Model Transformation with Hierarchical Discrete-Event Control

Thomas Huining Feng

Center for **H**ybrid and **E**mbedded **S**oftware **S**ystems

EECS, UC Berkeley

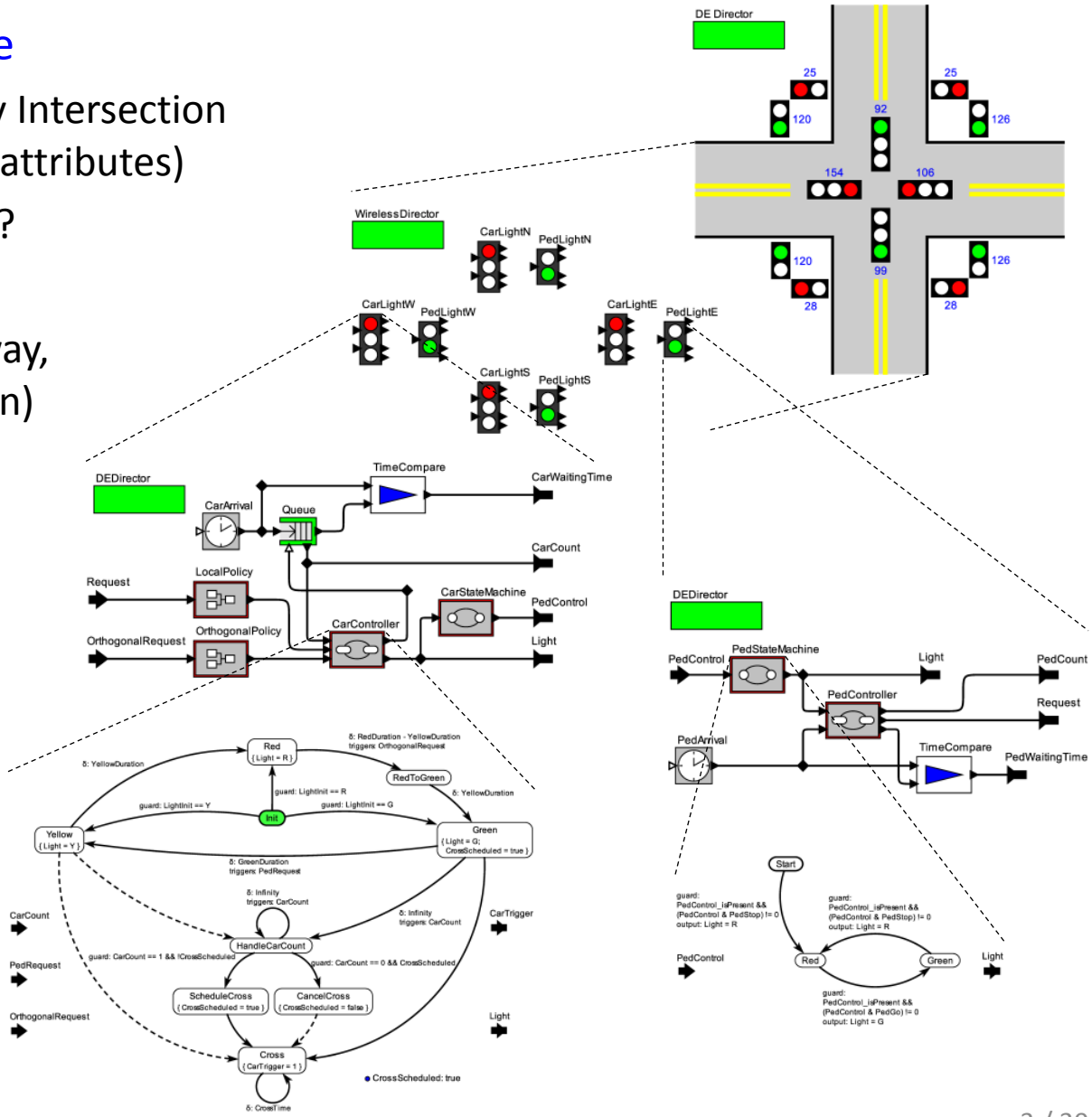
Advisor: Prof. Edward A. Lee

Outline

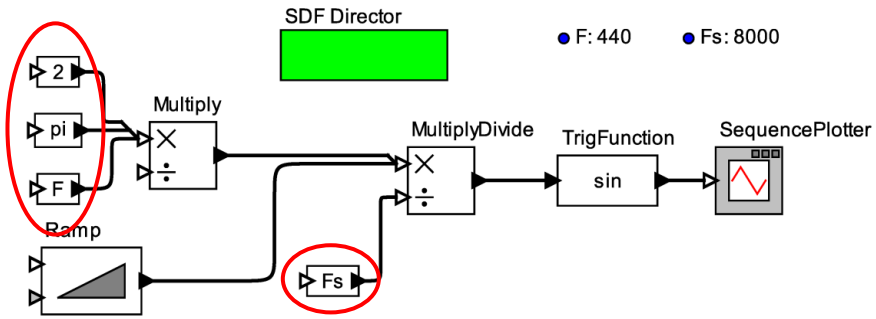
- **Motivation and Contribution**
- Transformation Rules: Syntax and Semantics
- Ptera (Ptolemy Event Relationship Actor)
- Model-Based Transformation
- Conclusion and Future Directions

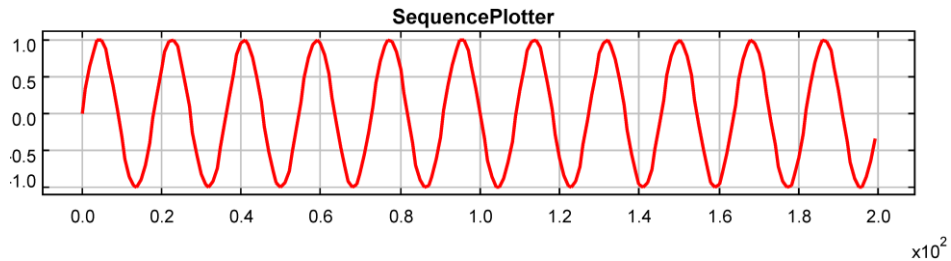
Motivation

- Models can get large
  - A simplified 4-way Intersection (230 actors, 2000 attributes)
  - How about Street?
  - How about City?
  - Other types? (*n*-way, irregular, European)
- Need automatic model construction and maintenance



Sinewave Example





- An SDF (synchronous dataflow) model to generate a sine wave with:

$$\sin\left(\frac{2\pi Fn}{Fs}\right)$$

- Static partial evaluation improves run-time performance (similar to program optimization in compilers)

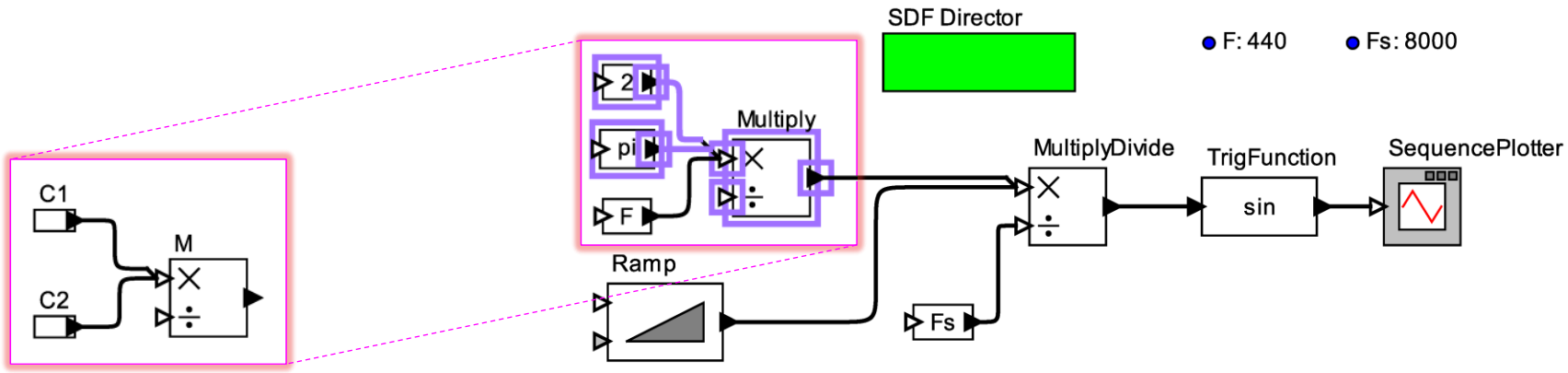
Contribution

- A model transformation tool
  - Convenient – an intuitive visual syntax
  - Flexible – compositional workflow
  - Efficient – by using an appropriate model of computation
- The Ptera model of computation
  - Syntax and semantics based on event graphs
  - Hierarchical composition
  - Composition with DE, FSM and more
- Applications
  - Transformation for actor models
  - Simulation of discrete-event processes

Outline

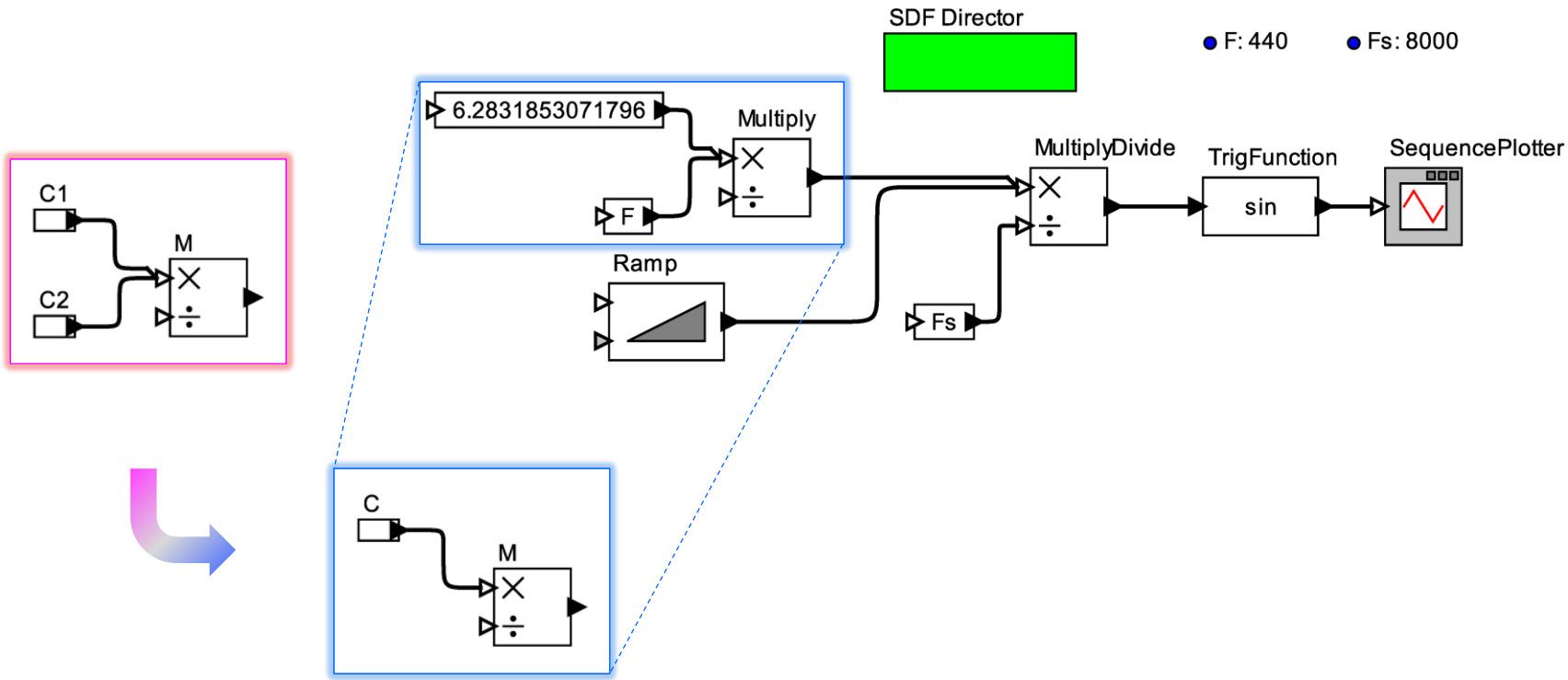
- Motivation and Contribution
- **Transformation Rules: Syntax and Semantics**
- Ptera (Ptolemy Event Relationship Actor)
- Model-Based Transformation
- Conclusion and Future Directions

Intuition of Model Transformation



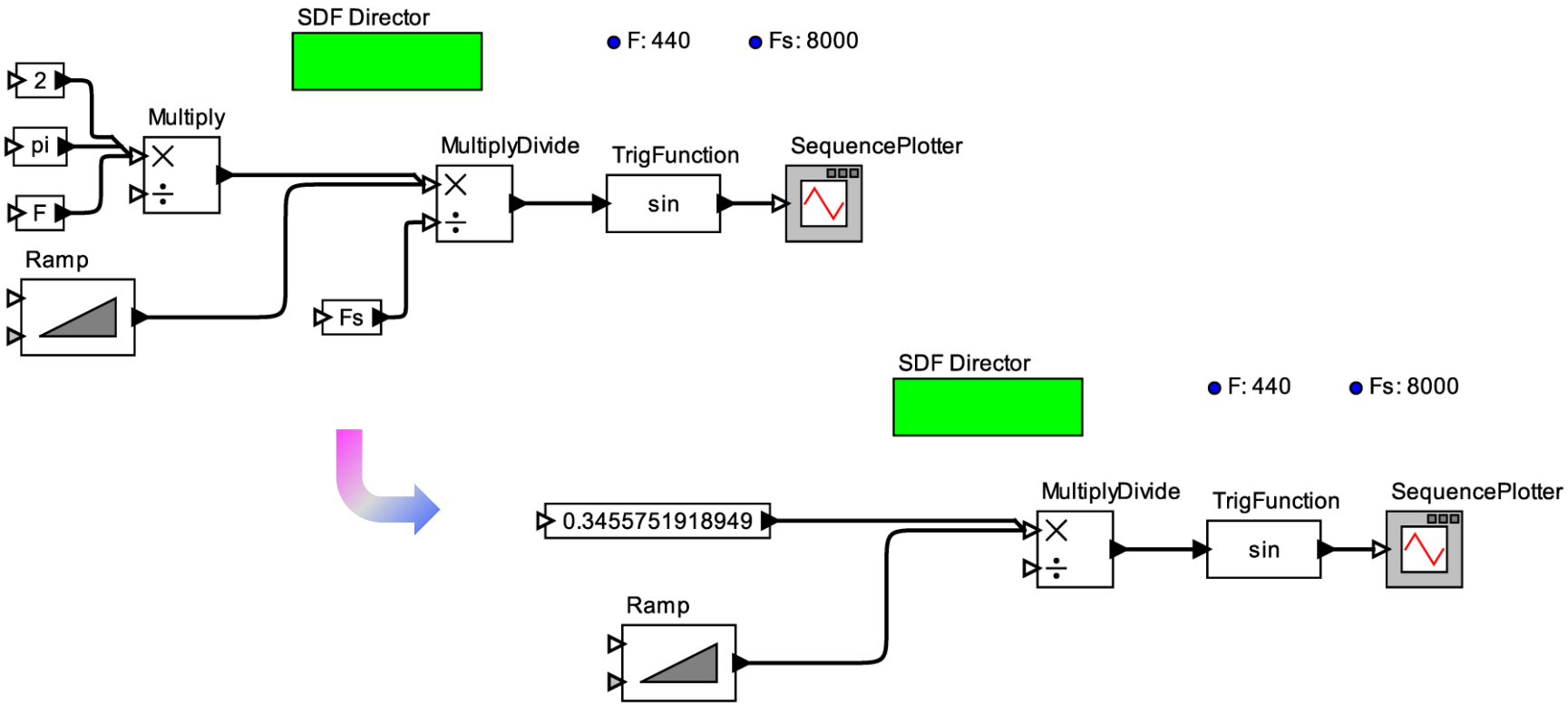
- Pattern matching (locate a part of the model)

Intuition of Model Transformation



- Pattern matching (locate a part of the model)
- Replace the part with a more compact design

Intuition of Model Transformation

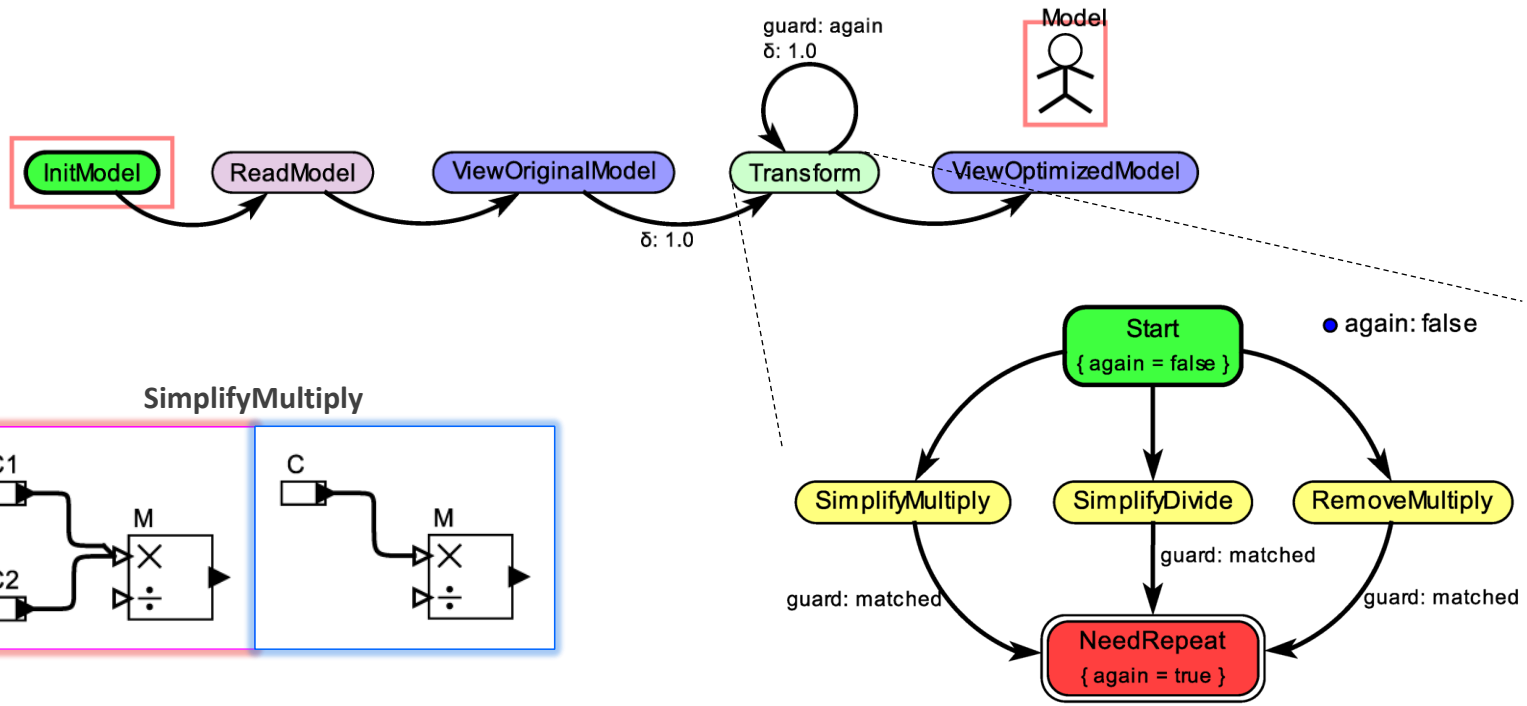


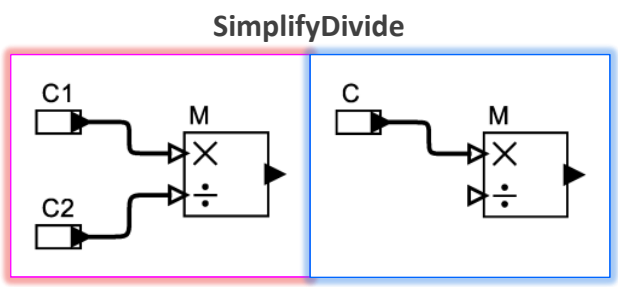
- Pattern matching (locate a part of the model)
- Replace the part with a more compact design
- Repeat transformations until fixpoint

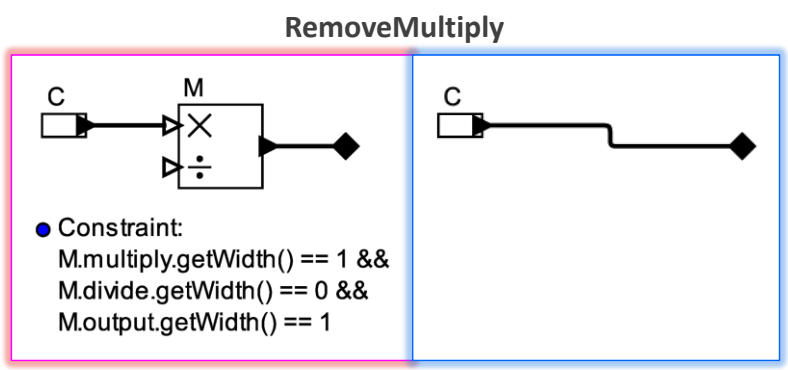
Control

- A control mechanism is needed
  - Transform until fixpoint
  - Try multiple alternative transformation rules
  - Pause and ask user a question
  - Branch with user input or current state
  - Perform IO before/after transformation
  - …
- Examples of existing approaches
  - Priority (AToM^3, AGG)
  - Control flow + Statecharts (FUJABA)
  - Abstract state machines (VIATRA2)
  - Dataflow + Implicit shared variables (GReAT)
  - Imperative programs (PROGRES)
- The Ptolemy II approach
  - Hierarchical heterogeneous models of computation
  - Ptera (Ptolemy event relationship actor)

A Ptera Transformation Workflow







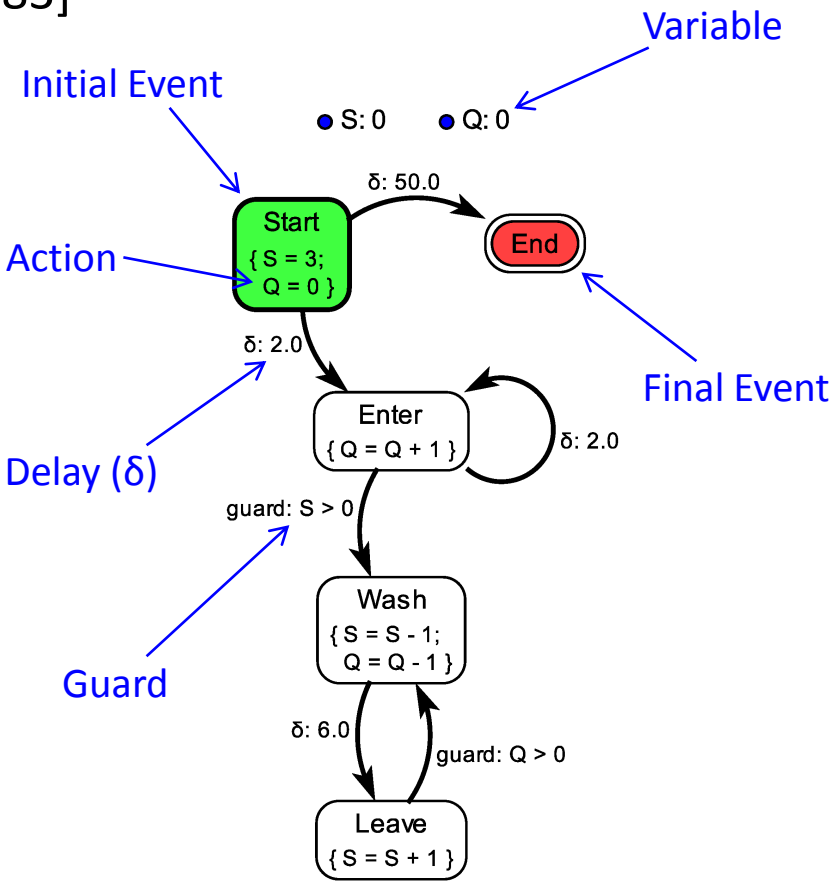
Outline

- Motivation and Contribution
- