$ also give rise to relational models. In a relational model, tests, programs, formulas, and environments are interpreted as binary relations on K. Tests and formulas denote subsets of the identity relation.

$$\begin{aligned}
[p]_K &\stackrel{\text{def}}{=} \mathfrak{m}_K(p), \quad p \text{ an atomic program} \\
[b]_K &\stackrel{\text{def}}{=} \{(s,s) \mid s \in \mathfrak{m}_K(b)\}, \quad b \text{ an atomic test} \\
[\bot]_K &\stackrel{\text{def}}{=} \varnothing \\
[p \sqcup q]_K &\stackrel{\text{def}}{=} [p]_K \cup [q]_K \\
[p \otimes q]_K &\stackrel{\text{def}}{=} [p]_K \circ [q]_K \\
[p^*]_K &\stackrel{\text{def}}{=} \bigcup_{n \geq 0} [p]_K^n \\
[p \to \varphi]_K &\stackrel{\text{def}}{=} \{(s,s) \mid \forall t\ (s,t) \in [p]_K \Rightarrow (t,t) \in [\varphi]_K\} \\
[\varepsilon]_K &\stackrel{\text{def}}{=} \{(s,s) \mid s \in K\} \\
[\Gamma, \Delta]_K &\stackrel{\text{def}}{=} [\Gamma]_K \circ [\Delta]_K.
\end{aligned}$$

Here $\circ$ denotes ordinary composition of binary relations. It follows that

$$\begin{aligned}
[\bar{b}]_K &= \{(s,s) \mid (s,s) \notin [b]_K\} \\
[\mathbf{1}]_K &= \{(s,s) \mid s \in K\}.
\end{aligned}$$

Writing $s \vDash \varphi$ for $(s,s) \in [\varphi]_K$, the defining clause for $p \to \varphi$ becomes

$$s \vDash p \to \varphi \iff \forall t\ (s,t) \in [p]_K \Rightarrow t \vDash \varphi,$$

thus the meaning of $p \to \varphi$ is essentially the same as the meaning of the box formula $[p]\varphi$ of DL.

The sequent $\Gamma \vdash \varphi$ is *valid* in the relational model on $(K, \mathfrak{m}_K)$ if for all $s, t \in K$, if $(s,t) \in [\Gamma]_K$, then $(t,t) \in [\varphi]_K$; equivalently, if the DL formula $[\Gamma]\varphi$ is true in all states under the rich-test semantics [4], where the environment $\Gamma = \ldots, p, \ldots, \psi, \ldots$ is interpreted as the rich-test program $\cdots; p; \cdots; \psi?; \cdots$.

3.4 Relationship between Trace-Based and Relational Models

It can be shown by induction on syntax that the map

$$r : X \quad \mapsto \quad \{(\mathbf{first}(\sigma), \mathbf{last}(\sigma)) \mid \sigma \in X\}$$

from sets of traces on K to binary relations on K maps $[\![p]\!]_K$ to $[p]_K$ and $[\![\varphi]\!]_K$ to $[\varphi]_K$, using the fact that r commutes with the operators $\cup$ and $\circ$ on sets of traces and binary relations. It follows that validity over relational models is the same as validity over trace models.

4 A Deductive System

The rules of System S are given in Figure 1. All rules are of the form

$$\frac{\Gamma_1 \vdash \varphi_1 \quad \ldots \quad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi} .$$

The sequents above the line are the *premises* and the sequent below the line is the *conclusion*.

We will use the notation $\Gamma \vdash \varphi$ ambiguously as both an object and a meta-assertion. As an object it denotes a sequent, *i.e.* a sequence of symbols over the appropriate vocabulary. As a meta-assertion it says that the sequent $\Gamma \vdash \varphi$ is provable in S. In particular, $\Gamma \nvdash \varphi$ means that the sequent $\Gamma \vdash \varphi$ is not provable in S. The proper interpretation should always be clear from context.

A rule is *admissible* if for any substitution instance for which the premises are provable, the conclusion is also provable. The proof of the conclusion may depend on the structure of the expressions substituted for the metasymbols appearing in the rule or on the proofs of the premises. To show admissibility, it suffices to derive the conclusion in S augmented with the premises as extra axioms, considering the metasymbols appearing in the rule as atomic symbols in the object language. Any such derivation will then be uniformly valid over all substitution instances.

Axiom: $b \vdash c$ where $b \to c$ is a classical propositional tautology

Arrow Rules:

(R $\to$) $\dfrac{\Gamma, p \vdash \varphi}{\Gamma \vdash p \to \varphi}$ **(L $\to$)** $\dfrac{\Gamma, p, \psi, \Delta \vdash \varphi}{\Gamma, p \to \psi, p, \Delta \vdash \varphi}$

Introduction Rules:

(I $\otimes$) $\dfrac{\Gamma, p, q, \Delta \vdash \varphi}{\Gamma, p \otimes q, \Delta \vdash \varphi}$ **(I $\sqcup$)** $\dfrac{\Gamma, p, \Delta \vdash \varphi \qquad \Gamma, q, \Delta \vdash \varphi}{\Gamma, p \sqcup q, \Delta \vdash \varphi}$

(I *) $\dfrac{\varphi, p \vdash \varphi}{\varphi, p^* \vdash \varphi}$ **(I $\perp$)** $\Gamma, \perp, \Delta \vdash \varphi$

Elimination Rules:

(E $\otimes$) $\dfrac{\Gamma, p \otimes q, \Delta \vdash \varphi}{\Gamma, p, q, \Delta \vdash \varphi}$

(E1 $\sqcup$) $\dfrac{\Gamma, p \sqcup q, \Delta \vdash \varphi}{\Gamma, p, \Delta \vdash \varphi}$ **(E2 $\sqcup$)** $\dfrac{\Gamma, p \sqcup q, \Delta \vdash \varphi}{\Gamma, q, \Delta \vdash \varphi}$

(E1 *) $\dfrac{\Gamma, p^*, \Delta \vdash \varphi}{\Gamma, \Delta \vdash \varphi}$ **(E2 *)** $\dfrac{\Gamma, p^*, \Delta \vdash \varphi}{\Gamma, p, \Delta \vdash \varphi}$

Weakening Rules:

(W ψ) $\dfrac{\Gamma, \Delta \vdash \varphi}{\Gamma, \psi, \Delta \vdash \varphi}$ **(W p)** $\dfrac{\Gamma \vdash \varphi}{p, \Gamma \vdash \varphi}$ **(W *)** $\dfrac{\Gamma, p^*, \Delta \vdash \varphi}{\Gamma, p^*, p^*, \Delta \vdash \varphi}$

Cut Rule:

(cut) $\dfrac{\Gamma \vdash \psi \qquad \Gamma, \psi, \Delta \vdash \varphi}{\Gamma, \Delta \vdash \varphi}$

Figure 1: Rules of System S

4.1 Basic Properties

Lemma 4.1 *The rule*

$$\textbf{(E 1)} \quad \frac{\Gamma, \mathbf{1}, \Delta \vdash \varphi}{\Gamma, \Delta \vdash \varphi}$$

is admissible.

Proof. From **(I ⊥)** and **(R →)** we get $\Gamma \vdash \mathbf{1}$. The desired conclusion follows from **(cut)**. ∎

Lemma 4.2 *The rule and sequent*

(i) $\dfrac{\varphi \vdash \psi}{p \to \varphi \vdash p \to \psi}$

(ii) $\varphi \vdash \varphi$

are admissible.

Proof. The following diagram gives a proof of (i).

$$\dfrac{\dfrac{\dfrac{\varphi \vdash \psi}{p, \varphi \vdash \psi}\,\textbf{(W }p\textbf{)}}{p \to \varphi, p \vdash \psi}\,\textbf{(L}\to\textbf{)}}{p \to \varphi \vdash p \to \psi}\,\textbf{(R}\to\textbf{)}$$

The identity sequent (ii) follows by induction on the structure of φ using (i). The basis $b \vdash b$ is an instance of the axiom. ∎

Lemma 4.3 *The rules*

$$\textbf{(E}\to\textbf{)} \quad \frac{\Gamma \vdash p \to \varphi}{\Gamma, p \vdash \varphi} \qquad\qquad \textbf{(W}\perp\textbf{)} \quad \frac{\Gamma \vdash \perp}{\Gamma, p \vdash \perp}$$

are admissible.

Proof. For $(\mathbf{E} \rightarrow)$, we have $\varphi \vdash \varphi$ by Lemma 4.2(ii). The following figure gives the remainder of the derivation.

$$\dfrac{\Gamma \vdash p \to \varphi \qquad \dfrac{\begin{array}{c}\dfrac{\varphi \vdash \varphi}{p, \varphi \vdash \varphi}\,\textbf{(W } p\textbf{)} \\ \vdots\ \textbf{(W } p\textbf{)}, \textbf{(W } \psi\textbf{)} \\ \Gamma, p, \varphi \vdash \varphi\end{array}}{\Gamma, p \to \varphi, p \vdash \varphi}\,\textbf{(L} \rightarrow\textbf{)}}{\Gamma, p \vdash \varphi}\,\textbf{(cut)}$$

To derive $(\mathbf{W} \perp)$, the sequent $\Gamma, \perp, p \vdash \perp$ is an instance of $(\mathbf{I} \perp)$. Applying **(cut)** to this and the premise $\Gamma \vdash \perp$ yields the desired conclusion. ∎

Lemma 4.4 *The rules*

$$\textbf{(curry)}\ \dfrac{\Gamma, p \to q \to \psi, \Delta \vdash \varphi}{\Gamma, pq \to \psi, \Delta \vdash \varphi} \qquad \textbf{(uncurry)}\ \dfrac{\Gamma, pq \to \psi, \Delta \vdash \varphi}{\Gamma, p \to q \to \psi, \Delta \vdash \varphi}$$

are admissible.

Proof. By **(cut)**, it suffices to show $pq \to \psi \vdash p \to q \to \psi$ and $p \to q \to \psi \vdash pq \to \psi$. For the first, starting with $pq \to \psi \vdash pq \to \psi$, apply $(\mathbf{E} \rightarrow)$ and $(\mathbf{E} \otimes)$ to get $pq \to \psi, p, q \vdash \psi$, then apply $(\mathbf{R} \rightarrow)$ twice. For the second, starting with $\psi \vdash \psi$, apply $(\mathbf{W}\ p)$ twice to get $p, q, \psi \vdash \psi$, then apply $(\mathbf{L} \rightarrow)$ twice to get $p \to q \to \psi, p, q \vdash \psi$. The result then follows from $(\mathbf{I} \otimes)$ and $(\mathbf{R} \rightarrow)$. ∎

Lemma 4.5 *Every φ is provably equivalent to some $p \to \perp$ in the sense that $\varphi \vdash p \to \perp$ and $p \to \perp \vdash \varphi$.*

Proof. The formula $q_1 \to \cdots \to q_n \to b$ is equivalent to $q_1 \cdots q_n \overline{b} \to \perp$. The proof of this fact is quite easy using Lemma 4.4 and is left to the reader. ∎

4.2 Relation to Kleene Algebra

We show in this section that S induces a left-handed Kleene algebra structure on programs. Recall that a *Kleene algebra* (KA) is an idempotent semiring such that p^*q is the least solution to $q + px \leq x$ and qp^* is the least solution to $q + xp \leq x$. Equivalently, a Kleene algebra is an idempotent semiring satisfying

$$1 + pp^* \;=\; 1 + p^*p \;=\; p \tag{1}$$

$$px \leq x \;\rightarrow\; p^*x \leq x \tag{2}$$

$$xp \leq x \;\rightarrow\; xp^* \leq x. \tag{3}$$

Boffa [1, 2], based on results of Krob [13], shows that for the equational theory of the regular sets, the right-hand rule (3) is unnecessary. We will call an idempotent semiring satisfying (1) and (2) a *left-handed Kleene algebra*. Boffa's result says that for regular expressions p and q, $R(p) = R(q)$ iff $p = q$ is a logical consequence of the axioms of left-handed Kleene algebra, where R is the usual interpretation of regular expressions as sets of strings.

More specifically, Krob [13] shows that the *classical equations* of Conway [3], along with a certain infinite but independently characterized set of axioms, logically entail all identities of the regular sets over P. The classical equations of Conway are the axioms of idempotent semirings, the equations (1), and the equations

$$\begin{array}{rclcrcl} (p+q)^* & = & (p^*q)^*p^* & \qquad & p^* & = & p^{**} \\ (pq)^* & = & 1 + p(qp)^*q & \qquad & p^* & = & (p^n)^*(1+p)^{n-1},\ n \geq 0. \end{array}$$

Boffa [1, 2] actually shows that these equations plus the rule

$$p^2 = p \;\;\rightarrow\;\; p^* = 1 + p \tag{4}$$

—which the reader will note is neither left- nor right-handed—imply all the axioms of Krob, therefore the classical equations of Conway plus Boffa's rule (4) are complete for the equational theory of the regular sets over P. The classical equations and Boffa's rule are all easily shown to be theorems of left-handed KA.

