CS 5860 A Theory of Events in First-Order Logic

Tue Nov 15, 2011

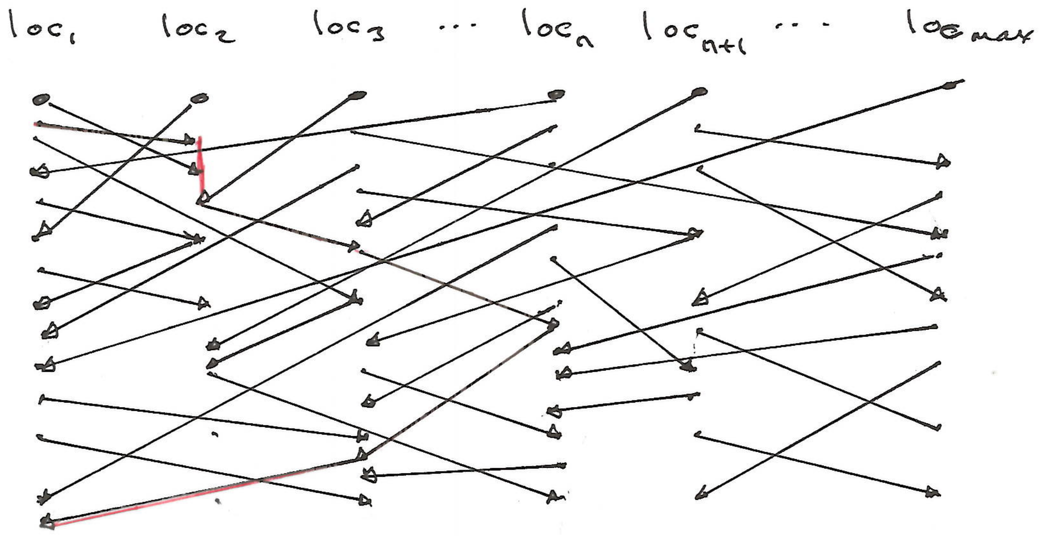
We have seen how to express a few protocols such as mutual exclusion, message acknowledgement, and collecting responses to liveness "pings" in the setting of computation by asynchronous message passing. We also presented a intuitive "theory of events" at a finite number of locations $loc_1, \ldots, loc_n$ which send and receive messages. We stressed the idea that there is no global notion of time (no global clock) and that we reason about time in terms of Lamport's notion of causal order among events.

Now we will see how to express these concepts in first-order logic. Unlike the case for first-order number theory where the domain of discourse D is the type of natural numbers, for a theory of events, we need to subdivide the domain into several sorts. We start with the sort of events and locations defined by decidable predicates $E(x)$, $Loc(x)$ on D. For convenience, we also use $Bool(x)$ for Booleans and $Unit(x)$ for a sort with one object written as $\bullet$; we also have the natural numbers $N(x)$ when we want them. All of these predicates simply divide D into separate sorts, all disjoint and decidable, i.e. $\forall x.(E(x) \vee \sim E(x))$, $\forall x.(Loc(x) \vee \sim Loc(x))$, $\forall x.(N(x) \vee \sim N(x))$, etc. Later we will see that type theory offers a richer and more flexible way to handle sorts and logic in a uniform way.

CS 5860

Tue Nov 15, 2011 A Theory of Events in First-Order Logic

Recall the picture of our model of computation. This picture is sometimes called a message sequence diagram.



At each location loc_i events are linearly ordered, creating a sequential notion of time in which events are totally ordered. It does not make sense to draw an arrow going back in time at a location, and causal order proceeds downward as illustrated by the red links from loc_1 to loc_2 to loc_3 to loc_n back to loc_3 back to loc_1. We show slow processes at a location by the fact that events are widely spaced, not by directing arrows backwards.

We will capture properties of events caused by computational processing executing at each location, so we also think of the locations as processes. These processes could be executing many threads of computation distinguished by the kind of events.

CS 5860

Tue Nov 15, 2011 A Theory of Events in FOL continued

Equality of events

We assume that equality on Locations and Events is decidable, thus $\forall x,y.(E(x) \& E(y) \Rightarrow x = y \vee \sim(x=y))$ $\forall x,y.(Loc(x) \& Loc(y) \Rightarrow x = y \vee \sim(x=y))$

Notation. It is convenient to write typed quantifiers to express the above concepts as well as many others, e.g. $\forall x,y{:}E.\ (x = y \vee \sim(x=y))$ and $\forall x{:}E.\ \forall i{:}Loc.\ P(x,i)$ means $\forall x,i.(E(x) \& Loc(i) \Rightarrow P(x,i))$.

Axiom $\forall x{:}E.\ \exists y.\ (E(y) \vee Loc(y))$

The realizer for this axiom is the term pred?(x), a computable term that defines a function on events whose value is a pair $\langle y, p\rangle$ where y is in the domain D and p is either inl(*) or inr(*).† footnote.

Given a particular event e at a location loc_i, the term pred?(x) will decide whether e is the initial event at loc_i and if so, it will return $\langle loc_i, inr(*)\rangle$. Otherwise, pred?(e) will compute to the previous event at the location.

By examining the second component of $\langle y, p\rangle$ we can tell whether the result is an event, e.g. p is inl(*), or a location.

If e is not the initial event, then we can examine the sequence of events at the location of e and find its predecessor. To do this, we need to be able to compute the location of the event.

† Footnote: Recall that if E(y) is true, the evidence is *, likewise for Loc(y).

CS5860

Tue Nov. 15, 2011 A Theory of Events in EOL

Axiom $\forall x:E.\ \exists y:Loc.\ Occurs\_at(x,y)$

The realizer is a term loc(x) which for any event x computes the unique location of the event. For simplicity we let loc(x) have value i rather than $\langle i, * \rangle$. The other option would be to have $locof(x) = \langle i, * \rangle$ and $loc(x) = spread(locof(x); i,a.\ i)$. All events happen at a process/location, and we postulate some unspecified mechanism to find the location. In the formal mathematical model we simply define an event to include its location, e.g. in one formal model from 2003 an event is a pair $\langle i, t \rangle$ where i is the location and t is a discrete time step.

Next we need an axiom for finding the sender of an event. If the event is not a receive, then we associate the unit value rather than the sender.

Axiom $\forall x:E.\ \exists y.(E(y) \vee Unit(y))$

The realizer is the term sender?(x). The term computes by finding the canonical form of the event. If it has the form rcv(w) then we find the sender from the header or the channel (as with pred?). If x is not a receive, then sender?(x) computes to the unit value •.

CS 5860

Tue Nov. 15, 2011 A Theory of Events in FOL

Next we will define $x \lhd y$, also written as the binary relation Pred(x,y). First we define these functions and predicates.

first? : E → Bool

first?(x) = spread(pred?(x); y,p. decide(p; l. false; r. true))

= let pred?(x) = (y,p) in if isl(p) then false else true

sender? : E → E + Unit

rcv?(x) = decide(sender?(x); l. true; r. false)

= if isl(sender?(x)) then true else false.

First(x) iff first?(x) = true

Rcv(x) iff rcv?(x) = true

Pred(x,y) iff $(\neg$ First(y) & x = pred(y)$)$

$\vee$ (Rcv(y) & x = sender(y))

where on y such that $\neg$ First(y),

pred(y) = spread(pred?(y); x,p. x)

and on y such that Rcv(y),

sender(y) = if isl(sender?(y)) then outl(y).

We will now form the transitive closure of Pred(x,y), this will be Lamport's causal order relation, $x < y$. We will be able to prove $\forall x,y{:}E.\ (x<y \vee \sim(x<y))$, but we need more axioms.

CS 5860

Tue Nov 15, 2011 A Theory of Events in FOL

Given a relation $R(x,y)$ we define its transitive closure as follows. Define $R^{(0)}(x,y)$ iff $R(x,y)$ and $R^{(n+1)}(x,y)$ iff $R(x,z)$ & $R^{(n)}(z,y)$ for some z.

$$R^{*}(x,y) \text{ iff } \exists n:\mathbb{N}.\ R^{(n)}(x,y).$$

Using the notation $x \lhd y$ for $\mathrm{Pred}(x,y)$, define

$x \lhd^{(0)} y$ iff $x \lhd y$ and $x \lhd^{(n+1)} y$ iff $\exists z.(x \lhd z \;\&\; z \lhd^{(n)} y)$.

Say $x \lhd^{*} y$ iff $\exists n:\mathbb{N}.\ x \lhd^{(n)} y$.

Definition: Lamports causal order on events is Pred^{*} (same as $\lhd^{*}$).

We will show that we can reason by induction on Pred^{*} An elegant way to do this is by postulating that $\mathrm{Pred}(x,y)$ is strongly well founded.

Axiom $\mathrm{Pred}(x,y)$ is strongly well founded, i.e.

$$\exists f: E \to \mathbb{N}.\ \forall e,e': E.\ \mathrm{Pred}(e,e') \Rightarrow f(e) < f(e').$$

We also need an axiom about $\mathrm{pred}(x)$.

Axiom The predecessor function, pred, is injective (i.e. one-to-one)

$$\forall e,e': E.\ (\mathrm{loc}(e) = \mathrm{loc}(e') \;\&\; \neg \mathrm{First}(e) \;\&\; \neg \mathrm{First}(e')) \Rightarrow$$
$$(\mathrm{pred}(e) = \mathrm{pred}(e')) \Rightarrow e = e'.$$

Theorem $\mathrm{Pred}^{*}(x,y)$, causal order, is strongly well founded.