15-812: Programming Language Semantics

Lecture Notes on Proofs About Terminating Loops

André Platzer

Carnegie Mellon University

Lecture 7

1 Introduction

The previous lectures gave us a good understanding of how to reason about $[\cdot]$ properties of programs. We saw how to decompose in logic properties of the form $[\alpha;\beta]P$ and $[\mathsf{if}(Q)\,\alpha\,\mathsf{else}\,\beta]P$ etc. That is all good and useful, but we need to remember that $[\alpha]P$ means that P holds after all runs of program α. Since while programs α are deterministic and, thus, have at most one run, the formula $[\alpha]P$, in particular, means that:

if α terminates, then P holds in the final state

But formula $[\alpha]P$ says nothing at all about whether the program terminates. Because it makes no statement about termination, this notion is also called *partial correctness* of program α with respect to postcondition P.

This is what the diamond modality $\langle\cdot\rangle$ is good for. The formula $\langle\alpha\rangle P$ says that there is a run of program α that reaches a final state in which postcondition P is true. Since while programs α have at most one run, the formula $\langle\alpha\rangle P$, in particular, means that:

α terminates *and* P holds in the final state.

Because it guarantees both a correct result and termination, this notion is also called *total correctness* of program α with respect to postcondition P.

Total correctness is the stronger notion compared to partial correctness, because it says that a program not only gives the right answer upon termination but also actually stops in finite time. Consequently, in order to understand total correctness and how to reason about total correctness of while programs, we investigate diamond modalities [**?**, **?**].

2 Diamond Axioms for Programs

Our approach to understanding programs with logic still is to design one reasoning principle for each program operator that describes its effect in logic with simpler logical operators. If we succeed doing that for every operator that a program can have, then we will understand even the most complicated programs just by repeatedly making use of the respective logical reasoning principles. Only this time it will be the combination of diamond modalities and the respective program operators we worry about.

2.1 Assignments

The easiest case to look into is what we need to prove in order to show the formula $\langle x := e\rangle p(x)$, which expresses that the assignment $x := e$ terminates and the formula $p(x)$ holds after the assignment $x := e$ that assigns the value of term e to variable x. How could we reduce this to another logical formula that is simpler?

Obviously assignments always terminate, because their only effect is to change the value of one variable which succeeds in (very) finite time. If we want to show that the formula $p(x)$ holds after assigning the new value e to variable x then we might as well show $p(e)$ right away. And, in fact, p is true of x after assigning e to x if and only if p is true of its new value e. That is, the formula $\langle x := e\rangle p(x)$ is equivalent to the formula $p(e)$. We capture this argument once and for all in the assignment axiom $\langle :=\rangle$:

$$\langle :=\rangle \quad \langle x := e\rangle p(x) \leftrightarrow p(e)$$

In the assignment axiom $\langle :=\rangle$, the formula $p(e)$ has the term e everywhere in place of where the formula $p(x)$ has the variable x. Of course, it is important for this substitution of e for x to avoid capture of variables and not make any replacements under the scope of a quantifier or modality binding an affected variable [**?**]. For example, the following formula is an instance of $\langle :=\rangle$:

$$\langle x := x^2 - 1\rangle x(x+1) \geq x + y \leftrightarrow (x^2-1)(x^2-1+1) \geq (x^2-1) + y$$

But the following is not because it would capture the replacement y that is used for x:

$$\langle x := y\rangle (x \geq 0 \land \forall y\, (x \geq y)) \leftrightarrow (y \geq 0 \land \forall y\, (y \geq y))$$

Instead, if we first rename $\forall y$ to $\forall z$ then the substitution works:

$$\langle x := y\rangle (x \geq 0 \land \forall z\, (x \geq z)) \leftrightarrow (y \geq 0 \land \forall z\, (y \geq z))$$

Indeed by combining the $[:=]$ and $\langle :=\rangle$ axioms, one can also show that both modalities are equivalent for assignments:

$$[x := e]p(x) \leftrightarrow \langle x := e\rangle p(x)$$

This makes sense because an assignment always terminates and has exactly one successor, so modalities quantifying over all or one successor actually express the same fact for assignments. But such an equivalence will not generally hold for other program operators!

Recall one implication for deterministic programs, which shows that total correctness implies partial correctness.

Lemma 1 (Deterministic program modality relation)**.** *Deterministic programs α make the following formula valid for all formulas P:*

$$\langle\alpha\rangle P \rightarrow [\alpha]P$$

There is no reason to believe both sides would be equivalent in general, because there are many programs that are partially correct just never terminate. In fact, can you find a program that is partially correct for all preconditions A and postconditions B?

Finally, since box and diamond modalities are equivalent in the case of assignments, the equational form of an assignment proof rule not only works for boxes but also for diamonds:

$$\langle{:=}\rangle_= \ \frac{\Gamma, y = e \vdash p(y), \Delta}{\Gamma \vdash \langle x := e\rangle p(x), \Delta} \quad (y \text{ new})$$

2.2 Conditionals

The next case we choose to look at is what we need to prove in order to show the formula $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$, which expresses that formula P holds in some final state reached after running the if-then-else conditional $\mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta$ that runs program α if formula Q is true and runs β otherwise. And that this conditional program will indeed terminate. Of course, the if will terminate but the question is whether the resulting subprograms α and/or β terminate. In order to understand it from a logical perspective, how could we express $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ in easier ways?

If Q holds then $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ says that P holds in some final state after running α. If Q does not hold then the same formula $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ says that P holds in some final state after running β. It is easy to say with a logical formula that P holds in some state after running α, which is precisely what $\langle\alpha\rangle P$ is good for. Likewise $\langle\beta\rangle P$ directly expresses in logic that P in some state after running β. Both of those formulas $\langle\alpha\rangle P$ as well as $\langle\beta\rangle P$ are simpler than the original formula $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$. But, of course, they express something else, because the program $\mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta$ only runs the respective programs conditionally depending on the truth-value of Q.