Our first task is to extend these results to Kleene algebra with tests and guarded strings.

Lemma 4.6 *Left-handed* KAT *is complete for the equational theory of the regular sets of guarded strings over* P *and* B*. In other words, for every pair of programs* p, q *in the language of* KAT*,* $GS(p) = GS(q)$ *if and only if the equation* $p = q$ *is a logical consequence of the axioms of left-handed* KAT*.*

Proof. We adapt an argument of [11], in which the same result was proved for KAT with both the left- and right-hand rule. It was shown there that for any program p, there is an equivalent program $\widehat{p}$ such that

(i) $p = \widehat{p}$ is a theorem of KAT, and

(ii) $GS(\widehat{p}) = R(\widehat{p})$, where $R(\widehat{p})$ is the regular set of strings over the alphabet $\mathsf{P} \cup \mathsf{B} \cup \{\overline{b} \mid b \in \mathsf{B}\}$ denoted by $\widehat{p}$ under the usual interpretation of regular expressions.

In other words, any p can be transformed by the axioms of KAT to another program $\widehat{p}$ such that the set of guarded strings denoted by $\widehat{p}$ is the same as the set of strings denoted by $\widehat{p}$.

Now to show completeness of KAT over guarded strings, [11] argued as follows. Suppose $GS(p) = GS(q)$. Then

$$R(\widehat{p}) = GS(\widehat{p}) = GS(p) = GS(q) = GS(\widehat{q}) = R(\widehat{q}).$$

Since KA is complete for the equational theory of the regular sets, $\widehat{p} = \widehat{q}$ is a theorem of KA. Combining this with (i) for p and q implies that $p = q$ is a theorem of KAT.

To adapt this to the present situation, we observe that $\widehat{p} = \widehat{q}$ is a theorem of left-handed KA by the results of Boffa and Krob. Thus in order to complete the proof, we need only ascertain that the right-hand rule (3) is not needed in the proof of $p = \widehat{p}$. This does not follow from Boffa's and Krob's results, since the argument is in KAT, not KA. However, a perusal of [11] reveals that the proof of $p = \widehat{p}$ uses neither the left- or the right-hand rule, but can be carried out using only the classical equations of Conway and the axioms of Boolean algebra. ∎

We now describe the left-handed KAT structure induced by S. Define $p \sqsubseteq q$ if $q \to \varphi \vdash p \to \varphi$ is admissible; that is, if $q \to \varphi \vdash p \to \varphi$ is provable for all φ. Define $p \equiv q$ if $p \sqsubseteq q$ and $q \sqsubseteq p$. The relation $\sqsubseteq$ is a preorder, therefore $\equiv$ is an equivalence relation and $\sqsubseteq$ is a partial order on $\equiv$-classes. Reflexivity is Lemma 4.2(ii) and transitivity follows from a single application of **(cut)**. Moreover, the rules **(E1** $\sqcup$**)**, **(E2** $\sqcup$**)**, and **(I** $\sqcup$**)** imply that $p \sqcup q$ is the $\sqsubseteq$-least upper bound of p and q modulo $\equiv$.

Lemma 4.7 *Let $\mathcal{P}/\!\equiv$ denote the set of $\equiv$-equivalence classes. The operations $\sqcup, \otimes$, and * are well-defined on $\mathcal{P}/\!\equiv$, and the quotient structure $(\mathcal{P}/\!\equiv, \sqcup, \otimes, ^*, \bot, \mathbf{1})$ is a left-handed* KA.

Proof. We must argue that all the following properties hold:

$$\begin{array}{ll} p \sqcup (q \sqcup r) \equiv (p \sqcup q) \sqcup r & p(qr) \equiv (pq)r \\ p \sqcup q \equiv q \sqcup p & \mathbf{1}p \equiv p\mathbf{1} \equiv p \\ p \sqcup \bot \equiv p & \bot p \equiv p\bot \equiv \bot \\ p \sqcup p \equiv p & \mathbf{1} \sqcup pp^* \equiv p^* \\ p(q \sqcup r) \equiv pq \sqcup pr & \mathbf{1} \sqcup p^*p \equiv p^* \\ (p \sqcup q)r \equiv pr \sqcup qr & pq \sqsubseteq q \to p^*q \sqsubseteq q. \end{array} \tag{5}$$

These are just the laws of left-handed KA written with the symbols of S.

