Modeling Heterogeneous Semantics in Ptolemy with Modular Actor Interfaces

Chris Shaver working with Professor Edward A. Lee
Models of Computation

- Controller
  - State Machine
- Sensor Network
  - Discrete Event
- Signal Processing
  - Synchronous Dataflow
- Physical Simulation
  - Continuous Time
Heterogeneous Systems

Different parts of a system and different levels of hierarchy can have different Models of Computation.
Reasoning About Heterogeneity

• There needs to be a way to reason about the composition of models with different Models of Computation.

• This is necessary for establishing a consistent semantics for heterogeneous systems.

• This can be used to do analyses and forms of verification that cross the boundaries of heterogeneous composition.
Ptolemy II

• An open source research platform for modeling systems.
• Implements an Actor Oriented language for model development.
• Allows hierarchical design and the heterogeneous composition of different Models of Computation.
• ~2.5 Million Lines of Code.
• Code generation.
Actors

Some Specific Actors

AddSubtract

FFT

Ramp

Plotter
Actor Graphs

Actor Graph

Ramp

DDFBooleanSelect

Scale 0.5

Const 0.5

Comparator

BooleanSwitch

Inside-the-loop Plotter

Outside-the-loop Plotter

SampleDelay {false}

test

Author: Gang Zhou
(based on Ptolemy Classic demo)
Models

The Director determines the semantics of the Actor Graph

Author: Gang Zhou
(based on Ptolemy Classic demo)
Dataflow Directors:
DDF, PSDF, CSDF, MDSDF, HDF, SDF, etc...

Producing Tokens
Actor

Ideal Queues

Consuming Tokens
Actor

SDF: Production and Consumption rates are fixed. Scheduability is decidable, and if a model is scheduable, it can be scheduled periodically with fixed-sized buffers.
Synchronous Directors:
SR, DE, CT, etc...

Determines Tokens  Single value per iteration  Depends on Tokens

SR: If actors in a model are monotonic then there exists a unique solution for the value of every relation that can be reached through a fixed-point iteration.
Composite Actors

<-- The Director determines the semantics of the Actor Graph

Author: Gang Zhou
(based on Ptolemy Classic demo)
Modular Actor Interface

Static Semantics
- Actor
  - Initial State
- State

Dynamic Semantics
- Fire
- Postfire
- Deadline
- Timeupdate
Fire

Produces outputs from inputs and state

Postfire

Updates to a new state from inputs and state
Deadline
Determines a deadline for execution from inputs and state

State

Timeupdate
Updates to a new state from inputs, state, and time
Directors compose the interfaces of Actors in a Model into those of the Composite Actor.
Mathematical Representations

The fixed-point semantics of SR can be understood in clear mathematical terms.

\[
\left( \tilde{F}_{s,x}(y) \right)(o) = F_j(s_j, (x, y) \upharpoonright I_j)(o)
\]

Letting \( y^*_{s,x} \) be the unique least fixpoint of \( \tilde{F}_{s,x} \).

\[
F(s, x) = y^*_{s,x} \upharpoonright O
\]

\[
P(s, x) = \left( P_1(s_1, (x, y^*_{s,x}) \upharpoonright I_1), \ldots, P_n(s_n, (x, y^*_{s,x}) \upharpoonright I_n) \right)
\]

From "A Modular Formal Semantics for Ptolemy", Tripakis et al.
Coroutine Model of Computation

Theorem 1. Given a non-strict Coroutine Model $\mathcal{M}$, if the input $\Pi_M$ and output $\Omega_M$ types of the model are finite-height pCPOs and operator $\oplus$ is monotonic, then the above recursive equations characterizing the kernel functions $\mathbf{e}$ and $\mathbf{f}$ have unique least fixed-point solutions in the partial order of functions with codomains $2^G$ and $\Omega_M$, respectively.

Theorem 2. Given a non-strict Coroutine Model $\mathcal{M}$, if the input $\Pi_M$ and output $\Omega_M$ types of the model are finite-height pCPOs and operator $\oplus$ is monotonic, and if for each $q \in Q$ the functions $\text{enter}_q$ and $\text{fire}_q$ are monotonic in terms of $\Pi_q$, and the mapping functions $m_{\Pi}$ and $m_{\Omega}$ are monotonic, then the non-strict kernels $\mathbf{e}$ and $\mathbf{f}$ are continuous in terms of $\Pi_M$.

I Conclude!

• Try out Ptolemy! It's open and free!
• Check out Modular Actor Interfaces.
• Feel free to design your systems modularly, hierarchically, and heterogeneously.
• http://ptolemy.eecs.berkeley.edu