Partial Correctness Assertions Provable in Dynamic Logics

Daniel Leivant*

Computer Science Department, Indiana University, Bloomington, IN 47405.
leivant@cs.indiana.edu.

Abstract. We consider a formalism **DL** for first order Dynamic Logic, based on Segerberg's axioms for modalities, and observe that **DL** is not conservative over Hoare Logic (**HL**) when the background theory is empty, but is conservative if the background theory is the complete theory of an expressive structure (in the sense of Cook). We identify Peano Arithmetic (**PA**) as the transition point between these two states of affairs: **DL** is conservative over **HL** in the presence of a number theory that contains **PA**, and is not conservative for the sub-theories of **PA** with a bound on the complexity of induction formulas.
We proceed to delineate a natural sub-formalism of **DL**, with Segerberg's induction restricted to first order formulas, and prove that the resulting calculus proves exactly the same partial correctness assertions as **HL**, regardless of the background first order theory.

1 Introduction

Hoare-style Logics prove partial correctness assertions (PCAs) about imperative programs. Their prominent role in program verification is due in part to their being syntax-directed: the inference rules follow the inductive buildup of programs. As a result, proofs can be be converted into program annotations, and inference rules can guide program derivation and transformation. In contrast, some central rules of Dynamic Logics are not syntax directed, allowing reasoning that intertwines formulas and programs in complex ways. This added complexity is obviously necessary when proving properties of programs that are themselves more complex than PCAs, such as $[\boldsymbol{\alpha}^*]\top \to \langle \boldsymbol{\alpha}^* \rangle \boldsymbol{\varphi}$.[1] But does Dynamic Logic buy us PCA's that Hoare's Logic fails to prove? That is, are there PCA that are proved in first order logic augmented with Segerberg's rules for program modalities, but are not provable using only PCA's along the way?[2]

The answer might depend, of course, on the background first order theory **T**, which in the case of Hoare's Logic manifests itself via the implicational first order formulas used in the Rule of Consequence,

* Research partially supported by NSF grant CCR-CCR-0105651.

[1] I.e., the formula stating that if all iterations of $\boldsymbol{\alpha}$ terminate then $\boldsymbol{\varphi}$ is true after some iterate.

[2] A formalism for Dynamic Logic considered by Harel [5,6] has an additional inference rule, dubbed Convergence. We discuss elsewhere that extension.

I. Walukiewicz (Ed.): FOSSACS 2004, LNCS 2987, pp. 304–317, 2004.

$$\frac{\varphi \to \varphi' \quad \varphi'[\alpha]\psi' \quad \psi' \to \psi}{\varphi[\alpha]\psi}$$

If a Hoare-style logic **H** is relatively complete for a structure $\mathcal{S}$, and **T** consists of the entire first order theory $Th(\mathcal{S})$ of $\mathcal{S}$, then Cook's Relative Completeness Theorem applies, and already **H**(**T**) (i.e. **H** based on **T**) proves all PCA's true in $\mathcal{S}$. Unfortunately, this observation is not particularly helpful, because $Th(\mathcal{S})$ is not effectively axiomatizable for most structures $\mathcal{S}$ of interest.

Consider then the opposite extreme case, where **T** is empty, i.e. the Consequence Rule invokes only implications that are provable in first order logic, no axioms added. Even though Dynamic Logic has a far more complex proof theory than Hoare Logic, it is not immediately obvious that this difference should manifest itself in more PCA's being proved. In general, extending a formalism with extra expressive or deductive power is no guarantee that new theorems of a simple form are proved.[3]

We shall show (Theorem 2) that, in fact, **DL** based on the empty theory proves far more PCA's than Hoare's Logic based on the empty theory. To focus on the essentials, we formulate this and subsequent theorems for a simple programming language, namely regular programs with first-order tests and assignments as atomic actions. (Guarded iterative programs, i.e. **while** programs, are definable in terms of these regular programs.) Also, to facilitate comparisons, we restrict our attention to the natural numbers as the only data type.

Since **DL** is conservative over **HL** when the background theory **T** is as strong and possible, and not conservative when **T** is empty, it is natural to ask whether there is a theory $\mathbf{T}_0$ that demarcates a transition between these two states of affair. In Theorem 3 we prove that Peano Arithmetic **PA** is such a transition point: **DL**(**T**) is conservative over **H**(**T**) for all extensions **T** of **PA**, but not when **T** is **PA** with a bound on the complexity of induction formulas.

Peano Arithmetic, albeit natural and transparent, is a surprisingly powerful theory, because it allows Induction for arbitrarily complex formulas. Thus, for all practical purposes Dynamic Logic is not conservative over Hoare's Logic. One would wish, then, to identify a variant of Dynamic Logic that is conservative over Hoare Logic for *any* background theory. In Theorem 7 we identify such a variant, obtained simply by restricting Segerberg's Induction to first order eigen-formulas, that is, with no reference to programs.

Theorem 7 is our main result, both technically and for its practical potentials. The main difficulty is in showing that Segerberg's Induction,

$$[\alpha^*](\varphi \to [\alpha]\varphi) \to (\varphi \to [\alpha^*]\varphi)$$

[3] For example, extending Peano Arithmetic with all true Π^0_1 sentences yields a theory vastly more powerful than **PA**, but without any new provably recursive functions [7].

(with φ program-free) does not prove more PCA's than the Invariance Rule of Hoare Logic for regular programs,

$$\frac{\varphi[\alpha]\varphi}{\varphi[\alpha^*]\varphi}$$

This is far from obvious, because the premise of the Invariance Rule requires φ to be an invariant of α in all states, whereas in the premise of Segerberg's Induction φ needs to be invariant under α just in states reached by iterated execution of α.