Yet, there is a way of expressing $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ in logic in easier ways with the help of other logical operators. Implications are perfect at expressing the conditions that an if-then statement states in a program. Indeed, if Q holds then $\langle\alpha\rangle P$ needs to be true and if Q does not hold then $\langle\beta\rangle P$ for $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ to hold. Indeed, $\langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P$ is true if and only if $(Q \rightarrow \langle\alpha\rangle P) \land (\neg Q \rightarrow \langle\beta\rangle P)$ is true. This is the diamond formulation of the if-then-else axiom $\langle\text{if}\rangle$:

$$\langle\text{if}\rangle \quad \langle \mathsf{if}(Q)\, \alpha\, \mathsf{else}\, \beta\rangle P \leftrightarrow (Q \rightarrow \langle\alpha\rangle P) \land (\neg Q \rightarrow \langle\beta\rangle P)$$

This axiom tells us everything we need to know about correct termination of if-then-else statements. When using the equivalence ⟨if⟩ from left to right, we can use it to simplify every question about an if-then-else statement of the form $\langle \mathsf{if}(Q)\,\alpha\,\mathsf{else}\,\beta\rangle P$ by a corresponding structurally simpler formula $(Q \rightarrow \langle\alpha\rangle P) \land (\neg Q\langle\beta\rangle P)$ that does not use the if-then-else statement any more but is logically equivalent. The axiom ⟨if⟩ achieves the same kind of compositionality that axiom [if] achieves, just for diamond modalities of if-then-else statements. The ⟨if⟩ axiom will enable us, for example to conclude this equivalence:

$$\langle \mathsf{if}(x{\geq}0)\, y := x\, \mathsf{else}\, y := -x\rangle y{=}|x| \leftrightarrow (x{\geq}0 \rightarrow \langle y := x\rangle y{=}|x|) \land (\neg x{\geq}0 \rightarrow \langle y := -x\rangle y{=}|x|)$$

This formula uses $|x|$ as notation for the absolute value of x.

Since axiom ⟨if⟩ justifies this equivalence, we will be able to reduce a question about whether its left hand side is valid with axiom ⟨if⟩ to the question whether its corresponding right hand side is valid.

$$\langle\text{if}\rangle\dfrac{\land\text{R}\dfrac{\rightarrow\text{R}\dfrac{\langle:=\rangle\dfrac{\mathbb{Z}\dfrac{*}{x{\geq}0 \vdash x{=}|x|}}{x{\geq}0 \vdash \langle y := x\rangle\, y{=}|x|}}{\vdash x{\geq}0 \rightarrow \langle y := x\rangle\, y{=}|x|} \qquad \rightarrow\text{R}\dfrac{\langle:=\rangle\dfrac{\mathbb{Z}\dfrac{*}{\neg x{\geq}0 \vdash -x{=}|x|}}{\neg x{\geq}0 \vdash \langle y := -x\rangle\, y{=}|x|}}{\vdash \neg x{\geq}0 \rightarrow \langle y := -x\rangle\, y{=}|x|}}{\vdash (x{\geq}0 \rightarrow \langle y := x\rangle\, y{=}|x|) \land (\neg x{\geq}0 \rightarrow \langle y := -x\rangle\, y{=}|x|)}}{\vdash \langle \mathsf{if}(x{\geq}0)\, y := x\, \mathsf{else}\, y := -x\rangle\, y{=}|x|}$$

This proof shows validity of the following formula, which says that the given program is totally correct in implementing the absolute value function $|\cdot|$ from mathematics:

$$\langle \mathsf{if}(x{\geq}0)\, y := x\, \mathsf{else}\, y := -x\rangle\, y{=}|x|$$

As usual the proof is developed starting with the desired conclusion at the bottom and working with proof rules to the top as usual in sequent calculus.

2.3 Test

The test statement $?Q$ also checks a condition on the current state but has a different effect. It has no effect on the state if Q is indeed true, but aborts and discards the execution if Q is not true. In particular, in the latter case, the program did not terminate (rather it was aborted unsuccessfully). How can we express $\langle ?Q\rangle P$ in logic in structurally simpler ways?

The formula $\langle ?Q\rangle P$ is true iff formula P holds in some final state after running the test $?Q$, which only even has a final state if Q is true. Consequently P holds in some final state after running the program $?Q$ iff postcondition P is true and if the test Q is. This is captured in the test axiom ⟨?⟩:

$$\langle ?\rangle \quad \langle ?Q\rangle P \leftrightarrow (Q \land P)$$

Even if diamonds of deterministic programs imply their own boxes, the test axiom $\langle ? \rangle$ already alerts us to the fact that some programs are only partially correct but not totally correct. The following box formula is valid

$$[?x = 2]x \cdot x = 4$$

because by axiom $[?]$ it is equivalent to the valid formula $x = 2 \rightarrow x \cdot x = 4$. But the corresponding diamond formula is not valid:

$$\langle ?x = 2 \rangle x \cdot x = 4$$

because axiom $\langle ? \rangle$ makes it equivalent to $x = 2 \land x \cdot x = 4$ so $x = 2$, which is not valid but merely satisfiable.

2.4 Sequential Compositions

The next most pressing case to worry about are sequential compositions. So how can we equivalently express $\langle \alpha; \beta \rangle P$ in simpler logic without sequential compositions? This formula expresses that P holds after some runs of $\alpha; \beta$, which first runs α and then runs β. How can this be expressed in an easier way in logic, again using just the subprograms α as well as β of $\alpha; \beta$ then?

