

Lecture Notes on Convergence

Matt Fredrikson

Carnegie Mellon University

Lecture 11

Tuesday, February 28, 2025

1 Introduction

We began the lecture with a consideration of induction over explicitly defined inductive types, notably unary natural numbers defined by zero and a successor function. We tacked this material on the end of the notes for [Lecture 9](#).

So far in our study of dynamic logic we have focused on $[\alpha]P$, meaning that P is true after every possible run of α . In the world of deterministic programs we call this *partial correctness*: the final state satisfies P , but only if α terminates. We also sometimes talk about a *safety property*: no matter what happens, if we terminate at least P will be true.

The other modality is $\langle\alpha\rangle P$ which means that there is a run of α such that P is true. For deterministic programs (that is, programs that have at most one final state), we call this *total correctness*: α will reach a final state, and it satisfies P . We also sometimes talk about a *liveness property*: something good (that is a final state that satisfied P) will eventually happen.

In this lecture we recall the semantics of $\langle\alpha\rangle P$ more formally and then examine how to break down programs for this particular modal operator by using axioms. This will be straightforward until we encounter α^* , which requires an axiom of *convergence* as a counterpart to the axiom of *induction*.

Learning goals. After this lecture, you should be able to:

- Express liveness properties in dynamic logic
- Reason with the axiom of convergence
- Reason with interacting $[-]$ and $\langle-\rangle$ modalities.

2 Box vs. Diamond

Recall that we defined

$$\begin{aligned}\omega \models [\alpha]Q & \text{ iff for every } \nu, \omega \llbracket \alpha \rrbracket \nu \text{ implies } \nu \models P \\ \omega \models \langle \alpha \rangle Q & \text{ iff there exists a } \nu \text{ such that } \omega \llbracket \alpha \rrbracket \nu \text{ and } \nu \models P\end{aligned}$$

Both of these are with respect to the same semantics $\omega \llbracket \alpha \rrbracket \nu$. In the first case, if ν is reachable then P must be true; in the second case some such ν must be reachable.

Recall the definitions

$$\begin{aligned}\text{skip} & \triangleq ?\text{true} \\ \text{abort} & \triangleq ?\text{false}\end{aligned}$$

From this definition we can deduce the following properties. You should make sure you understand each line.

$$\begin{aligned}[\text{skip}]P & \text{ iff } P \\ \langle \text{skip} \rangle P & \text{ iff } P \\ [\text{abort}]P & \text{ always} \\ \langle \text{abort} \rangle P & \text{ never} \\ [\alpha^*]\text{true} & \text{ always} \\ \langle \alpha^* \rangle \text{true} & \text{ always} \\ [\alpha^*]\text{false} & \text{ never} \\ \langle \alpha^* \rangle \text{false} & \text{ never}\end{aligned}$$

3 One Axiom for Diamonds

It turns out that in dynamic logic we can give a single axiom characterizing $\langle \alpha \rangle Q$:

$$\langle \alpha \rangle Q \leftrightarrow \neg[\alpha]\neg Q$$

Let's reason this through, starting on the right-hand side:

$$\begin{aligned}\omega \models \neg[\alpha]\neg Q & \text{ iff it is not the case that for all } \nu, \omega \llbracket \alpha \rrbracket \nu \text{ implies that } \nu \models \neg Q \\ & \text{ iff there exists a } \nu \text{ such that } \omega \llbracket \alpha \rrbracket \nu \text{ and it is not the case that } \nu \models \neg Q \\ & \text{ iff there exists a } \nu \text{ such that } \omega \llbracket \alpha \rrbracket \nu \text{ and } \nu \models Q \\ & \text{ iff } \omega \models \langle \alpha \rangle Q\end{aligned}$$

Here we have used some fundamental laws of (classical) reasoning in our language of mathematical discourse, such as $\neg(P \rightarrow Q) \leftrightarrow (P \wedge \neg Q)$ and $\neg(\exists x. Q) \leftrightarrow \forall x. \neg Q$.

This observation will carry us quite far, but it will not help us when we come to induction.

4 Other Axioms for Diamonds

We would like to break down the programs in $\langle \alpha \rangle Q$ in order to generate a verification condition in pure arithmetic. In some cases this works just as for $[\alpha]Q$, in other cases it is very different.

We start with assignment. This will always terminate in one step, so a property of all runs is the same as a property of one run.

$$\langle x \leftarrow e \rangle Q(x) \leftrightarrow \forall x'. x' = e \rightarrow Q(x') \quad (x' \text{ not in } e, Q(x))$$

Sequential composition also does not change matters in any essential way. Just for a change of style, let's use the axiom from the previous section to derivation what should hold.

$$\begin{aligned} \langle \alpha ; \beta \rangle Q & \text{ iff } \neg[\alpha ; \beta] \neg Q \\ & \text{ iff } \neg[\alpha][\beta] \neg Q \\ & \text{ iff } \langle \alpha \rangle \neg[\beta] \neg Q \\ & \text{ iff } \langle \alpha \rangle \langle \beta \rangle \neg \neg Q \\ & \text{ iff } \langle \alpha \rangle \langle \beta \rangle Q \end{aligned}$$

Informally, we can argue as follows: there is a run of $\alpha ; \beta$ if there is a run of α to some intermediate state, and a run of β from there after which Q is true. And that's the same as running α to a state from which β can reach a state in which Q is true.

For nondeterministic choice $\alpha \cup \beta$, we can reach a final state either by choosing α or choosing β .

$$\begin{aligned} \langle \alpha \cup \beta \rangle Q & \text{ iff } \neg[\alpha \cup \beta] \neg Q \\ & \text{ iff } \neg([\alpha] \neg Q \wedge [\beta] \neg Q) \\ & \text{ iff } (\neg[\alpha] \neg Q) \vee (\neg[\beta] \neg Q) \\ & \text{ iff } \langle \alpha \rangle Q \vee \langle \beta \rangle Q \end{aligned}$$

We see that through the negations this dualizes the axiom for $[\alpha \cup \beta]$, used in the second step above.

Finally, for guards they are opposites in a different say.

$$\begin{aligned} \langle ?P \rangle Q & \text{ iff } \neg[?P] \neg Q \\ & \text{ iff } \neg(P \rightarrow \neg Q) \\ & \text{ iff } P \wedge Q \end{aligned}$$

Finally, we come to repetition. There is a simple analogue of the axiom to unroll a loop, for the same reason as nondeterministic choice. We won't go through steps, just show the final equivalence.

$$\langle \alpha^* \rangle Q \leftrightarrow Q \vee \langle \alpha \rangle \langle \alpha^* \rangle Q$$

As before, this finite unrolling is of limited utility.

5 Convergence

In practice, unrolling a loop a finite number of times is insufficient to prove most programs. Instead, we work with the induction axiom and then invariants when proving $[\alpha^*]Q$. Recall:

$$\begin{aligned} [\alpha^*]Q &\leftrightarrow Q \wedge [\alpha^*](Q \rightarrow [\alpha]Q) && \text{(induction)} \\ &\leftarrow J \wedge \Box(J \rightarrow [\alpha]J) \wedge \Box(J \rightarrow Q) \end{aligned}$$

What is the analogue for induction for $\langle \alpha^* \rangle Q$? We can work through it and see what the mechanical approach yields.

$$\begin{aligned} \langle \alpha^* \rangle Q &\text{ iff } \neg[\alpha^*]\neg Q \\ &\text{ iff } \neg(\neg Q \wedge [\alpha^*](\neg Q \rightarrow [\alpha]\neg Q)) \\ &\text{ iff } Q \vee \neg[\alpha^*](\neg Q \rightarrow [\alpha]\neg Q) \\ &\text{ iff } Q \vee \langle \alpha^* \rangle(\neg(\neg Q \rightarrow [\alpha]\neg Q)) \\ &\text{ iff } Q \vee \langle \alpha^* \rangle(\neg Q \wedge \langle \alpha \rangle Q) \end{aligned}$$

Unfortunately, the resulting axiom (while true) is not very useful.

$$\langle \alpha^* \rangle Q \leftrightarrow Q \vee \langle \alpha^* \rangle(\neg Q \wedge \langle \alpha \rangle Q)$$

It states that there is a way to reach a poststate where Q is true either if it already happens to be true in the current state (and we go around the loop zero times), or there is a way to go around the loop some number of times in such a way that, after that, Q is false but we can restore it with one more iteration.

Instead, we have to somehow capture, in a slightly more abstract way, the reasoning behind the `variant` contracts in Why3 that guarantee termination.