2 Dynamic Logic and Arithmetic

2.1 Dynamic Logic over Regular Programs

To focus on the essentials, we refer to the simplest non-trivial imperative programming language, namely regular programs over assignments, with first-order tests [12,6]. That is, we fix a vocabulary V (a finite set of function and relational identifiers, each assigned a non-negative integer as an arity). Let A be the set of *V-assignments,* that is expressions of the form $x := \mathbf{t}$, where $\mathbf{t}$ is a V-term. The set P of V-regular programs is generated inductively by the following clauses for abstract syntax.

$$\begin{array}{ll} A \ni a & (V\text{-assignments}) \\ \Phi \ni F & (\text{first order } V\text{-formulas}) \\ P \ni \alpha ::= a \mid ?F \mid \alpha;\alpha \mid \alpha \cup \alpha \mid \alpha^* & \end{array}$$

As usual, guarded iterative programs ("while programs") are definable by programs in P: **skip** $\equiv ?\top$, **abort** $\equiv ?\bot$, (**if** F **then** α) **else** $\beta \equiv (?F;\alpha) \cup (?\neg F;\beta)$, and (**while** F **do** α) $\equiv (?F;\alpha)^*;(?\neg F)$. Given a V-structure $\mathcal{S}$, the semantics of programs α is defined by a straightforward recurrence on the complexity of α (see e.g. [6]).

A DL formula of the form $\varphi \to [\alpha]\psi$, with φ and ψ first-order, is said to be a *partial-correctness assertion (PCA).* It is often useful to abbreviate the formula above by $\varphi[\alpha]\psi$. The first-order formula φ is dubbed the PCAs *pre-condition*, and ψ is its *post-condition.*

The following are the rules of the deductive calculus **DL** for Dynamic Logic, to be added to a deductive calculus for first order logic. The rules are due to Segerberg, who formulated them for Propositional Dynamic Logic (see [12], [6, §5.5]).

I Axiom-templates and rules for programs in general. These are the rules of the rudimentary modal logic **K**:

Box Distribution: $[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi)$.

Generalization (Necessitation): $\dfrac{\vdash \varphi}{\vdash [\alpha]\varphi}$

II Rules for atomic programs; these define the intended meaning of atomic programs in terms of first order logic:

$$\text{Assignment: } [x := \mathbf{t}]\varphi[x] \leftrightarrow \varphi[\mathbf{t}]$$

$$\text{Test: } [?\chi]\,\varphi \leftrightarrow (\chi \to \varphi)$$

III Syntax directed rules for regular-program constructs; these relate the meaning of $[\alpha]$ for a compound program α to the meaning of $[\beta]$ for the immediate components β of α.

$$\text{Composition: } [\alpha;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi$$

$$\text{Branching: } [\alpha \cup \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\beta]\varphi$$

$$\text{Iteration: } [\alpha^*]\varphi \leftrightarrow \varphi \wedge [\alpha][\alpha^*]\varphi$$

IV Induction; this schema conveys the inductive meaning of the iteration operator $*$:

$$[\alpha^*](\varphi \to [\alpha]\varphi) \to (\varphi \to [\alpha^*]\varphi)$$

If $\mathbf{T}$ is a first order V-theory, we write $\mathbf{DL}(\mathbf{T})$ for the deductive calculus above, based on first order logic, and using $\mathbf{T}$ as axioms. We refer to $\mathbf{T}$ as the *background theory.*

2.2 An Interpretation of Peano Arithmetic in Dynamic Logic

While Dynamic Logic is intended as a logical formalism, that is with arbitrary relational structures as potential universes, the semantic of iteration is defined as standard iteration, i.e. with natural numbers as counters. This makes it possible to enforce natural numbers as values for variables. Namely, the modal operator $[\mathbf{N}(x)]$, where $\mathbf{N}(x)$ is the program $x := \mathbf{0}; (x := \mathbf{s}(x))^*$ (with $\mathbf{s}$ denoting the successor function), forces x to range precisely over the natural numbers in its scope. That is, $[\mathbf{N}(x)]\varphi$ is true in a structure $\mathcal{S}$ and an environment η therein iff the formula $\forall x.\ N(x) \to \varphi$ is true in that environment with the unary identifier N interpreted as the set of denotations in $\mathcal{S}$ of the numerals $\mathbf{0}, \mathbf{s}(\mathbf{0}), \mathbf{s}(\mathbf{s}(\mathbf{0})), \ldots, \mathbf{s}^{[n]}(\mathbf{0}), \ldots$.

*Peano Arithmetic (***PA***)* is the first order theory over the vocabulary consisting of identifiers for $\mathbf{0}$, $\mathbf{s}$ (the successor function), $+$ and $\times$. There are three groups of axioms:

1. The two separation axioms for $\mathbb{N}$, i.e. Peano's Third and Fourth Axioms

$$\forall x.\, \mathbf{s}(x) \neq \mathbf{0} \qquad \text{and} \qquad \forall x, y.\, \mathbf{s}(x) = \mathbf{s}(y) \to x = y$$

2. The defining recurrence equations for addition and multiplication.
3. All instances of the schema of Induction,

$$\forall x.\, (\varphi[x] \to \varphi[\mathbf{s}x]) \to (\varphi[\mathbf{0}] \to \forall x.\, \varphi[x])$$

It will be convenient to consider a definitional extension of Peano Arithmetic, as follows. It is well known that there are canonical arithmetization of syntax for which the basic syntactic operations, such as term substitution and correctness of inferences, are represented by primitive recursive (in fact, elementary-recursive) functions. We posit function identifiers for these functions (as well as for the auxiliary functions used to define them), and stipulate that Peano Arithmetic has as axioms all defining primitive-recursion equations for these functions. We write $\boldsymbol{\nu}$ for the conjunction of the separation axioms and the defining equations for the functions of the theory.