In order to express $\langle \alpha; \beta \rangle P$ what we need to say is that after some run of α it is the case that P holds after some run of β. It is comparably easy to say that P holds after some runs of β just with the formula $\langle \beta \rangle P$. But where does this formula need to hold? After some runs of α! In particular, all we need to say is that $\langle \beta \rangle P$ holds after some run of α, which is exactly what the formula $\langle \alpha \rangle \langle \beta \rangle P$ says. This is the sequential composition axiom $\langle ; \rangle$ for diamonds:

$$\langle ; \rangle \quad \langle \alpha; \beta \rangle P \leftrightarrow \langle \alpha \rangle \langle \beta \rangle P$$

Indeed, after all runs of $\alpha; \beta$ does P hold if and only if after all runs of α it is the case that after all runs of β does P hold.

This enables us to consider another absolute value function implementation that is partially correct but *not* totally correct! The following formula is easily seen to be valid:

$$[y := -x; ?y \geq 0]\, y = |x|$$

A proof of validity is easy from the box axioms and arithmetic facts about the absolute value function $|\cdot|$:

$$\begin{array}{rl}
 & * \\ \hline
\mathbb{Z} & \vdash -x \geq 0 \rightarrow -x = |x| \\ \hline
[?] & \vdash [?-x \geq 0]\, -x = |x| \\ \hline
[:=] & \vdash [y := -x][?y \geq 0]\, y = |x| \\ \hline
[;] & \vdash [y := -x; ?y \geq 0]\, y = |x|
\end{array}$$

A corresponding attempt to prove total correctness will, however, fail:

$$\langle;\rangle\,\dfrac{\langle:=\rangle\,\dfrac{\langle?\rangle\,\dfrac{\text{cut}\,\dfrac{\vdash -x \geq 0 \qquad \land\text{R}\,\dfrac{\text{id}\,\dfrac{*}{-x \geq 0 \vdash -x \geq 0} \qquad \mathbb{Z}\,\dfrac{*}{-x \geq 0 \vdash -x = |x|}}{-x \geq 0 \vdash -x \geq 0 \land -x = |x|}}{\vdash -x \geq 0 \land -x = |x|}}{\vdash \langle ?{-x} \geq 0\rangle\, -x = |x|}}{\vdash \langle y := -x\rangle\langle ?y \geq 0\rangle\, y = |x|}}{\vdash \langle y := -x; ?y \geq 0\rangle\, y = |x|}$$

In fact, the total correctness property is not valid:

$$\langle y := -x; ?y \geq 0\rangle\, y = |x|$$

Only if the formula $-x \geq 0$ in the remaining open proof branch is assumed in the initial state is the total correctness formula valid:

$$x \leq 0 \rightarrow \langle y := -x; ?y \geq 0\rangle\, y = |x|$$

3 Soundness

The above axioms can again all be shown to be sound. We only show the proof of one axiom in order to leave you sufficiently many other axioms to practice soundness proofs on.

Lemma 2. *The sequential composition axiom* $\langle;\rangle$ *is sound, i.e. all its instances are valid:*

$$\langle;\rangle \quad \langle\alpha;\beta\rangle P \leftrightarrow \langle\alpha\rangle\langle\beta\rangle P$$

Proof. Recall the semantics of sequential composition:

$$I[\![\alpha;\beta]\!] = I[\![\alpha]\!] \circ I[\![\beta]\!] = \{(\omega,\nu) \;:\; (\omega,\mu) \in I[\![\alpha]\!], (\mu,\nu) \in I[\![\beta]\!]\}$$

In order to show that the formula $\langle\alpha;\beta\rangle P \leftrightarrow \langle\alpha\rangle\langle\beta\rangle P$ is valid, i.e. $\vDash \langle\alpha;\beta\rangle P \leftrightarrow \langle\alpha\rangle\langle\beta\rangle P$, consider any state ω and show that $\omega \in I[\![\langle\alpha;\beta\rangle P \leftrightarrow \langle\alpha\rangle\langle\beta\rangle P]\!]$. We prove this biimplication by separately proving both implications.

"$\leftarrow$" Assume the right hand side $\omega \in I[\![\langle\alpha\rangle\langle\beta\rangle P]\!]$ and show $\omega \in I[\![\langle\alpha;\beta\rangle P]\!]$. To show the latter, we need to show that there is a state ν with $(\omega,\nu) \in I[\![\alpha;\beta]\!]$ for which $\nu \in I[\![P]\!]$. By the semantics of sequential composition, $(\omega,\nu) \in I[\![\alpha;\beta]\!]$ iff there is a state μ such that $(\omega,\mu) \in I[\![\alpha]\!]$ and $(\mu,\nu) \in I[\![\beta]\!]$. The assumption implies that there is a state μ with $(\omega,\mu) \in I[\![\alpha]\!]$ such that $\mu \in I[\![\langle\beta\rangle P]\!]$. The latter implies that there is a ν with $(\mu,\nu) \in I[\![\beta]\!]$ such that $\nu \in I[\![P]\!]$. Thus, $\nu \in I[\![P]\!]$ and, as desired, $(\omega,\nu) \in I[\![\alpha;\beta]\!]$, because $(\omega,\mu) \in I[\![\alpha]\!]$ and $(\mu,\nu) \in I[\![\beta]\!]$. Hence $\omega \in I[\![\langle\alpha;\beta\rangle P]\!]$.

"$\leftarrow$" Conversely, assume the left hand side $\omega \in I[\![\langle\alpha;\beta\rangle P]\!]$ and show $\omega \in I[\![\langle\alpha\rangle\langle\beta\rangle P]\!]$. Consequently, there is a state ν such that $(\omega,\nu) \in I[\![\alpha;\beta]\!]$ and $\nu \in I[\![P]\!]$. Now $(\omega,\mu) \in I[\![\alpha]\!]$ and $(\mu,\nu) \in I[\![\beta]\!]$ iff $(\omega,\nu) \in I[\![\alpha;\beta]\!]$ by the semantics of sequential composition. Hence, there is a state μ such that $(\omega,\mu) \in I[\![\alpha]\!]$ and $\mu \in I[\![\langle\beta\rangle P]\!]$. Thus, $\omega \in I[\![\langle\alpha\rangle\langle\beta\rangle P]\!]$. □

4 While Loops

The move to total correctness by investigating diamond modalities worked quite well, except that our understanding of loops is still lagging behind. Unwinding of loops is easy but not sufficient:

$$\langle\text{unwind}\rangle\quad \langle \mathsf{while}(Q)\,\alpha\rangle P \leftrightarrow \langle \mathsf{if}(Q)\,\{\alpha;\mathsf{while}(Q)\,\alpha\}\rangle P$$

Since not all loops have a fixed finite number of rounds, this axiom isn't quite sufficient. But that's similar to the case of box modalities, where we also first understood the elementary axioms reducing programs and later went for a study of loop invariants for while loops with unbounded repetitions. Our key to understanding what to do with $[\mathsf{while}(Q)\,\alpha]P$ formulas is to first understand induction principles for $[\alpha^*]P$ with more general nondeterministic repetitions α^* of program α, under complete ignorance of the loop guard. We will proceed in similar ways again for diamond properties of while loops.

5 Recap: Nondeterministic Repetitions

Recall the nondeterministic repetition α^* which expresses that the program α repeats any arbitrary unspecified nondeterministic number of times:

6. $I[\![\alpha^*]\!] = \{(\omega,\nu)$: there are an n and states $\mu_0 = \omega, \mu_1, \mu_2, \dots, \mu_n = \nu$ such that $(\mu_i, \mu_{i+1}) \in I[\![\alpha]\!]$ for all $0 \le i < n\}$
   That is, state μ_{i+1} is reachable from state μ_i by running α for all i.

Recall its core induction principle for $[\alpha^*]P$.

Lemma 3. *The induction axiom I is sound:*

$$\text{I}\quad [\alpha^*]P \leftrightarrow P \land [\alpha^*](P \to [\alpha]P)$$

The loop invariant rule for nondeterministic repetitions derives from this axiom:

$$\text{loop}\ \frac{\Gamma \vdash J, \Delta \quad J \vdash [\alpha]J \quad J \vdash P}{\Gamma \vdash [\alpha^*]P, \Delta}$$

6 Diamonds for Nondeterministic Repetitions

The induction axiom I led to a study of loop invariants, which can prove box properties of loops using loop invariants. As the name suggests, loop *invariants* are about properties that do *not* change as the loop runs. But if nothing ever changes, then how would that argue that the loop also makes progress toward eventually terminating? Of course, it wouldn't.

Lemma 4. *The convergence axiom is sound for programs with integer arithmetic:*

$$\text{C}\quad \begin{array}{l}[\alpha^*]\forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1))\\ \quad\to \forall n{\geq}0\,(p(n) \to \langle\alpha^*\rangle p(0))\end{array} \qquad (n \notin \alpha)$$

The proof of the soundness of the convergence axiom C is based on the observation that the value of n for which $p(n)$ is true will eventually decrease down to 0 after repeating α^* sufficiently often if it initially starts from a nonnegative $n \geq 0$, provided that after any number of repetitions of α^* and for any positive $n > 0$ for which $p(n)$ holds it is the case that there is a way of running one round of α to make $p(n-1)$ true. We refer to the literature for detail [**?**, **?**, **?**].

A program that does not terminate is also sometimes referred to as a diverging program.

Lemma 5. *The loop convergence proof rule* con *derives from axiom* C*:*

$$\text{con}\ \frac{\Gamma \vdash \exists n{\geq}0\,p(n), \Delta \quad p(n), n{>}0 \vdash \langle\alpha\rangle p(n-1) \quad p(0) \vdash P}{\Gamma \vdash \langle\alpha^*\rangle P, \Delta} \qquad (n\ \textit{fresh})$$

Proof. Because n does not occur in $\langle\alpha^*\rangle p(0)$, axiom C is equivalent to

$$\begin{array}{l}[\alpha^*]\forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1))\\ \to \big((\exists n{\geq}0\,p(n)) \to \langle\alpha^*\rangle p(0)\big)\end{array}$$

From this, we derive

$$\text{M}\ \dfrac{\text{C}\ \dfrac{\wedge\text{R}\ \dfrac{\Gamma \vdash \exists n{\geq}0\,p(n), \Delta \qquad \text{G}\ \dfrac{\forall\text{R},\to\text{R}\ \dfrac{p(n), n{>}0 \vdash \langle\alpha\rangle p(n-1)}{\vdash \forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1))}}{\Gamma \vdash [\alpha^*]\forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1)), \Delta}}{\Gamma \vdash \exists n{\geq}0\,p(n) \wedge [\alpha^*]\forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1)), \Delta}}{\Gamma \vdash \langle\alpha^*\rangle p(0), \Delta} \qquad p(0) \vdash P}{\Gamma \vdash \langle\alpha^*\rangle P, \Delta}$$

□

In order to get to know this proof rule better, let's do a few example proofs.

$$\to\text{R}\ \dfrac{\text{con}\ \dfrac{\mathbb{Z}\ \dfrac{*}{x{\geq}0 \vdash \exists n{\geq}0\, x = n} \qquad \langle{:=}\rangle\ \dfrac{\mathbb{Z}\ \dfrac{*}{x = n, n > 0 \vdash x - 1 = n - 1}}{x = n, n{>}0 \vdash \langle x := x - 1\rangle x = n - 1} \qquad \mathbb{Z}\ \dfrac{*}{x = 0 \vdash x = 0}}{x \geq 0 \vdash \langle(x := x - 1)^*\rangle x = 0}}{\vdash x \geq 0 \to \langle(x := x - 1)^*\rangle x = 0}$$

If the loop body decrements x by 2, of course, we cannot know that the value of x will eventually be 0 but merely that it will be between 0 (inclusive) and 2 (exclusive).

The loop variant $p(n) \overset{\text{def}}{\equiv} 2n \leq x < 2n+2$ relates the program variable x to the loop progress variable n appropriately.

$$\text{→R}\dfrac{\text{con}\dfrac{\mathbb{Z}\dfrac{*}{x\geq0 \vdash \exists n\geq0\, 2n\leq x<2n+2} \quad \langle:=\rangle\dfrac{\mathbb{Z}\dfrac{*}{2n\leq x<2n+2, n>0 \vdash 2n-2\leq x-2<2n}}{2n\leq x<2n+2, n>0 \vdash \langle x:=x-2\rangle 2n-2\leq x<2n} \quad \mathbb{Z}\dfrac{*}{0\leq x<2 \vdash 0\leq x<2}}{x\geq 0 \vdash \langle(x:=x-2)^*\rangle 0\leq x<2}}{\vdash x\geq 0 \rightarrow \langle(x:=x-2)^*\rangle 0\leq x<2}$$

Suppose the loop body still decrements by 2 but we also know that the value of x starts out even (written $2|x$) then we can prove that x is eventually 0 by a minor modification of the above proof, using evenness as an additional conjunct in the loop variant $p(n) \equiv 2|x \land 2n \leq x < 2n+2$:

$$\text{→R,∧L}\dfrac{\text{con}\dfrac{\mathbb{Z}\dfrac{*}{x\geq0, 2|x \vdash \exists n\geq0\, (2|x\land 2n\leq x<2n+2)} \quad \langle:=\rangle\dfrac{\mathbb{Z}\dfrac{*}{2|x \land 2n\leq x<2n+2, n>0 \vdash 2|x-2 \land 2n-2\leq x-2<2n}}{2|x\land 2n\leq x<2n+2, n>0 \vdash \langle x:=x-2\rangle(2|x\land 2n-2\leq x<2n)} \quad \mathbb{Z}\dfrac{*}{2|x\land 0\leq x<2 \vdash x=0}}{x\geq 0, 2|x \vdash \langle(x:=x-2)^*\rangle x=0}}{\vdash x\geq 0\land 2|x \rightarrow \langle(x:=x-2)^*\rangle x=0}$$

Note that the loop variant $p(n) \equiv 2|x \land 2n \leq x < 2n+2$ has a part that does not actually depend on the progress counter n. That's peculiar. What does it mean?

Some loop variants such as $p(n) \equiv q \land p(n)$ that consist of some conjuncts q that are independent of the loop progress counter n together with other conjuncts $p(n)$ that do depend on n. In that case, the n-independent part q plays a similar role as a loop invariant ass something that does not change, while only the the n-dependent part $p(n)$ is a proper progress condition that does change toward the eventual goal.

In the above example, it, indeed, was an invariant of the loop that x always remains even. The proper progress condition expressed that x is between $2n$ and $2n+2$ as a function of n, so making progress toward $0 \leq x < 2$ as n decreases.

7 Variants of Mixed Invariants and Variants

Using loop variants $p(n)$ with parts that depend on n (progress condition) and parts that do not (invariants) is one approach. And a perfectly reasonable and well-defined one, too. A fair number of program verification tools, however, instead expect separate declarations of loop invariants and loop variants. What could that possibly mean? How is it reasonable to worry separately about loop invariants and loop variants?

It does make some intuitive sense to worry separately first about partial correctness (if a program terminates then its result will have the correct outcome) and second about termination (the program indeed terminates) to finally show total correctness (the program always terminates and gives the correct answer, too). But how can such informal arguments be made rigorous, in order to make sure there's no flaw in our reasoning?

A partial correctness property of a loop shows that if formula A holds in the initial state then formula B always holds after running the loop α^*:

$$A \to [\alpha^*]B$$

A total correctness property of a loop is of the form:

$$C \to \langle\alpha^*\rangle D$$

How can we meaningfully combine two properties of total and partial correctness leading to mixed box and diamond modalities? Never minding the fact that we generally might need some preconditions A or C, respectively, to make the above partial or total correctness true, let's assume we have some way of justifying that a box property $[\alpha^*]B$ as well as a diamond property $\langle\alpha^*\rangle D$ are true in some state.

If $[\alpha^*]B$ and $\langle\alpha^*\rangle D$ are both true, then what do we know?

If B holds after all runs of α^* and D holds after at least one run of α^*, then the conjunction $B \land D$ also holds after at least one run of α^*. Consequently, a variant and an invariant, when established separately, should still hold jointly. We should make that rigorous! But what if an invariant, established separately, is needed during the proof of the variant? How is that a proof? We should make that rigorous, too!

So let's go one step at a time.

If we know that $\langle\alpha^*\rangle B$ and $\langle\alpha^*\rangle D$ then we don't know a whole lot because the first formula says there is a run of α^* to a state where B is true while the second formula says that there also is a run of α^* to a state where D is true. But both B and D could be true after a different number of iterations, so that $B \land D$ might never be true. Yet if we know $[\alpha^*]B$ and $\langle\alpha^*\rangle D$, then we know that $B \land D$ are true after some number of repetitions, because B is true after any number of iterations while D is true after some appropriate number of iterations, at which point B will also be true and hence their conjunction.

