



# Concurrent Models of Computation for Embedded Software

Edward A. Lee

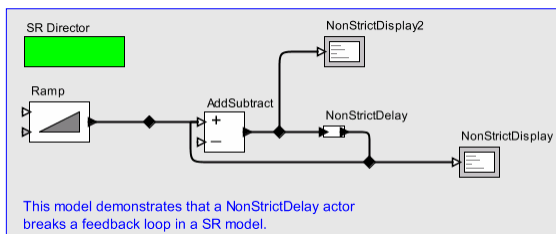
Professor, UC Berkeley  
EECS 290n – Advanced Topics in Systems Theory  
Fall, 2004

Copyright © 2004, Edward A. Lee, All rights reserved

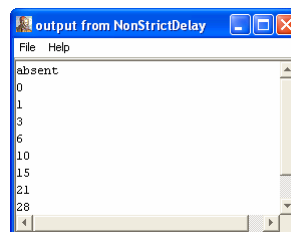
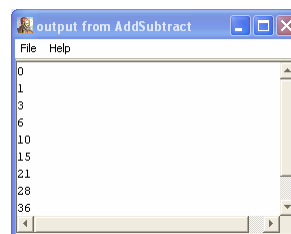
Lecture 10: Synchronous Reactive Semantics

## SR Domain in Ptolemy II

At each tick of a global “clock,” every signal has a value or is absent.



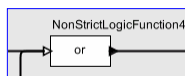
The job of the SR director is to find the value at each tick.



## Cycles

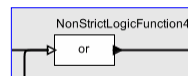
Note that there are cycles in this graph, so that if you require that all inputs be known to find the output, then this cannot execute.

The “non strict” actors are key: They do not need to know all their inputs to determine the outputs.



Lee 10: 3

## Non-Strict Logical Or



The non-strict or (often called the “parallel or”) can produce a known output even if the input is not completely known. Here is a table showing the output as a function of two inputs:

		input 1			
		$\perp$	$\epsilon$	F	T
input 2	$\perp$	$\perp$	$\perp$	$\perp$	T
	$\epsilon$	$\perp$	$\epsilon$	F	T
	F	$\perp$	F	F	T
	T	T	T	T	T

Lee 10: 4

## Simple Execution Policy

At each tick, start with all signals “unknown.” Evaluate non-strict actors and source actors. Then keep evaluating any actors that can be evaluated until all signals become known or until no further progress can be made.

Q: How do we know this will work?

A: Least fixed point semantics.

Lee 10: 5

## The Flat CPO

Consider a set of possible values  $T = \{t_1, t_2, \dots\}$ . Let

$$A = T \cup \{\perp, \varepsilon\}$$

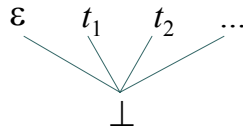
where  $\perp$  represents “unknown” and  $\varepsilon$  represents “absent.”

Let  $(A, \leq)$  be a partial order where:

- $\perp \leq \varepsilon$
- for all  $t$  in  $T$ ,  $\perp \leq t$
- all other pairs are incomparable

Lee 10: 6

## Hasse Diagram for the Flat CPO



Note that this is obviously a CPO  
(all chains have a LUB)

All chains have length 2.

Lee 10: 7

## Monotonic Functions on This CPO

In this CPO, any function  $f: A \rightarrow A$  is monotonic if

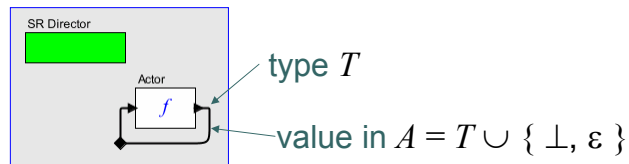
$$f(\perp) = a \neq \perp \Rightarrow f(b) = a \text{ for all } b \in A$$

I.e., if the function yields a “known” output when the input is unknown, then it will not change its mind about the output once the input becomes known.

Since all chains are finite, every monotonic function is continuous.

Lee 10: 8

## Applying Fixed Point Theorem 1



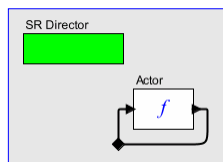
At each tick of the clock

- Start with signal value  $\perp$
- Evaluate  $f(\perp)$
- Evaluate  $f(f(\perp))$
- Stop when a fixed point is reached

Unlike PN, a fixed point is always reached in a finite number of steps (one, in this case).