We interpret **PA** in **DL** as follows, writing $\boldsymbol{\varphi}^D$ for the **DL** formula that interprets a **PA** formula $\boldsymbol{\varphi}$. An equation $\mathbf{t} = \mathbf{t}'$ is interpreted as itself. Our interpretation commutes with the propositional connectives: $(\neg\boldsymbol{\varphi})^D \equiv_{\text{df}} \neg(\boldsymbol{\varphi}^D)$, etc. Quantifiers are interpreted using the modal operators: $(\forall x.\boldsymbol{\varphi})^D \equiv_{\text{df}} [\mathbf{N}(x)](\boldsymbol{\varphi}^D)$, and $(\exists x.\boldsymbol{\varphi})^D \equiv_{\text{df}} \langle\mathbf{N}(x)\rangle(\boldsymbol{\varphi}^D)$.

Theorem 1. *Let $\boldsymbol{\varphi}$ be a closed formula in the language of* **PA** *as above.*

1. *$\boldsymbol{\varphi}$ is true in the standard model of arithmetic iff $\boldsymbol{\nu} \to \boldsymbol{\varphi}^D$ is valid.*
2. *$\boldsymbol{\varphi}$ is a theorem of* **PA** *iff $\boldsymbol{\nu} \to \boldsymbol{\varphi}^D$ is provable in* **DL**.

Proof Outline. The proof of (1) is by a straightforward induction on the structure of $\boldsymbol{\varphi}$.

The forward direction of (2) is proved by induction on the **PA** proof of $\boldsymbol{\varphi}$. The only interesting case is Induction, i.e. with $\boldsymbol{\varphi}$ of the form

$$\forall x\,(\boldsymbol{\psi}[x] \to \boldsymbol{\psi}[\mathbf{s}x]) \to (\boldsymbol{\psi}[\mathbf{0}] \to \forall x\,\boldsymbol{\psi}[x])$$

Then $\boldsymbol{\varphi}^D$ is

$$[x := \mathbf{0};\ (x := \mathbf{s}(x))^*](\boldsymbol{\psi}^D[x] \to \boldsymbol{\psi}^D[\mathbf{s}(x)]) \ \to\ \boldsymbol{\psi}^D[\mathbf{0}] \to [x := \mathbf{0};\ (x := \mathbf{s}(x))^*]\boldsymbol{\psi}[x]$$

By the Box-Distribution rule of **DL**, it suffices to prove

$$[(x := \mathbf{s}(x))^*](\boldsymbol{\varphi}^D[x] \to \boldsymbol{\varphi}^D[\mathbf{s}(x)]) \ \to\ \boldsymbol{\varphi}^D[x] \to [(x := \mathbf{s}(x))^*]\boldsymbol{\varphi}[x]$$

or, equivalently (by the Assignment Rule),

$$[(x := \mathbf{s}(x))^*](\boldsymbol{\varphi}^D[x] \to [x := \mathbf{s}(x)]\boldsymbol{\varphi}^D[x]) \ \to\ \boldsymbol{\varphi}^D[x] \to [(x := \mathbf{s}(x))^*]\boldsymbol{\varphi}[x]$$

But this is an instance of Segerberg's Induction Schema of **DL**.

The backward direction of (2) is proved by interpreting **DL** in Peano's Arithmetic. This has been done, e.g., in [1,3,4]. ⊣

Note that our interpretation of **PA** in **DL** does not use quantification in **DL**. This is because all basic data is generated inductively, and so can be referred to as the output of a program. This does not show, however, that quantification is generally redundant in Dynamic Logic. For example, **DL** specifications for programs over graphs would naturally use quantifiers over vertices and edges.

The use of the nondeterministic * operator in the program $\mathbf{N}(x)$ is not essential here. We could use instead the deterministic program

$$\mathbf{N}'(x) \quad \equiv \quad y := x;\ \textbf{while}\ y \neq 0\ \textbf{do}\ y := y - 1\ \textbf{end}$$

We would then interpret $\forall x.\boldsymbol{\varphi}$ by $\forall x.\ [\mathbf{N}'(x)]\,\boldsymbol{\varphi}^D$. Of course, in the absence of nondeterministic program constructs we can no longer dispense with quantifiers.

2.3 Interpreting Inductive Algebras

We can use DL modalities to force variables to range over any given inductively generated algebra. For example, the set Σ^* of words over a finite alphabet Σ can be identified with the free algebra generated from the constant $\boldsymbol{\varepsilon}$, denoting the empty word, and, for each $a \in \Sigma$, a unary function identifier a. For the case $\Sigma = \{0, 1\}$ the constructors are the 0-ary $\boldsymbol{\varepsilon}$ and the unary $\mathbf{0}$ and $\mathbf{1}$. A word such as 011 is represented in the algebra as $\mathbf{0}(\mathbf{1}(\mathbf{1}(\boldsymbol{\varepsilon})))$.

Define now a program analogous to $\mathbf{N}(x)$:

$$\mathbf{W}_{\{0,1\}}(x) \quad \equiv \quad x := \boldsymbol{\varepsilon};\ ((x := \mathbf{0}(x)) \cup (x := \mathbf{1}(x)))^*$$

Then $[\mathbf{W}_{\{0,1\}}(x)]\ \boldsymbol{\varphi}$ is true in a structure $\mathcal{S}$ and environment η therein exactly when $\boldsymbol{\varphi}$ is true for all denotations of terms representing $\{0, 1\}^*$.

The definition is similar for arbitrary word algebras Σ, and, indeed, for any free algebra. Multi-sorted free algebras can also be represented by such iterative programs. For example, to have x range over the algebra of lists over $\mathbb{N}$, with Λ denoting NIL and $\mathbf{c}$ denoting **cons**, we use the program

$$\mathbf{L}_N(x) \quad \equiv \quad x := \Lambda\ ;\ (\ y := \mathbf{0}\ ;\ (y := \mathbf{s}(y))^*;\ x := \mathbf{c}(y, x))^*$$

It is not hard to prove for every inductive algebra (even multi-sorted) statements analogous to the two parts of Theorem 1. We shall not have use for these generalizations here.

3 Dynamic Logic vs. Hoare's Logic: The Role of the Background Theory

3.1 Hoare's Logic for Regular Programs

Let V be a vocabulary, and $\mathbf{T}$ a V-theory, all of whose axioms are closed formulas. We define a Hoare calculus $\mathbf{H}^*(\mathbf{T})$ for reasoning about PCAs for regular V-programs with assignments. The distinctive feature of a Hoare logic is the reference to only PCAs and first-order V-formulas.

In the following, $\vdash$ stands for provability in first-order logic (for example, using natural deduction derivations). The inference rules are as follows.

Assignment	$\{\mathbf{t}/x\}\varphi \ \ [x := \mathbf{t}] \ \ \varphi$
Composition	$\dfrac{\psi \ [\alpha] \ \chi \qquad \chi \ [\beta] \ \varphi}{\psi \ [\alpha;\beta] \ \varphi}$
Branching	$\dfrac{\psi \ [\alpha] \ \varphi \qquad \psi \ [\beta] \ \varphi}{\psi \ [\alpha \cup \beta] \ \varphi}$
Iteration	$\dfrac{\varphi \ [\alpha] \ \varphi}{\varphi \ [\alpha^*] \ \varphi}$
Query	$\dfrac{\psi \wedge \chi \to \varphi}{\psi[?\chi]\varphi}$ χ quantifier-free
Pre-consequence	$\dfrac{\psi \ [\alpha] \ \varphi \qquad \mathbf{T} \vdash \psi' \to \psi}{\psi' \ [\alpha] \ \varphi}$
Post-consequence	$\dfrac{\psi \ [\alpha] \ \varphi \qquad \mathbf{T} \vdash \varphi \to \varphi'}{\psi \ [\alpha] \ \varphi'}$

A formalism $\mathbf{H}(\mathbf{T})$ for reasoning about PCAs for guarded iterative programs is obtained by replacing the rules for Branching, Query, and Iteration by rules for the remaining program constructs of guarded iterative programs. The rules are exhibited in the following table.

Skip	φ [`skip`] φ
Abort	φ [`abort`] $\bot$
Cases	$\dfrac{(\psi \wedge \chi) \ [\alpha] \ \varphi \qquad (\psi \wedge \neg\chi) \ [\beta] \ \varphi}{\psi \ [\ \texttt{if}\ \chi\ \texttt{then}\ \alpha\ \texttt{else}\ \beta\ ] \ \varphi}$
Iteration	$\dfrac{(\varphi \wedge \chi) \ [\alpha] \ \varphi}{\varphi \ [\texttt{while}\ \chi\ \ \texttt{do}\ \alpha] \ (\varphi \wedge \neg\chi)}$

If α is a regular V-program and $\mathbf{T}$ is a V-theory, we write $\mathbf{T} \vdash_H \psi \ [\alpha] \ \varphi$ when $\psi \ [\alpha] \ \varphi$ is derivable in $\mathbf{H}^*(\mathbf{T})$.

3.2 The Cases of Maximal and Minimal Background Theories

The largest possible first-order theory $\mathbf{T}$ for the structure $\mathcal{N}$ is the set $Th(\mathcal{N})$ of all V-formulas that are true in the standard model $\mathcal{N}$ for the natural numbers (with function identifiers interpreted as the primitive recursive functions they are intended to denote). For this choice of $\mathbf{T}$, $\mathbf{H}(\mathbf{T})$ proves exactly the PCA's that are true in $\mathcal{N}$, by Cook's Relative Completeness Theorem [2]. Since $\mathbf{DL}(\mathbf{T})$ is sound for $\mathcal{N}$, it cannot possibly prove additional PCA's.

At the other extreme we have as $\mathbf{T}$ the empty theory. The proof theory of **DL** is far richer and more complex than that of Hoare's Logic, but (as discussed in the Introduction) this by itself does not necessarily imply that more PCA's are proved in **DL**. The needed link between proof theoretic power and PCA's is provided by the following.

Theorem 2. *In the absence of a background theory, first order dynamic logic is not conservative over Hoare's Logic: there are PCA's that are provable in Dynamic Logic, but not in Hoare's Logic.*

Proof Outline. For $k \geqslant 0$, let $\mathbf{PA}_k$ be Peano Arithmetic with Induction restricted to Π_k formulas. Let $\boldsymbol{\chi}$ be a universal sentence of the form $\forall x.\, \mathbf{t}[x] = \mathbf{0}$ which is a theorem of **PA** but not of $\mathbf{PA}_1$, for example a sentence expressing the consistency of $\mathbf{PA}_1$, i.e. the fact that no x codes a proof for $\mathbf{PA}_1 \vdash \bot$. Referring to the interpretation above of **DL** in **PA**, $\boldsymbol{\chi}^D$ is the PCA $[\mathbf{N}(x)](\mathbf{t} = \mathbf{0})$, and so the PCA

$$\boldsymbol{\pi} \quad \equiv_{\mathrm{df}} \quad \boldsymbol{\nu}[\mathbf{N}(x)](\mathbf{t} = \mathbf{0})$$

is provable in $\mathbf{DL}(\emptyset)$.

Towards contradiction, suppose that $\boldsymbol{\pi}$ is provable in $\mathbf{H}(\emptyset)$. By [8], there are then first order formulas $\boldsymbol{\xi}_i$ for which the first order formula

$$(\boldsymbol{\nu} \wedge \wedge_i (Cl_N[\boldsymbol{\xi}_i] \rightarrow \boldsymbol{\xi}_i[x])) \;\rightarrow\; \mathbf{t}[x] = \mathbf{0}$$

is provable (in first order logic!), where

$$Cl_N[\xi] \;\equiv_{\mathrm{df}}\; \xi[\mathbf{0}] \wedge \forall z.\xi[z] \rightarrow \xi[\mathbf{s}z]$$

For each numeral $\bar{n}$, the formula $Cl_N[\boldsymbol{\xi}_i] \rightarrow \boldsymbol{\xi}_i[\bar{n}]$ has a trivial n-step proof, so for each $n \in \mathbb{N}$ we obtain a first order proof D_n of $\boldsymbol{\nu} \;\rightarrow\; \mathbf{t}[\bar{n}] = \mathbf{0}$. Moreover, normalizing D_n yields a proof purely in the language of Peano Arithmetic. This entire argument is formalizable in $\mathbf{PA}_1$, with the conclusion that $\mathbf{PA}_1$ proves $\forall x.\mathbf{t}[x] = \mathbf{0}$, contradicting the choice of $\mathbf{t}$. ⊣

Note. The use of the nondeterministic program $\mathbf{N}(x)$ is, again, inessential here; the argument above can be modified to use the program $\mathbf{N}'(x)$ instead, albeit with some loss of elegance.

3.3 The Boundary of Conservativeness Is Peano's Arithmetic

We have seen that **DL** is conservative over **H** in the presence of the complete first order theory $Th(\mathcal{N})$ of $\mathcal{N}$, but is not conservative over **H** when the background theory is empty. It is natural to ask for a transition point.

Since the proof theoretic power of **DL** itself is akin to that of Peano Arithmetic, as illustrated by the results of [1,3,4], it is not surprising that the transition point is Peano Arithmetic.

Theorem 3. *Let* $\mathbf{T}$ *be a subtheory of* $Th(\mathcal{N})$.

1. *If* $\mathbf{T}$ *ccontains Peano Arithmetic, then* $\mathbf{DL}(\mathbf{T})$ *is conservative over* $\mathbf{H}(\mathbf{T})$.
2. $\mathbf{DL}(\mathbf{PA}_k)$ *(or even* $\mathbf{DL}(\emptyset)$*) is not conservative over* $\mathbf{H}(\mathbf{PA}_k)$.

Proof Outline. A straightforward approach for proving (1) is to emulate the proof of Cook's Relative Completeness Theorem, replacing the property "true in $\mathcal{N}$" by "provable in $\mathbf{DL}(\mathbf{T})$." However, this approach cannot work verbatim: our axiomatization of DL (contrary to Harel's [5]) allows non-standard models for $\mathbf{DL}(\mathbf{T})$ (no matter what $\mathbf{T}$ is), and so we cannot have a provable variant of expressiveness: there is no first order formula $\boldsymbol{\psi}[x]$ such that $\mathbf{DL}(\mathbf{T}) \vdash \boldsymbol{\psi}[x] \leftrightarrow \langle N(y)\rangle\ y = x$.

Thus, we take a slightly different approach, and refer directly to first order rendition of program semantics, rather than of weakest-preconditions. For each regular program $\boldsymbol{\alpha}$, over variables $\vec{x}$, one defines a first order formula $\boldsymbol{\mu}_\alpha[\vec{x}, \vec{v}]$ of **PA** that coveys the input/output semantics of the program $\boldsymbol{\alpha}$ over the structure $\mathcal{N}$. For instance, $\boldsymbol{\mu}_{\alpha;\beta}[\vec{x}, \vec{v}]$ is defined as $\exists \vec{z}\,.\, \boldsymbol{\mu}_\alpha[\vec{x}, \vec{z}] \wedge \boldsymbol{\mu}_\beta[\vec{z}, \vec{v}]$, and $\boldsymbol{\mu}_{\alpha^*}[\vec{x}, \vec{v}]$ is defined from $\boldsymbol{\mu}_\alpha$ using sequence-coding. By induction on $\boldsymbol{\alpha}$, one then proves that for all first-order formulas $\boldsymbol{\varphi}$,

$$\mathbf{H}(\mathbf{PA}) \vdash (\forall \vec{v}\,.\, \boldsymbol{\mu}_\alpha[\vec{x}, \vec{v}] \to \boldsymbol{\varphi}[\vec{v}])\ \ [\boldsymbol{\alpha}]\ \ \boldsymbol{\varphi}[\vec{x}] \tag{1}$$

Also, all axioms of **DL** become provable in **PA** under the interpretation of formulas $([\boldsymbol{\alpha}]\boldsymbol{\xi})[\vec{y}]$ as $\forall \vec{v}\,.\, \boldsymbol{\mu}_\alpha[\vec{y}, \vec{v}] \to \boldsymbol{\xi}[\vec{v}]$; and the generalization rule of **DL** becomes a derived rule of **PA** under that interpretation. Since $\mathbf{T}$ contains **PA**, it follows that if $\mathbf{DL}(\mathbf{T})$ proves a PCA $\boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\psi}$, where the free variables in $\boldsymbol{\alpha}$ are among $\vec{x}$, then the formula $\boldsymbol{\varphi} \wedge \boldsymbol{\mu}_\alpha[\vec{x}, \vec{v}] \to \{\vec{v}/\vec{x}\}\boldsymbol{\psi}$ is provable in $\mathbf{T}$. Combining this with (1), we obtain by the rule of Pre-Consequence of **H** that $\boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\psi}$ is provable in $\mathbf{H}(\mathbf{PA})$.

The proof of (2) is similar to the proof of Theorem 2. ⊣

4 A Dynamic Logic Conservative over Hoare's Logic

4.1 The Dynamic Logic $\mathbf{DL}_{\mathrm{fo}}$

The expressiveness of Dynamic Logic is far greater than that of Hoare's Logic, a gain which is valuable on many counts. For example, consider the extension

of Hoare's Logic to account for recursive procedures. The added inference rules refer (at least implicitly) to implications between PCA's, rather than only to PCA's and first order formulas. However, Hoare's logic has the advantage of being syntax directed, so formal program annotations may be viewed as a syntactic variant of Hoare's Logic. Reconciling Dynamic Logic and Hoare's Logic is therefore of both theoretical and practical interest.

We have seen in Theorem 3 that this can be done when the background theory is as powerful as **PA**. However, from a computational viewpoint, Peano Arithmetic is an exceedingly powerful theory. Theorem 3 shows, therefore, that from a practical viewpoint Dynamic Logic is far too powerful.

We therefore consider an alternative approach for reconciling Dynamic Logic with Hoare Logic, namely weakening the deductive power of Dynamic Logic, to match that of Hoare's Logic. It is natural to consider here Dynamic Logic with Segerberg's Induction restricted to first-order (i.e. program free) formulas. We write $\mathbf{DL}_{\mathrm{fo}}$ for **DL** so restricted.

At first blush, it is far from clear that this restriction is going far enough. Segerberg's Induction,

$$[\boldsymbol{\alpha}^*](\boldsymbol{\varphi} \to [\boldsymbol{\alpha}]\boldsymbol{\varphi}) \to (\boldsymbol{\varphi} \to [\boldsymbol{\alpha}^*]\boldsymbol{\varphi})$$

even with $\boldsymbol{\varphi}$ program-free, has a premise that refers to states reachable by iterated execution of $\boldsymbol{\alpha}$. This premise is weaker than the premise of the Invariance Rule of Hoare Logic,

$$\frac{\boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\varphi}}{\boldsymbol{\varphi}[\boldsymbol{\alpha}^*]\boldsymbol{\varphi}}$$

Thus, the restricted form of Segerberg's induction is stronger than the Invariance Rule. Nonetheless, we show that when it comes to proving PCA's, $\mathbf{DL}_{\mathrm{fo}}$ is conservative over Hoare's Logic, *regardless* of the background theory.

This said, we will use an auxiliary deductive calculus $\mathbf{DL}_{\mathrm{iter}}$, formally weaker than $\mathbf{DL}_{\mathrm{fo}}$, in which Segerberg's Induction is replaced by a rule of iteration:

$$\frac{\vdash \boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\varphi}}{\vdash \boldsymbol{\varphi}[\boldsymbol{\alpha}^*]\boldsymbol{\varphi}}$$

We will not only tie Hoare's Logic with $\mathbf{DL}_{\mathrm{fo}}$, but with a weak formalism for second order logic, namely the Gentzen-Prawitz deductive calculus for logic with relational quantification, but with comprehension restricted to first order formulas.

4.2 Explicit Rendition of Program Semantics

The operational semantics of programs $\boldsymbol{\alpha} \in P$ can be defined explicitly within an extension of first order logic with relational variables and quantification over them (rather than via a numeric coding, as in the definition of the formulas $\boldsymbol{\mu}_\alpha$ above). For each program $\boldsymbol{\alpha}$ whose variables are among $\vec{x} = x_1 \ldots x_k$, we define a formula $M^k_\alpha \equiv M^k_\alpha[\vec{x}, \vec{v}]$ with free variables among the $2k$ distinct variables

$\vec{x}$ and $\vec{v} = v_1 \dots v_k$, with the following property. For every V-structure $\mathcal{S}$, and every environment $\boldsymbol{\eta}$ therein, $\mathcal{S}, \eta \models M^k_\alpha[\vec{x}, \vec{v}]$ iff there is an execution of $\boldsymbol{\alpha}$ starting in environment η and terminating in environment $\eta[\vec{x} := \eta\vec{v}]$.

$$
\begin{array}{lll}
\text{For } \boldsymbol{\alpha} \equiv \; x_i := t[\vec{x}] & M^k_\alpha[\vec{x}, \vec{v}] & \equiv \; v_i = t[\vec{x}] \wedge \bigwedge_{j \neq i} v_j = x_j \\
\boldsymbol{\alpha} \equiv \; ?q[\vec{x}] & M^k_\alpha[\vec{x}, \vec{v}] & \equiv \; q[\vec{x}] \wedge \vec{v} = \vec{x} \\
\boldsymbol{\alpha} \equiv \; \boldsymbol{\beta};\boldsymbol{\gamma} & M^k_\alpha[\vec{x}, \vec{v}] & \equiv \; \exists \vec{u}\,. M^k_\beta[\vec{x}, \vec{u}] \wedge M^k_\gamma[\vec{u}, \vec{v}] \\
\boldsymbol{\alpha} \equiv \; \boldsymbol{\beta} \cup \boldsymbol{\gamma} & M^k_\alpha[\vec{x}, \vec{v}] & \equiv \; M^k_\beta[\vec{x}, \vec{v}] \vee M^k_\gamma[\vec{x}, \vec{v}] \\
\boldsymbol{\alpha} \equiv \; \boldsymbol{\beta}^* & M^k_\alpha[\vec{x}, \vec{v}] & \equiv \; \forall Q.\, Q(\vec{u}) \wedge \mathrm{Cl}^k_\beta[Q] \;\rightarrow Q(\vec{v}) \\
\text{where} & \mathrm{Cl}^k_\beta[Q] & \equiv \; \forall \vec{z}, \vec{w}. \\
 & & \quad Q(\vec{z}) \wedge M^k_\beta[\vec{z}, \vec{w}] \rightarrow Q(\vec{w})
\end{array}
$$

We omit the superscript k when in no danger of confusion.