To capture this logically we assume that a predicate V is parameterized by an integer variable n , written as $V(n)$. We prohibit the variable n from appearing in programs; instead we use V to *relate* n to expressions occurring in the program. The axiom of convergence then says

It is possible to reach a poststate with $V(0)$ after a finite number of iterations of α if (1) initially $V(n)$ for some $n \geq 0$,
and (2) at each iteration, assuming $V(n)$ for $n > 0$
implies we can reach a poststate with $V(n - 1)$.

Translating this an axiom gives us

$$\begin{aligned} \langle \alpha^* \rangle V(0) &\leftarrow (\exists n. n \geq 0 \wedge V(n)) \\ &\quad \wedge [\alpha^*](\forall n. n > 0 \wedge V(n) \rightarrow \langle \alpha \rangle V(n - 1)) \\ &\quad (n \text{ not in } \alpha) \end{aligned}$$

It is interesting that this axiom incorporates $[\alpha^*]P$ because we need to make sure that no matter how many iterations we need until we reach 0 the decrease of n will always take place.

To make this effective we take one more step: we think of $V(n)$ as the *predicate variant* of the iteration and use it to prove an arbitrary postcondition Q . As before, this replaces $[\alpha^*]P$ by $\Box P$, and makes sure the variant predicate implies the postcondition. This is slightly different than the *variant expression* we use in Why3, which we address in the next section.

$$\begin{aligned} \langle \alpha^* \rangle Q \leftarrow & (\exists n. n \geq 0 \wedge V(n)) \\ & \wedge \Box(\forall n. n > 0 \wedge V(n) \rightarrow \langle \alpha \rangle V(n-1)) \\ & \wedge \Box(V(0) \rightarrow Q) \\ & (n \text{ not in } \alpha \text{ or } Q) \end{aligned}$$

As an example, let's prove

$$x \geq 0 \rightarrow \langle (x \leftarrow x - 1)^* \rangle x = 0$$

In order to apply convergence we have to define the variant formula $V(n)$. In this case, it is easy and we choose

$$V(n) = (x = n)$$

that is, n just tracks the value of x . We proceed:

To prove (init): $x \geq 0 \rightarrow \exists n. n \geq 0 \wedge x = n$ True (pick $n = x$)

To prove (step): $x \geq 0 \rightarrow \Box(\forall n. n > 0 \wedge x = n \rightarrow \langle x \leftarrow x - 1 \rangle x = n - 1)$

True if $\forall n. n > 0 \wedge x = n \rightarrow \forall x'. x' = x - 1 \rightarrow x' = n - 1$

True if $\forall n. n > 0 \wedge x = n \rightarrow x - 1 = n - 1$ By arithmetic

To prove (post): $x \geq 0 \rightarrow \Box(x = 0 \rightarrow x = 0)$

True if $x = 0 \rightarrow x = 0$

To illustrate how we have to think about picking $V(n)$, consider the slightly more complicated example

$$x \geq 0 \rightarrow \langle (x \leftarrow x - 2)^* \rangle (x = 0 \vee x = 1)$$

Consider what variant formula $V(n)$ might allow us to do this proof.

We pick $V(n) = (x = 2n \vee x = 2n + 1)$. Then $V(0) = (x = 0 \vee x = 1)$ and $V(n - 1) = (x = 2n - 2 \vee x = 2n - 1)$. We reason:

To prove (init): $x \geq 0 \rightarrow \exists n. n \geq 0 \wedge (x = 2n \vee x = 2n + 1)$

True (every number is either even or odd)

To prove (step): $x \geq 0 \rightarrow \Box(\forall n. n > 0 \wedge (x = 2n \vee x = 2n + 1)$

$\rightarrow \langle x \leftarrow x - 2 \rangle (x = 2n - 2 \vee x = 2n - 1))$

True if $\forall n. n > 0 \wedge (x = 2n \vee x = 2n + 1 \rightarrow x - 2 = 2n - 2 \vee x - 2 = 2n - 1)$

By arithmetic

To prove (post): $x \geq 0 \rightarrow \Box(x = 0 \vee x = 1 \rightarrow x = 0 \vee x = 1)$

True if $x = 0 \vee x = 1 \rightarrow x = 0 \vee x = 1$

Valid

6 Interactions Between Box and Diamond

Already, the axiom of convergence mixes $[\alpha]P$ and $\langle \alpha \rangle P$. This interaction is a bit tricky, so we consider a few simpler cases on how these modalities interact.

$$[\alpha](P \rightarrow Q) \rightarrow ([\alpha]P \rightarrow [\alpha]Q) \quad \text{Valid}$$

If P implies Q in every poststate of α , then if P is also true in every poststate, so must Q be.

$$\langle \alpha \rangle(P \rightarrow Q) \rightarrow (\langle \alpha \rangle P \rightarrow \langle \alpha \rangle Q) \quad \text{Not valid}$$

There is a poststate in which P implies Q and also a poststate in which P is true. Since these two poststate may be different, we cannot be certain that there will be a poststate in which Q is true.

$$[\alpha](P \rightarrow Q) \rightarrow (\langle \alpha \rangle P \rightarrow \langle \alpha \rangle Q) \quad \text{Valid}$$

If P implies Q in every poststate of α , then this will also be true in the poststate in which P is true. Therefore, Q will be true in that poststate.

In the next two we explore the consequence of an invariant J

$$[\alpha]J \rightarrow (\langle \alpha \rangle(J \rightarrow Q) \rightarrow \langle \alpha \rangle Q) \quad \text{Valid}$$

If J is true in every poststate of α , and there is a poststate where J implies Q , then Q must be true in that poststate.

$$[\alpha]J \rightarrow (\langle \alpha \rangle Q \rightarrow \langle \alpha \rangle(J \wedge Q)) \quad \text{Valid}$$

If J is true in every poststate of J , and there is a poststate where Q is true, then both J and Q must be true in that poststate.

7 From Variant Formulas to Variant Expressions

We generalize the axiom of convergence with *variant predicates* to one with *variant expressions* allowing “big steps” where the expressions may decrease by more than 1. In this formulation we explicitly highlight an invariant J together with the variant expression e . Both of these may mention program variables but not the new variable n which tracks the value of the variant in the axiom. This closely approximates what the verification condition generator for Why3 does for while-loops.

One of the key ideas here is that the invariant may help us to establish the variant. In lecture we stated:

$$\begin{aligned} \langle \alpha^* \rangle Q \leftarrow & J \\ & \wedge \Box(J \rightarrow e \geq 0) \\ & \wedge \Box(\forall n. J \wedge e = n \rightarrow \langle \alpha \rangle (J \wedge e < n)) \\ & \wedge \Box(J \rightarrow Q) \\ & (n \text{ not in } J, e, \text{ or } Q) \end{aligned}$$

While this is true, this is not very useful:¹ if we know J and $\Box(J \rightarrow Q)$ then we can conclude $\langle \alpha^* \rangle Q$ immediately with zero iterations.

So we skip to the version for while loops, recalling that

$$\text{while } P \alpha \triangleq (?P ; \alpha)^* ; ?\neg P$$

We can then justify the following axiom (which don’t formally prove sound):

$$\begin{aligned} \langle \text{while } P \alpha \rangle Q \leftarrow & J \\ & \wedge \Box(J \wedge P \rightarrow e \geq 0) \\ & \wedge \Box(\forall n. J \wedge P \wedge e = n \rightarrow \langle \alpha \rangle (J \wedge e < n)) \\ & \wedge \Box(J \wedge \neg P \rightarrow Q) \\ & (n \text{ not in } \alpha, J, P, e, \text{ or } Q) \end{aligned}$$

As an example you may consider the following correctness statement for computing Fibonacci numbers, using simultaneous assignment as a shorthand.

$$x \geq 0 \rightarrow \langle a \leftarrow 0 ; b \leftarrow 1 ; i \leftarrow 0 ; \text{while } (i < x) (a, b \leftarrow b, a + b ; i \leftarrow i + 1) \rangle a = \text{fib } x$$

To conduct this proof we pick

$$\begin{array}{ll} e &= (x - i) && \text{variant expression} \\ J &= (0 \leq i \leq x \wedge a = \text{fib}(i) \wedge b = \text{fib}(i + 1)) && \text{invariant} \end{array}$$

It is then a mechanical exercise to verify the conditions of the axioms for while with invariants and variant expressions.

¹as was pointed out by a student after lecture