Here is an explicit derivation of the distributive law $p(q \sqcup r) \sqsubseteq pq \sqcup pr$:

$$\dfrac{\dfrac{\dfrac{\dfrac{pq \sqcup pr \to \varphi \vdash pq \sqcup pr \to \varphi}{pq \sqcup pr \to \varphi, pq \sqcup pr \vdash \varphi}\ (\mathbf{E} \to)}{\dfrac{pq \sqcup pr \to \varphi, pq \vdash \varphi}{pq \sqcup pr \to \varphi, p, q \vdash \varphi}\ (\mathbf{E} \otimes) \qquad \dfrac{pq \sqcup pr \to \varphi, pr \vdash \varphi}{pq \sqcup pr \to \varphi, p, r \vdash \varphi}\ (\mathbf{E} \otimes)}\ (\mathbf{E1} \sqcup), (\mathbf{E2} \sqcup)}{pq \sqcup pr \to \varphi, p, q \sqcup r \vdash \varphi}\ (\mathbf{I} \sqcup)}{pq \sqcup pr \to \varphi \vdash p(q \sqcup r) \to \varphi}\ (\mathbf{I} \otimes), (\mathbf{R} \to)$$

All the other axioms of idempotent semirings follow in an equally straightforward manner. It follows that the operators $\sqcup$ and $\otimes$ are monotone with respect to $\sqsubseteq$, therefore are well-defined on $\equiv$-classes.

The inequality $p^*p^* \sqsubseteq p^*$ follows from $(\mathbf{W}\ ^*)$ by:

$$\dfrac{\dfrac{\dfrac{p^* \to \varphi \vdash p^* \to \varphi}{p^* \to \varphi, p^* \vdash \varphi}\ (\mathbf{E} \to)}{p^* \to \varphi, p^*, p^* \vdash \varphi}\ (\mathbf{W}\ ^*)}{p^* \to \varphi \vdash p^*p^* \to \varphi}\ (\mathbf{I} \otimes), (\mathbf{R} \to)$$

The inequalities $\mathbf{1} \sqsubseteq p^*$ and $p \sqsubseteq p^*$ follow from $(\mathbf{E1}\ ^*)$ and $(\mathbf{E2}\ ^*)$, respectively, in a similar fashion. Arguing equationally, we have

$$\mathbf{1} \sqcup pp^* \quad \sqsubseteq \quad p^* \sqcup p^*p^* \quad \sqsubseteq \quad p^* \sqcup p^* \quad \sqsubseteq \quad p^*$$

and similarly, $\mathbf{1} \sqcup p^*p \sqsubseteq p^*$. We also have $p^* \sqsubseteq p^{**}$ and $p^* \sqsubseteq p^*\mathbf{1} \sqsubseteq p^*p^*$.

For the remaining inequalities involving *, we will need the left-hand rule (5). This is established by the following derivation:

$$\cfrac{\cfrac{\cfrac{q \to \varphi \vdash pq \to \varphi}{q \to \varphi, p \vdash q \to \varphi}\ \textbf{(E} \to\textbf{)}, \textbf{(E} \otimes\textbf{)}, \textbf{(R} \to\textbf{)}}{q \to \varphi, p^* \vdash q \to \varphi}\ \textbf{(I}\ ^*\textbf{)}}{q \to \varphi \vdash p^*q \to \varphi}\ \textbf{(E} \to\textbf{)}, \textbf{(I} \otimes\textbf{)}, \textbf{(R} \to\textbf{)}$$

Since $p^*p^* \sqsubseteq p^*$, by (5) we have $p^{**}p^* \sqsubseteq p^*$, therefore

$$p^{**} \;=\; p^{**}\mathbf{1} \;\sqsubseteq\; p^{**}p^* \;\sqsubseteq\; p^*.$$

It remains to show $p^* \sqsubseteq \mathbf{1} \sqcup pp^*$ and $p^* \sqsubseteq \mathbf{1} \sqcup p^*p$. By inequalities already established, we have $p(\mathbf{1} \sqcup pp^*) \sqsubseteq \mathbf{1} \sqcup pp^*$ and $p(\mathbf{1} \sqcup p^*p) \sqsubseteq \mathbf{1} \sqcup p^*p$. The results then follow from (5).