4.3 Explicit Rendition of Dynamic Logic Formulas

It is obvious how to use the formulas M_α above to obtain an explicit rendition "$\boldsymbol{\varphi}$" for each DL formula $\boldsymbol{\varphi}$. Namely, if all variables in $\boldsymbol{\alpha}$ are among $\vec{x} = x_1 \dots x_k$, then $[\boldsymbol{\alpha}]\boldsymbol{\varphi}$ is rendered by

$$\text{“}[\boldsymbol{\alpha}]\boldsymbol{\varphi}\text{”} \quad \equiv_{\mathrm{df}} \quad \forall \vec{v}\,.\, M_\alpha[\vec{x}, \vec{v}] \rightarrow \text{“}\boldsymbol{\psi}[\vec{v}]\text{”}$$

Here $\vec{v} = v_1 \dots v_k$ are fresh variables, mutually distinct and different from $\vec{x}$, and $\boldsymbol{\varphi}[\vec{v}]$ stands for the result $\{\vec{v}/\vec{x}\}\boldsymbol{\varphi}$ of simultaneously substitution $\vec{v}$ in $\boldsymbol{\varphi}$ for all free occurrences of $\vec{x}$.

Now, given any deductive calculus $\mathbf{L}$ whose underlying language includes relational quantification, we can speak of the *partial correctness theory of* $\mathbf{L}$, $PCA(\mathbf{L})$, namely the collection of those PCAs $\boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\psi}$ whose explicit rendition "$\boldsymbol{\varphi}[\boldsymbol{\alpha}]\boldsymbol{\psi}$" is provable in $\mathbf{L}$. This definition has nothing to do with the proof theoretic strength of $\mathbf{L}$: it can be spectacularly powerful, say logic in all finite types, or extremely weak, such as second order logic with comprehension restricted to quantifier free formulas.

4.4 Proof Formalisms for Second Order Logic

Since the set of valid second order formulas is not RE, there is no sound and complete proof system for the language. Nonetheless, a widely used natural formalism for second order logic is obtained by extending first order logic with

variables and quantifiers over relations, and with the "set existence" (so-called Comprehension) schema

$$\forall \vec{x}.\exists R \forall \vec{u}.\ R(\vec{u}) \leftrightarrow \boldsymbol{\psi}[\vec{u}, \vec{x}] \qquad R \text{ not free in } \boldsymbol{\psi} \tag{2}$$

This schema may be conveyed by inference rules for relational quantifiers. For natural deduction the rules for universal quantification over relations are

$$\frac{\boldsymbol{\varphi}[Q]}{\forall R\, \boldsymbol{\varphi}[R]} \qquad \frac{\forall R\, \boldsymbol{\varphi}[R]}{\boldsymbol{\varphi}[\lambda \vec{x} \boldsymbol{\psi}])} \quad (\mathit{Var}(\boldsymbol{\psi}) \subseteq \vec{x})$$

Here $\boldsymbol{\varphi}[\lambda \vec{x} \boldsymbol{\psi}]$ stands for the result of replacing every subformula $R(\mathbf{t}_1 \ldots \mathbf{t}_k)$ of $\boldsymbol{\varphi}$ by $\{\vec{\mathbf{t}}/\vec{x}\}\boldsymbol{\varphi}$. For the introduction rule one stipulates that Q is not free in open assumptions,

Let $\mathbf{L}_2$ be the formalism for second order logic, as above. Trivially, $\mathbf{L}_2$ is sound for the standard semantics of the relational variables. Although not complete, $\mathbf{L}_2$ is a powerful formalism, e.g. full second order arithmetic (i.e. Classical Analysis) is interpretable in it (see e.g. [11]).

Of considerable interest are sub-formalisms of $\mathbf{L}_2$ in which comprehension is restricted to a class $\mathcal{C}$ of formulas, with no variables other than the ones referred to in the Comprehension Schema. Alternatively, one requires that the eigen-formula $\boldsymbol{\psi}$ in the rule of relational $\forall$-elimination, be in $\mathcal{C}$. We focus here on the case where comprehension is restricted to first order predicates, whose parameters are global to the proof. That is,

$$\exists R \forall \vec{u}.\ R(\vec{u}) \leftrightarrow \boldsymbol{\psi}[\vec{u}, \vec{x}] \qquad \boldsymbol{\psi} \text{ first-order, } R \text{ not free in } \boldsymbol{\psi} \tag{3}$$

The "global parameters" $\vec{x}$ will correspond to the free variables occurring in pre- and post-conditions of PCA's.

4.5 The Formalism $\mathbf{L}_2^0$.

We refer to first order logic with a distinction between *variables*, for which we use $u, v, w, u_i \ldots$, and *(global) parameters*, for which we use $x, y, x_i \ldots$.[4] Terms are built using both variables and parameters, but the parameters are "global to the proof", and are not quantified. This distinction is useful in conveying the restricted comprehension schema (3) by the inference rules below for relational quantification.