Lee 10: 9

## Causality Loops

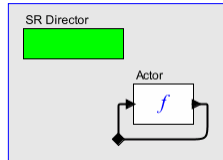


What is the behavior in the following cases?

- $f$  is the identity function.
- $f$  is the logical NOT function.
- $f$  is the nonstrict delay function with initial value 0.
- $f$  is the nonstrict delay function with no initial value.

Lee 10: 10

## Causality Loops

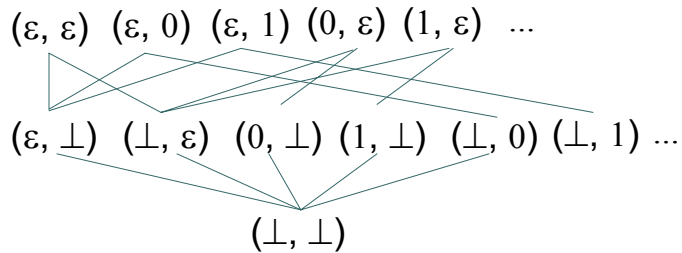
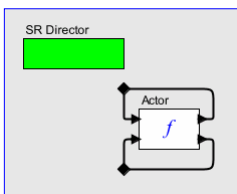


What is the behavior in the following cases?

- $f$  is the identity function:  $\perp$
- $f$  is the logical NOT function:  $\perp$
- $f$  is the nonstrict delay function with initial value 0: 0
- $f$  is the nonstrict delay function with no initial value:  $\varepsilon$

Lee 10: 11

## Generalizing to Multiple Signals

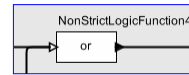


product CPO assuming  $T = \{0, 1\}$ .

- The Cartesian product of flat CPOs under pointwise ordering is also a CPO.
- All chains are still finite.
- Can now apply to any composition, as done with PN.

Lee 10: 12

## Non-Strict Logical Or is Monotonic



The non-strict or is a monotonic function  $f: A \times A \rightarrow A$  where  $A = \{\perp, \varepsilon, T, F\}$  as can be verified from the truth table:

		input 1			
		$\perp$	$\varepsilon$	F	T
input 2	$\perp$	$\perp$	$\perp$	$\perp$	T
	$\varepsilon$	$\perp$	$\varepsilon$	F	T
	F	$\perp$	F	F	T
	T	T	T	T	T

Lee 10: 13

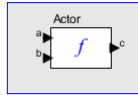
## Compositional Reasoning

So far, with both PN and SR, we deal with composite systems by reducing them to a monotonic function of all the signals. An alternative approach is to convert an arbitrary composition to a continuous function.

Lee 10: 14

## Example to Use for Compositional Reasoning

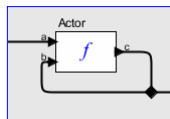
Consider an actor:



Assume  $a, b, c \in A$ , where  $A$  is a CPO.

Assume that the actor function  $f: A \times A \rightarrow A$  is continuous

Consider the following composition:



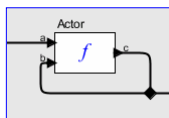
We would like to consider this a function from  $a$  to  $c$ .

Lee 10: 15

## First Option: Currying (Named after Haskell Curry)

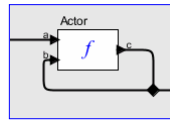
Given a function  $f: A \times A \rightarrow A$ , we can alternatively think of this in stages as  $f_1: A \rightarrow [A \rightarrow A]$ , where  $[A \rightarrow A]$  is the set of all functions from  $A$  to  $A$ .

For the following example, for each given value of  $a$  we get a new function  $f_1(a)$  for which we can find the least fixed point. That least fixed point is the value of  $c$ .



Lee 10: 16

## Example: Non-Strict OR



Suppose  $f$  is a non-strict logical OR function. Then:

- If  $a = true$ , then the resulting function  $f_1(a)$  always returns  $true$ , for all values of the input  $b$ .

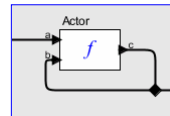
In this case, the least fixed point yields  $c = true$ .

- If  $a = false$ , then the resulting function  $f_1(a)$  always returns  $b$ , for all values of the input  $b$ .

In this case, the least fixed point yields  $c = \perp$ .