It follows by equational reasoning that * is also monotone with respect to $\sqsubseteq$, therefore well-defined on $\equiv$-classes: if $p \sqsubseteq q$, then $pq^* \sqsubseteq qq^* \sqsubseteq q^*q^* \sqsubseteq q^*$. By (5), $p^* \sqsubseteq p^*\mathbf{1} \sqsubseteq p^*q^* \sqsubseteq q^*$. ∎

Lemma 4.8 *If $b \to c$ is a classical tautology, then $b \sqsubseteq c$. Thus the tests form a Boolean algebra modulo $\equiv$.*

Proof. We have $c \to \varphi, b \vdash c$ by the axiom $b \vdash c$ and the weakening rule $\textbf{(W}\ \psi\textbf{)}$, and we have $c \to \varphi, c \vdash \varphi$ by $\textbf{(E} \to\textbf{)}$. The desired conclusion $c \to \varphi \vdash b \to \varphi$ then follows from **(cut)** and $\textbf{(R} \to\textbf{)}$. ∎

Combining Lemmas 4.7 and 4.8 and the fact that the regular sets of guarded strings form the free KAT on generators P and B, we have

Lemma 4.9 *The structure $(\mathcal{P}/\equiv,\ \mathcal{B}/\equiv,\ \sqcup,\ \otimes,\ ^*,\ ^-,\ \bot,\ \mathbf{1})$ is a left-handed KAT and is isomorphic to the algebra of regular sets of guarded strings over P and B. Thus for any programs p and q, $p \sqsubseteq q$ iff $GS(p) \subseteq GS(q)$ and $p \equiv q$ iff $GS(p) = GS(q)$.*

5 Soundness

Theorem 5.1 *If $\Gamma \vdash \varphi$ is provable, then it is valid in all trace and relational models.*

Proof. We need only show soundness over trace models. This is easily established by induction on proofs in S with one case for each proof rule. We argue the cases **(cut)** and **(L $\rightarrow$)** explicitly.

For **(cut)**, we need to show that $[\![\Gamma, \Delta]\!]_K \subseteq [\![\Gamma, \Delta, \varphi]\!]_K$ under the assumptions $[\![\Gamma]\!]_K \subseteq [\![\Gamma, \psi]\!]_K$ and $[\![\Gamma, \psi, \Delta]\!]_K \subseteq [\![\Gamma, \psi, \Delta, \varphi]\!]_K$. Using monotonicity of $\circ$,

$$
\begin{aligned}
[\![\Gamma, \Delta]\!]_K \;&=\; [\![\Gamma]\!]_K \circ [\![\Delta]\!]_K \;\subseteq\; [\![\Gamma, \psi]\!]_K \circ [\![\Delta]\!]_K \\
&=\; [\![\Gamma, \psi, \Delta]\!]_K \;\subseteq\; [\![\Gamma, \psi, \Delta, \varphi]\!]_K \;=\; [\![\Gamma]\!]_K \circ [\![\psi]\!]_K \circ [\![\Delta, \varphi]\!]_K \\
&\subseteq\; [\![\Gamma]\!]_K \circ [\![\mathbf{1}]\!]_K \circ [\![\Delta, \varphi]\!]_K \;=\; [\![\Gamma]\!]_K \circ [\![\Delta, \varphi]\!]_K \;=\; [\![\Gamma, \Delta, \varphi]\!]_K.
\end{aligned}
$$

For **(L $\rightarrow$)**, we need to show that if $[\![\Gamma, p, \psi, \Delta]\!]_K \subseteq \mathbf{last}^{-1}([\![\varphi]\!]_K)$, then $[\![\Gamma, p \rightarrow \psi, p, \Delta]\!]_K \subseteq \mathbf{last}^{-1}([\![\varphi]\!]_K)$. It suffices to show that $[\![p \rightarrow \psi]\!]_K \circ [\![p]\!]_K \subseteq [\![p]\!]_K \circ [\![\psi]\!]_K$. But

$$
\begin{aligned}
& \tau \in [\![p \rightarrow \psi]\!]_K \circ [\![p]\!]_K \\
& \quad \Leftrightarrow \;\; \mathbf{first}(\tau) \in [\![p \rightarrow \psi]\!]_K \text{ and } \tau \in [\![p]\!]_K \\
& \quad \Rightarrow \;\; \tau \in [\![p]\!]_K \text{ and } \mathbf{last}(\tau) \in [\![\psi]\!]_K \\
& \quad \Leftrightarrow \;\; \tau \in [\![p]\!]_K \circ [\![\psi]\!]_K.
\end{aligned}
$$

The other cases are equally straightforward. ∎

6 Completeness

Theorem 6.1 *If $\Gamma \nvdash \varphi$, then there exist a linear trace model K and a trace $\sigma \in [\![\Gamma]\!]_K$ such that $\mathbf{last}(\sigma) \notin [\![\varphi]\!]_K$.*

Proof. By Lemma 4.5, we can assume without loss of generality that φ is of the form $p \to \bot$. The proof proceeds by induction on the length of Γ. For the basis of the induction, suppose Γ is empty, so that $\nvdash p \to \bot$. Then $p \not\equiv \bot$. By Lemma 4.9, $GS(p) \neq \varnothing$. Construct a Kripke frame K consisting of a single linear trace σ such that $\mathrm{gs}(\sigma) \in GS(p)$. By Lemma 3.2, $\sigma \in [\![p]\!]_K$. Then $\mathbf{first}(\sigma) \in [\![\varepsilon]\!]_K$ and $\mathbf{first}(\sigma) \notin [\![p \to \bot]\!]_K$.

For the induction step in which the environment ends with a program, say $\Gamma, p \nvdash \varphi$, we have $\Gamma \nvdash p \to \varphi$ by $(\mathbf{E} \to)$. Applying the induction hypothesis, there exist a linear trace model K and traces σ and τ such that $\sigma \in [\![\Gamma]\!]_K$, $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$, $\tau \in [\![p]\!]_K$, and $\mathbf{last}(\tau) \notin [\![\varphi]\!]_K$. Then $\sigma\tau \in [\![\Gamma, p]\!]_K$ and $\mathbf{last}(\sigma\tau) \notin [\![\varphi]\!]_K$.

Finally, we argue the induction step in which the environment ends with a formula, say $\Gamma, \psi \nvdash \varphi$. By Lemma 4.5, we can rewrite this as $\Gamma, q \to \bot \nvdash p \to \bot$. Let w be an expression representing the set of all guarded strings (see Lemma 3.1). Let r and s be programs such that $GS(r) = GS(p) \cap GS(qw)$ and $GS(s) = GS(p) - GS(qw)$. These programs exist by Lemma 3.1, and $GS(p) = GS(r \sqcup s)$. By Lemma 4.9, we can replace p by $r \sqcup s$ to get $\Gamma, q \to \bot \nvdash r \sqcup s \to \bot$. By $(\mathbf{R} \to)$, $\Gamma, q \to \bot, r \sqcup s \nvdash \bot$, and by $(\mathbf{I} \sqcup)$, either $\Gamma, q \to \bot, r \nvdash \bot$ or $\Gamma, q \to \bot, s \nvdash \bot$. But it cannot be the former, since $\Gamma, q \to \bot, q, w \vdash \bot$, therefore $\Gamma, q \to \bot \vdash qw \to \bot$, and by Lemma 4.9, $r \sqsubseteq qw$, therefore by **(cut)**, $\Gamma, q \to \bot \vdash r \to \bot$.

Thus it must be the case that $\Gamma, q \to \bot, s \nvdash \bot$, so $\Gamma, q \to \bot \nvdash s \to \bot$. By weakening we have $\Gamma \nvdash s \to \bot$. Then by the induction hypothesis, there exist a linear trace model K and traces $\sigma \in [\![\Gamma]\!]_K$ and $\tau \in [\![s]\!]_K$ such that $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$. Construct a trace model M consisting only of the linear trace $\sigma\tau$. By Lemma 3.2, $\tau \notin [\![qw]\!]_M$, therefore no prefix of τ is in $[\![q]\!]_M$. Then $\mathbf{last}(\sigma) \in [\![q \to \bot]\!]_M$, therefore $\sigma \in [\![\Gamma, q \to \bot]\!]_M$. Moreover, $\mathbf{last}(\sigma) \notin [\![p \to \bot]\!]_M$, since $\mathbf{last}(\sigma) = \mathbf{first}(\tau)$ and $\tau \in [\![p]\!]_M$. ∎

7 Future Work

Several interesting questions present themselves for further investigation.

1. The completeness proof relies on the results of Boffa [1, 2], which are

based in turn on the results of Krob [13]. Krob's proof is fairly involved, comprising an entire journal issue. One would like to have a proof of completeness based on first principles.

2. The relative expressive and deductive power of S compared with similar systems such as KAT, PDL, and PHL is not completely understood. S is at least as expressive as PHL and the equational theory of KAT, and apparently more so, since it is not clear how to express general sequents $\varphi_1, p_1, \varphi_2, \ldots, p_{n-1}, \varphi_n \vdash \psi$ in PHL or KAT. On the other hand, it is not clear how to express general Horn formulas of KA such as $px = xq \to p^*x = xq^*$ in S.

3. Application of the linear implication operator $\to$ is limited to programs on the left-hand side and formulas on the right-hand side. It would be interesting to see whether more general forms correspond to anything useful and whether the system can be extended to handle them. The operator $\to$ is a form of residuation (see [14, 8]), and this connection bears further investigation.

4. How hard is it to decide whether a given sequent is valid? It is in *EXPTIME* but at least *PSPACE*-hard. We conjecture that the problem is *PSPACE*-complete.

5. We would like to extend the system to handle liveness properties and total correctness in a way that is consistent with its intuitionistic foundations.

6. We would like to undertake a deeper investigation into the structure of proofs with an eye toward establishing normal form and cut elimination theorems.

Acknowledgements

The support of the National Science Foundation under grant CCR-9708915 and Polish KBN Grant 8 T11C 035 14 is gratefully acknowledged.

References

[1] Maurice Boffa. Une remarque sur les systèmes complets d'identités rationnelles. *Informatique Théoretique et Applications/Theoretical Informatics and Applications*, 24(4):419–423, 1990.

[2] Maurice Boffa. Une condition impliquant toutes les identités rationnelles. *Informatique Théoretique et Applications/Theoretical Informatics and Applications*, 29(6):515–518, 1995.

[3] John Horton Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, London, 1971.

[4] Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, 1979.

[5] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[6] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types.* Cambridge University Press, 1989.

[7] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic.* MIT Press, Cambridge, MA, 2000.

[8] Dexter Kozen. On action algebras. In J. van Eijck and A. Visser, editors, *Logic and Information Flow*, pages 78–88. MIT Press, 1994.

[9] Dexter Kozen. On Hoare logic and Kleene algebra with tests. *Trans. Computational Logic*, 1(1):60–76, July 2000.

[10] Dexter Kozen and Maria-Cristina Patron. Certification of compiler optimizations using Kleene algebra with tests. In John Lloyd, Veronica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catuscia Palamidessi, Luis Moniz Pereira, Yehoshua Sagiv, and Peter J. Stuckey, editors, *Proc. 1st Int. Conf. Computational Logic (CL2000)*, volume 1861 of *Lecture Notes in Artificial Intelligence*, pages 568–582, London, July 2000. Springer-Verlag.

[11] Dexter Kozen and Frederick Smith. Kleene algebra with tests: Completeness and decidability. In D. van Dalen and M. Bezem, editors, *Proc. 10th Int. Workshop Computer Science Logic (CSL'96)*, volume 1258 of *Lecture Notes in Computer Science*, pages 244–259, Utrecht, The Netherlands, September 1996. Springer-Verlag.

[12] Dexter Kozen and Jerzy Tiuryn. On the completeness of propositional Hoare logic. In J. Desharnais, editor, *Proc. 5th Int. Seminar Relational Methods in Computer Science (RelMiCS 2000)*, pages 195–202, January 2000.

[13] Daniel Krob. A complete system of B-rational identities. *Theoretical Computer Science*, 89(2):207–343, October 1991.

[14] Vaughan Pratt. Action logic and pure induction. In J. van Eijck, editor, *Proc. Logics in AI: European Workshop JELIA '90*, volume 478 of *Lecture Notes in Computer Science*, pages 97–120, New York, September 1990. Springer-Verlag.

[15] A. S. Troelstra. *Lectures on Linear Logic*, volume 29 of *CSLI Lecture Notes*. Center for the Study of Language and Information, 1992.