We refer to the following sequential calculus. By *sequent* we mean is a pair, written $\Gamma \Longrightarrow \Delta$, where Γ and Δ are finite sets of formulas. Such sequent is *initial* if $\Gamma \cup \Delta \neq \emptyset$. Referring to implication and universal quantification as the only logical operation aside from the logical constants $\top$ and $\bot$ (which is no loss

[4] Parameters will correspond to the free variables of pre- and post-conditions of PCA's. By referring to parameters we will be able to state an appropriate set existence principle, which refer to formulas with parameters, but no free variables.

of generality since the logic is classical), the inference rules are as follows. As usual, we write Γ, φ for $\Gamma \cup \{\varphi\}$ (note that $\varphi \in \Gamma$ is not excluded).

$$\frac{\Gamma, \varphi \Longrightarrow \Delta \quad \Gamma \Longrightarrow \Delta, \psi}{\Gamma, \psi \to \varphi \Longrightarrow \Delta} \qquad \frac{\Gamma, \psi \Longrightarrow \Delta, \varphi}{\Gamma \Longrightarrow \Delta, \psi \to \varphi}$$

$$\frac{\Gamma, \varphi[\mathbf{t}] \Longrightarrow \Delta}{\Gamma, \forall u.\varphi[u] \Longrightarrow \Delta} \qquad \frac{\Gamma \Longrightarrow \Delta, \varphi[v]}{\Gamma \Longrightarrow \Delta, \forall u.\varphi[u]}$$

$\mathbf{t}$ a term substitutable for u in φ — v not free in Γ, Δ

$$\frac{\Gamma, \varphi[\lambda \vec{u}.\psi] \Longrightarrow \Delta}{\Gamma, \forall R.\varphi[R] \Longrightarrow \Delta} \qquad \frac{\Gamma \Longrightarrow \Delta, \varphi[Q]}{\Gamma \Longrightarrow \Delta, \forall R.\varphi[R]}$$

R a relational variable — $\underline{arity}(Q) = \underline{arity}(R)$, Q not free in Γ, Δ

$\underline{arity}(\vec{z}) = \underline{arity}(R)$

ψ first-order with all free variables among $\vec{z}$ (arbitrary parameters allowed)

Theorem 4. *If a DL formula φ is provable in $\mathbf{DL}_{\mathrm{iter}}(\mathbf{T})$, then the second order formula "φ" is provable in $\mathbf{L}^0_2$ from $\mathbf{T}$.*

The proof is by induction on the proof of φ in $\mathbf{DL}_{\mathrm{iter}}$, and bears similarity to the proof in [8] that if a PCA P is provable in $\mathbf{DL}(\mathbf{T})$ then "P" is provable in $\mathbf{L}^0_2$ from $\mathbf{T}$.

4.6 $\mathbf{DL}_{\mathrm{iter}}$, $\mathbf{L}^0_2$, and Hoare's Logic

Theorem 5. *Let $\mathbf{T}$ be a first order theory, π a PCA. The following conditions are equivalent.*

1. *π is provable in $\mathbf{H}(\mathbf{T})$.*
2. *π is provable in $\mathbf{DL}_{\mathrm{iter}}(\mathbf{T})$.*
3. *"π" is provable in $\mathbf{L}^0_2$ from $\mathbf{T}$.*

Proof. (1) implies (2) trivially. Theorem 4 establishes that (2) implies (3). (3) implies (1) by the main result of [8]. (An alternative proof for the latter, using a different method, is give in [9].) ⊣

4.7 $\mathbf{DL}_{\mathrm{fo}}$ Is Conservative over Hoare's Logic

Theorem 6. *For every first order theory $\mathbf{T}$, $\mathbf{DL}_{\mathrm{fo}}(\mathbf{T})$ is conservative over $\mathbf{DL}_{\mathrm{iter}}(\mathbf{T})$ for PCA's.*

The proof uses an analysis of the structure of natural deduction derivations for $\mathbf{DL}_{\mathrm{fo}}$, and will be given in the full paper. (It bears similarity to the proof in [10, Lemma 2] that the one-quantifier induction schema is conservative over the one-quantifier induction rule for Π^0_2 sentences.)

Combining Theorems 6 and 5, we obtain

Theorem 7. *Let* $\mathbf{T}$ *be a first order theory.* $\mathbf{DL}_{\mathrm{fo}}(\mathbf{T})$ *is conservative over* $\mathbf{H}(\mathbf{T})$*; that is, every PCA provable in* $\mathbf{DL}_{\mathrm{fo}}(\mathbf{T})$ *is provable in* $\mathbf{H}(\mathbf{T})$*.*

References

1. J.A. Bergstra and J.V. Tucker. Hoare's Logic and Peano's Arithmetic. *Theoretical Computer Science*, 22:265–284, 1983.
2. Stephen A. Cook. Soundness and completeness of an axiom system for program verification. *SIAM J. Computing*, 7(1):70–90, 1978.
3. Petr Hajek. Arithmetical interpretations of Dynamic Logic. *Journal of Symbolic Logic*, 48:704–713, 1983.
4. Petr Hajek. A simple dynamic logic. *Theoretical Computer Science*, 46:239–259, 1986.
5. David Harel. *First-order Dynamic Logic.* LNCS 68. Springer-Verlag, Berlin, 1979.
6. David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic.* MIT Press, Cabridge, MA, 2000.
7. G. Kreisel. Mathematical logic. In T. Saaty, editor, *Lectures on Modern Mathematics*, volume III, pages 95–195. John Wiley, New York, 1965.
8. Daneil Leivant. Logical and mathematical reasoning about imperative programs. In *Conference Record of the Twelfth Annual Symposium on Principles of Programming Languages*, pages 132–140, New York, 1985. ACM.
9. J.A Makowsky and I. Sain. Weak second order characterizations of various program verification systems. *Theoretical Computer Science*, 66:299–321, 1989.
10. Charles Parsons. On n-quantifier induction. *The Journal of Symbolic Logic*, 37:466–482, 1972.
11. D. Prawitz. *Natural Deduction.* Almqvist and Wiksell, Uppsala, 1965.
12. Krister Segerberg. A completeness theorem in the modal logic of programs (preliminary report). *Notics of the American Mathematical Society*, 24(6):A–552, 1977.