Lemma 6. *Kripke's modal modus ponens axioms are sound:*

$$\text{K} \quad [\alpha](P \to Q) \to ([\alpha]P \to [\alpha]Q)$$

$$\text{K}_{\langle\cdot\rangle} \quad [\alpha](P \to Q) \to (\langle\alpha\rangle P \to \langle\alpha\rangle Q)$$

Lemma 7. *The invariant to variant conversion is a derived axiom:*

$$\text{inv2var} \quad [\alpha]J \to (\langle\alpha\rangle(J \to p) \to \langle\alpha\rangle p)$$

Proof. The inv2var axiom proves using axiom $\text{K}_{\langle\cdot\rangle}$ and monotonicity rule $\text{M}[\cdot]$:

$$\cfrac{\cfrac{\cfrac{\cfrac{*}{J \vdash (J \to p) \to p}}{[\alpha]J \vdash [\alpha]\big((J \to p) \to p\big)}\,\text{M}[\cdot]}{[\alpha]J \vdash \langle\alpha\rangle(J \to p) \to \langle\alpha\rangle p}\,\text{K}_{\langle\cdot\rangle}}{\vdash [\alpha]J \to (\langle\alpha\rangle(J \to p) \to \langle\alpha\rangle p)}\,\to\text{R}$$

□

The inv2var axiom enables us to assume a proved invariant property during a diamond proof. A common special case of this is when we establish an invariant property and a variant property separately and just want to put them together, which is what the minor variation in derived axiom inv∧var supports.

Lemma 8. *The invariant plus variant conversion is a derived axiom:*

$$\text{inv}\land\text{var}\quad [\alpha]P \land \langle\alpha\rangle Q \to \langle\alpha\rangle(P \land Q)$$

Proof.

$$\cfrac{\cfrac{\cfrac{\cfrac{*}{Q \vdash P \to P \land Q}\,{\scriptstyle \to R,\land R,\text{id}}}{\langle\alpha\rangle Q \vdash \langle\alpha\rangle(P \to P \land Q)}\,{\scriptstyle M}}{[\alpha]P, \langle\alpha\rangle Q \vdash \langle\alpha\rangle(P \land Q)}\,{\scriptstyle \text{inv2var}}}{\vdash [\alpha]P \land \langle\alpha\rangle Q \to \langle\alpha\rangle(P \land Q)}\,{\scriptstyle \to R,\land L}$$

□

This proof uses the diamond formulation of the monotonicity rule, the dual to M[·]:

$$\text{M}\ \frac{P \vdash Q}{\Gamma, \langle\alpha\rangle P \vdash \langle\alpha\rangle Q, \Delta}$$

8 Total Correctness of While Loops

For nondeterministic repetitions α^*, the proof rule con based on the axiom C predicts the exact number of loop iterations by way of the variable n. This is unnecessarily precise for while loops ,$\mathsf{while}(Q)\,\alpha$ where the loop itself already conveys when it stops, namely exactly when the loop guard Q turns false.

For while loops it is, thus, often more convenient to work with a more lenient proof rule that just has a term p whose value decreases along the repetitions of the loop and stays nonnegative while the loop repeats. This proof rule, var, can be derived from the convergence rule con.

Lemma 9. *The variant proof rule for while loops is a derived rule:*

$$\text{var}\ \frac{\Gamma \vdash J, \Delta \quad J, Q, p = n \vdash \langle\alpha\rangle(J \land p < n) \quad J, Q \vdash p \geq 0 \quad J, \neg Q \vdash P}{\Gamma \vdash \langle\mathsf{while}(Q)\,\alpha\rangle P, \Delta}\quad (n\ \textit{fresh})$$

Proof. In order to relax the need for an exact loop repetition count in con, this proof uses the following equivalent definition of while loops: $\mathsf{while}(Q)\,\alpha \equiv \{\mathsf{if}(Q)\,\alpha\}^*; ?\neg Q$. It also uses a clever loop variant $p(n) \overset{\text{def}}{\equiv} J \land (Q \to p < n)$.

$$\cfrac{\cfrac{\cfrac{\Gamma \vdash \exists n{\geq}0\, p(n), \Delta \qquad p(n), n > 0 \vdash \langle\mathsf{if}(Q)\,\alpha\rangle p(n-1) \quad p(0) \vdash \neg Q \land P}{\Gamma \vdash \langle\{\mathsf{if}(Q)\,\alpha\}^*\rangle(\neg Q \land P), \Delta}\,{\scriptstyle \text{con}}}{\Gamma \vdash \langle\{\mathsf{if}(Q)\,\alpha\}^*; ?\neg Q\rangle P, \Delta}\,{\scriptstyle \langle;\rangle,\langle?\rangle}}{\Gamma \vdash \langle\mathsf{while}(Q)\,\alpha\rangle P, \Delta}$$

We consider the three remaining premises separately. The first premise proves as follows, where the bottom step uses validity $\exists n\,(J \land p(n)) \leftrightarrow J \land \exists n\, p(n)$ since $n \not\in J$, and where the right premise uses the witness $p+1$ for n:

$$
\begin{prooftree}
\AxiomC{$\Gamma \vdash J, \Delta$}
\AxiomC{$*$}
\LeftLabel{$\mathbb{Z}$}
\UnaryInfC{$\Gamma \vdash \exists n{\geq}0\,(Q \to p < n), \Delta$}
\LeftLabel{$\land$R}
\BinaryInfC{$\Gamma \vdash J \land \exists n{\geq}0\,(Q \to p < n), \Delta$}
\UnaryInfC{$\Gamma \vdash \exists n{\geq}0\,(J \land (Q \to p < n)), \Delta$}
\UnaryInfC{$\Gamma \vdash \exists n{\geq}0\, p(n), \Delta$}
\end{prooftree}
$$

The second premise proves as follows (the second branch of $\to$L for $Q \to p < n$ is elided, marked by $\triangleright$, and proves since Q is among the assumptions, the top step in the first branch is a monotonicity step choosing the implicitly universally quantified variable n to be $p+1$):

$$
\begin{prooftree}
\AxiomC{$J, Q, p = n \vdash \langle\alpha\rangle(J \land p < n)$}
\LeftLabel{M}
\UnaryInfC{$J, Q, p < n, n > 0 \vdash \langle\alpha\rangle(J \land p < n-1)$}
\LeftLabel{M}
\UnaryInfC{$J, Q, p < n, n > 0 \vdash \langle\alpha\rangle p(n-1)$}
\AxiomC{$\triangleright$}
\LeftLabel{$\to$L}
\BinaryInfC{$J, Q \to p < n, n > 0, Q \vdash \langle\alpha\rangle p(n-1)$}
\AxiomC{$*$}
\LeftLabel{id}
\UnaryInfC{$J \vdash J$}
\AxiomC{$*$}
\LeftLabel{$\neg$L,id}
\UnaryInfC{$\neg Q, Q \vdash p < n-1$}
\LeftLabel{$\to$R}
\UnaryInfC{$\neg Q \vdash Q \to p < n-1$}
\LeftLabel{$\land$R,WL}
\BinaryInfC{$J, Q \to p < n, n > 0, \neg Q \vdash p(n-1)$}
\LeftLabel{$\land$R,$\to$R}
\BinaryInfC{$J, Q \to p < n, n > 0 \vdash (Q \to \langle\alpha\rangle p(n-1)) \land (\neg Q \to p(n-1))$}
\LeftLabel{$\langle$if$\rangle$,$\land$L}
\UnaryInfC{$p(n), n > 0 \vdash \langle \text{if}(Q)\, \alpha\rangle p(n-1)$}
\end{prooftree}
$$

The third premise proves as follows (again the second branch of $\to$L for $p \to p < 0$ is elided, marked by $\triangleright$ and proves since Q is among the assumptions):

$$
\begin{prooftree}
\AxiomC{$J, Q \vdash \varphi \geq 0$}
\AxiomC{$*$}
\LeftLabel{$\mathbb{Z}$}
\UnaryInfC{$\varphi < 0, \varphi \geq 0 \vdash$}
\LeftLabel{cut,WL}
\BinaryInfC{$J, \varphi < 0, Q \vdash$}
\AxiomC{$\triangleright$}
\LeftLabel{$\to$L,WR}
\BinaryInfC{$J, Q \to \varphi < 0, Q \vdash$}
\LeftLabel{$\neg$R}
\UnaryInfC{$J, Q \to \varphi < 0 \vdash \neg Q$}
\LeftLabel{$\land$L}
\UnaryInfC{$p(0) \vdash \neg Q$}
\AxiomC{$*$}
\LeftLabel{id}
\UnaryInfC{$\neg Q \vdash \neg Q$}
\AxiomC{$J, \neg Q \vdash P$}
\LeftLabel{$\land$L,WL}
\UnaryInfC{$p(0), \neg Q \vdash P$}
\LeftLabel{$\land$R,WL}
\BinaryInfC{$p(0), \neg Q \vdash \neg Q \land P$}
\LeftLabel{cut}
\BinaryInfC{$p(0) \vdash \neg Q \land P$}
\end{prooftree}
$$

$\square$

As a simple example, the following formula easily proves with rule var using $J \equiv$ *true* and $x - 5$ for p:

$$
\begin{prooftree}
\AxiomC{$*$}
\LeftLabel{$\top$R}
\UnaryInfC{$\vdash \mathit{true}$}
\AxiomC{$*$}
\LeftLabel{$\mathbb{Z}$}
\UnaryInfC{$x \geq 5, x - 5 = n \vdash x - 2 - 5 < n$}
\LeftLabel{$\langle:=\rangle$}
\UnaryInfC{$x \geq 5, x - 5 = n \vdash \langle x := x - 2\rangle x - 5 < n$}
\AxiomC{$*$}
\LeftLabel{$\mathbb{Z}$}
\UnaryInfC{$x \geq 5 \vdash x - 5 \geq 0$}
\AxiomC{$*$}
\LeftLabel{$\mathbb{Z}$}
\UnaryInfC{$\neg x \geq 5 \vdash x < 5$}
\LeftLabel{var}
\QuaternaryInfC{$\vdash \langle \text{while}(x \geq 5)\, x := x - 2\rangle\, x < 5$}
\end{prooftree}
$$

The respective proof rules for proving the trivial succedent *true* or proving a sequent with the impossible assumption *false* are as is to be expected:

$$\top\mathrm{R}\ \frac{}{\Gamma \vdash \mathit{true}, \Delta}$$

$$\bot\mathrm{L}\ \frac{}{\mathit{false}, \Gamma \vdash \Delta}$$

9 Example

Recall the proof of the square-by-add program from a previous lecture

$$x \geq 0 \rightarrow [s := 0; i := 0; \mathsf{while}(i < x)\,\{s := s + 2 * i + 1; i := i + 1\}]\, s = x * x \tag{1}$$

When α denotes the loop body $s := s + 2 * i + 1; i := i + 1$ recall that we proved it using the following loop invariant

$$J \overset{\text{def}}{\equiv} i \leq x \land s = i * i \tag{2}$$

$$\cfrac{\cfrac{x{\geq}0, s{=}0, i{=}0 \vdash J \qquad J, i{<}x \vdash [\alpha]J \quad J, \neg i{<}x \vdash s{=}x{*}x}{x \geq 0, s = 0, i = 0 \vdash [\mathsf{while}(i < x)\,\alpha]s = x * x}\,??}{x \geq 0 \vdash [s := 0; i := 0; \mathsf{while}(i < x)\,\alpha]s = x * x}\,[;],??$$

