Lecture 23: Futures and speculations

15-312: Principles of Programming Languages
Rob Simmons

Tuesday, April 15, 2013

The dynamics and sequential, structural dynamics of futures and speculations are covered in the book (Chapter 41 of the current online version). Here I will give a different version of the parallel dynamics of futures based on abstract machines.

Futures and speculations can both exist naturally in a language with nested parallelism. We will have a global sequential transition relation $s \underset{\Sigma}{\mapsto} s'$ – as we will see, symbols can now appear within expressions, so the sequential abstract machine dynamics will need to mention the set of symbols that can appear within the abstract machine states s or s'. This is a very uniform extension like it was when we augmented the expression evaluation relation $e \underset{\Sigma}{\mapsto} e'$ in Algol.

The global states for the dynamics of nested parallelism had the form $\nu\Sigma\ \{\mu\}$, where μ contains the evolving state of the parallel machines and has the form $a_1 \hookrightarrow s_1 \otimes \ldots \otimes a_k \hookrightarrow s_k$. These dynamics will now will have the form $\nu\Sigma\ \{\mu \parallel \varphi\}$, where φ is the *memo table* and contains bindings with one of two forms: $a_i \hookrightarrow \bullet$, meaning that the value associated with a_i has not yet been computed, or $a_i \hookrightarrow \lhd v_i$, meaning that the value associated with a_i is now (permanently) v_i.

$$\frac{\forall a \sim \tau \in \Sigma,\ \text{either} \quad \vdash_\Sigma \mu(a) : \tau \quad \text{or} \quad \varphi(a) = \lhd v \ \text{and}\ \emptyset \vdash_\Sigma v : \tau \ \text{and}\ v\ \mathsf{val}_\Sigma}{\nu\Sigma\ \{\mu \parallel \varphi\}\ \mathsf{ok}}$$

Whereas the active computation gets divied up by global transitions, the part of the memo table that contains values (and thus doesn't change) may be shared across all the local transitions in a global transition:

$$\frac{\begin{array}{c}\nu\Sigma_1, \Sigma\ \{\mu_1 \parallel \varphi \otimes \varphi_1\} \underset{loc}{\longmapsto} \nu\Sigma_1', \Sigma\ \{\mu_1' \parallel \varphi \otimes \varphi_1'\} \\ \ldots \qquad \nu\Sigma_n, \Sigma\ \{\mu_n \parallel \varphi \otimes \varphi_n\} \underset{loc}{\longmapsto} \nu\Sigma_n', \Sigma\ \{\mu_n' \parallel \varphi \otimes \varphi_n'\}\end{array}}{\begin{array}{c}\nu\Sigma_0, \Sigma_1, \ldots, \Sigma_n, \Sigma\ \{\mu_0 \otimes \mu_1 \otimes \ldots \otimes \mu_n \parallel \varphi \otimes \varphi_1 \otimes \ldots \otimes \varphi_n\} \qquad \underset{glo}{\longmapsto} \\ \nu\Sigma_0, \Sigma_1', \ldots, \Sigma_n', \Sigma\ \{\mu_0 \otimes \mu_1 \otimes \ldots \otimes \mu_n \parallel \varphi \otimes \varphi_1' \otimes \ldots \otimes \varphi_n'\}\end{array}}$$

The simplest local transition is still one that takes a single step according to the sequential semantics.

$$\frac{s \underset{\Sigma}{\mapsto} s'}{\nu\Sigma, a \sim \tau'\ \{a \hookrightarrow s \parallel \varphi\} \underset{loc}{\longmapsto} \nu\Sigma, a \sim \tau'\ \{a \hookrightarrow s' \parallel \varphi\}}\,(\texttt{locstep-step})$$

1 Futures

Futures immediately evaluate to values by returning a reference to that assignable:

— $\nu\Sigma, a \sim \tau' \; \{a \hookrightarrow (\; k \rhd \mathtt{fut}(e)\;) \parallel \varphi\}$
$\underset{loc}{\longmapsto} \nu\Sigma, a \sim \tau', b \sim \tau \; \{a \hookrightarrow (\; k \lhd \mathtt{fut}[b]\;) \otimes b \hookrightarrow (\; \epsilon \rhd e\;) \parallel \varphi, b \hookrightarrow \bullet\}$

A computation that begins this way can use a local transition to store its ultimate result in the memo table

— $\nu\Sigma, b \sim \tau \; \{b \hookrightarrow (\; \epsilon \lhd v\;) \parallel \varphi, b \hookrightarrow \bullet\}$
$\underset{loc}{\longmapsto} \nu\Sigma, b \sim \tau \; \{\emptyset \parallel \varphi, b \hookrightarrow v\}$

We *synchronize* on a future in order to create a computation that is waiting until the memo table contains a given value. Unlike synchronization in nested parallelism, this does not consume the memory binding, as we may still need to synchronize on the same memoized value again.

— $k \rhd \mathtt{fsyn}(e) \underset{\Sigma}{\longmapsto} k; \mathtt{fsyn}(\square) \rhd e$

— $\nu\Sigma, a \sim \tau', b \sim \tau \; \{a \hookrightarrow (\; k; \mathtt{fsyn}(\square) e \lhd \mathtt{fut}[b]\;) \parallel \varphi, b \hookrightarrow v\}$
$\underset{loc}{\longmapsto} \nu\Sigma, a \sim \tau', b \sim \tau \; \{a \hookrightarrow (\; k \lhd v\;) \parallel \varphi, b \hookrightarrow v\}$

Futures are simply intended to be reorderings of the existing computation, which means that if a future doesn't terminate, it's incorrect for the computation as a whole to terminate. This means that final states should only have one active process left: the original one, which wasn't associated with a $a \hookrightarrow \bullet$ binding in the memo table so won't get stored there or removed from the active set of computations.

$$\frac{\emptyset \vdash_{\emptyset} e : \tau}{\nu a \hookrightarrow \tau \; \{a \hookrightarrow (\; \epsilon \rhd e\;) \parallel \emptyset\}} \; \mathsf{initial} \qquad \frac{}{\nu\Sigma, a \hookrightarrow \tau \; \{a \hookrightarrow (\; \epsilon \lhd v\;) \parallel \varphi\}} \; \mathsf{final}_{\Sigma}$$

2 Speculations

The main difference between futures and speculations is that speculations do *not* need to run in order for the computation to terminate. If a speculation was never forced, then any work done on that speculation was wasted (and thus speculations are not work-efficient). Thus, we can model speculations instead of futures by changing the specification of final states to permit available computation in a final state as long as all that computation is for the benefit of un-forced speculations.

$$\frac{\forall (b \hookrightarrow s) \in \mu, \qquad (b \hookrightarrow \bullet) \in \varphi}{\nu\Sigma, a \hookrightarrow \tau \; \{a \hookrightarrow (\; \epsilon \lhd v\;) \otimes \mu \parallel \varphi\}} \; \mathsf{final}_{\Sigma}$$

This assumes we want to model speculations *instead of* futures. If we wanted to model both, we could do it with two different placeholder bindings in φ, for instance $a \hookrightarrow \bullet$ for speculations versus $b \hookrightarrow \circ$ for futures.