Lee 10: 17

## Second Option: Lifting (Named after Heavy Lifting)



Given a function  $f: A \times A \rightarrow A$ , we are looking for a function  $g: A \rightarrow A$  such that

$$c = g(a)$$

In the model we have  $b = c$  and  $c = f(a, b)$  so

$$g(a) = f(a, g(a))$$

This looks like a fixed point problem, but the “unknown” on both sides is  $g$ , a function not a value. If we can find the function  $g$  that satisfies this equation, then we can use it always to calculate  $c$  given  $a$ .

Lee 10: 18

## Posets of Functions

Suppose  $(A, \leq)$  and  $(B, \leq)$  are CPOs.

Consider functions  $f, g \in [A \rightarrow B]$ .

Define the *pointwise order* on these functions to be

$$f \leq g \Leftrightarrow \forall a \in A, f(a) \leq g(a)$$

Let  $X \subset [A \rightarrow B]$  be the set of all continuous total functions from  $A$  to  $B$ .

**Theorem:**  $(X, \leq)$  is a CPO under the pointwise order.

**Proof:** See handout.

Lee 10: 19

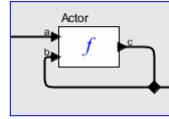
## Least Function in the CPO of Functions

Let  $X \subset [A \rightarrow B]$  be the set of all continuous total functions from  $A$  to  $B$ . Since  $X$  is a CPO, it must have a bottom. The bottom is a function  $\perp_X: A \rightarrow B$  where for all  $a \in A$ ,

$$\perp_X(a) = \perp_B \in B$$

Lee 10: 20

## Consequence of this Theorem



Given a continuous function  $f: A \times A \rightarrow A$ , the function  $g: A \rightarrow A$  such that

$$c = g(a)$$

is the least fixed point of a continuous function

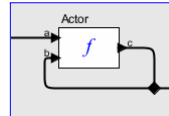
$$F: X \rightarrow X$$

where  $X \subset [A \rightarrow A]$  is the set of all continuous total functions from  $A$  to  $A$ .

We need to now determine the continuous function  $F$ .

Lee 10: 21

## Consequence of this Theorem (Continued)



We need to find a function that  $g$  satisfies:

$$g(a) = f(a, g(a))$$

Let  $X \subset [A \rightarrow A]$  be the set of all continuous total functions from  $A$  to  $A$  and let  $F$  be a continuous function  $F: X \rightarrow X$ .

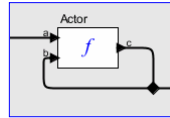
Then  $g \in X$  is the least function such that  $F(g) = g$  where all  $a \in A$ ,

$$(F(g))(a) = f(a, g(a))$$

The theorem, with fixed point theorem 1, tells us that  $F$  has a least fixed point, and tells us how to find it.

Lee 10: 22

## Example: Non-Strict OR



Suppose  $f$  is a non-strict logical OR function. Then:

$$(F(g))(a) = \begin{cases} true & \text{if } a = true \\ g(a) & \text{otherwise} \end{cases}$$

The least fixed point of this is the function  $f$  given by:

$$g(a) = \begin{cases} true & \text{if } a = true \\ \perp & \text{otherwise} \end{cases}$$

To find this, start with  $F(\perp)$ , then find  $F(F(\perp))$ , etc., until you get a fixed point (which happens immediately).

Lee 10: 23

## Showing that $F$ is Continuous

Need to show that given a chain of continuous total functions  $C = \{g_1, g_2, \dots\}$  that:

$$F(\vee C) = \vee \hat{F}(C)$$

For all  $a \in A$ :

$$\begin{aligned} (F(\vee C))(a) &= f(a, (\vee C)(a)) \\ &= f(a, \vee \{g_1(a), g_2(a), \dots\}) && \text{because each } g_i \text{ is continuous} \\ &= \vee \hat{f}(a, \{g_1(a), g_2(a), \dots\}) && \text{because } f \text{ is continuous} \\ &= (\vee \hat{F}(C))(a) && \text{QED} \end{aligned}$$

Lee 10: 24

## Conclusion and Open Issues

- In SR, fixed point semantics is simpler than in PN because the CPO has only finite chains.
- The fancier techniques of Currying and Lifting can be applied equal well to PN, but we introduce them here because the simpler CPO makes them easier to understand.
- The fixed point semantics of SR talks only about the behavior at a tick of the clock. The behavior across ticks of the clock will require a *clock calculus*.