This establishes partial correctness of the square-by-add program but does not guarantee that it also always terminates. Changing the modality from a box to a diamond will require us to prove it using the variance rule var instead of invariant rule **??**. The variance rule var requires both an invariant J, for which we still use (2), and a variant term p, for which we use $p \overset{\text{def}}{=} x - i$ since that decreases to 0 when running the square-by-add program. This leads to the following proof start:

$$\cfrac{\cfrac{x{\geq}0, s{=}0, i{=}0 \vdash J \qquad J, i < x, p = n \vdash \langle\alpha\rangle(J \land p < n) \quad J, i{<}x \vdash p \geq 0 \quad J, \neg i{<}x \vdash s{=}x{*}x}{x{\geq}0, s{=}0, i{=}0 \vdash \langle\mathsf{while}(i < x)\,\alpha\rangle s = x * x}\,\mathrm{var}}{\vdash x{\geq}0 \rightarrow \langle s := 0; i := 0; \mathsf{while}(i < x)\,\alpha\rangle s = x * x}\,\langle;\rangle,\langle:=\rangle_=$$

This first and fourth premise are the same as for the proof of (1) and, thus, still prove in the same way. The third premise proves by arithmetic since $i < x$ implies $x - i \geq 0$:

$$\cfrac{\cfrac{*}{i \leq x \land s = i * i, i{<}x \vdash x - i \geq 0}\,\mathbb{Z}}{J, i{<}x \vdash p \geq 0}$$

The only remaining premise, the induction step in the second premise could be proved again, or proved in a clever way reusing the fact that we already established partial correctness for the program. In particular, we already have a proof of $J, i{<}x \vdash [\alpha]J$, which derived axiom inv$\land$var can exploit.

$$
{\scriptstyle\text{inv}\land\text{var}}\dfrac{{\scriptstyle\land\text{R}}\dfrac{{\scriptstyle\text{WL}}\dfrac{\dfrac{\text{above}}{J, i < x \vdash [\alpha]J}{\scriptstyle\langle;\rangle,\langle:=\rangle}}{J, i < x, p = n \vdash [\alpha]J} \qquad \dfrac{\dfrac{{\scriptstyle\mathbb{Z}}\dfrac{*}{J, i < x, x - i = n \vdash x - (i+1) < n}}{J, i < x, x - i = n \vdash \langle s := s + 2 * i + 1; i := i + 1\rangle x - i < n}}{J, i < x, p = n \vdash \langle\alpha\rangle p < n}}{J, i < x, p = n \vdash [\alpha]J \land \langle\alpha\rangle p < n}}{J, i < x, p = n \vdash \langle\alpha\rangle(J \land p < n)}
$$

10 Summary

A split into a partial correctness argument together with a pure termination analysis is especially helpful when most parts of the behavior of the program are irrelevant for its termination. The other reason to reason like this is that partial correctness and total correctness are often considered separately, at which point it is helpful to reuse the results of the partial correctness analysis for the total correctness analysis. Having said that, a direct analysis with just one single proof that directly targets total correctness is more concise. Today's axioms and proof rules are summarized in Fig. 1.

$$\langle{:=}\rangle\ \langle x := e\rangle p(x) \leftrightarrow p(e)$$

$$\langle ?\rangle\ \langle ?Q\rangle P \leftrightarrow (Q \land P)$$

$$\langle \text{if}\rangle\ \langle \mathsf{if}(Q)\,\alpha\,\mathsf{else}\,\beta\rangle P \leftrightarrow (Q \to \langle\alpha\rangle P) \land (\neg Q \to \langle\beta\rangle P)$$

$$\langle ;\rangle\ \langle \alpha;\beta\rangle P \leftrightarrow \langle\alpha\rangle\langle\beta\rangle P$$

$$\langle \text{unwind}\rangle\ \langle \mathsf{while}(Q)\,\alpha\rangle P \leftrightarrow \langle \mathsf{if}(Q)\,\{\alpha;\mathsf{while}(Q)\,\alpha\}\rangle P$$

$$\text{C}\ \begin{array}{l}[\alpha^*]\forall n{>}0\,(p(n) \to \langle\alpha\rangle p(n-1))\\ \to \forall n{\ge}0\,(p(n) \to \langle\alpha^*\rangle p(0))\end{array} \qquad (n \notin \alpha)$$

$$\text{con}\ \frac{\Gamma \vdash \exists n{\ge}0\, p(n), \Delta \quad p(n), n{>}0 \vdash \langle\alpha\rangle p(n-1) \quad p(0) \vdash P}{\Gamma \vdash \langle\alpha^*\rangle P, \Delta} \qquad \text{(n fresh)}$$

$$\text{var}\ \frac{\Gamma \vdash J, \Delta \quad J, Q, p = n \vdash \langle\alpha\rangle(J \land p < n) \quad J, Q \vdash p \ge 0 \quad J, \neg Q \vdash P}{\Gamma \vdash \langle \mathsf{while}(Q)\,\alpha\rangle P, \Delta} \text{(n fresh)}$$

$$\text{K}\ [\alpha](P \to Q) \to ([\alpha]P \to [\alpha]Q)$$

$$\text{K}_{\langle\cdot\rangle}\ [\alpha](P \to Q) \to (\langle\alpha\rangle P \to \langle\alpha\rangle Q)$$

$$\text{inv2var}\ [\alpha]J \to (\langle\alpha\rangle(J \to p) \to \langle\alpha\rangle p)$$

$$\text{inv}\land\text{var}\ [\alpha]P \land \langle\alpha\rangle Q \to \langle\alpha\rangle(P \land Q)$$

$$\text{M}\ \frac{P \vdash Q}{\Gamma, \langle\alpha\rangle P \vdash \langle\alpha\rangle Q, \Delta}$$

Figure 1: Diamond axioms and proof rules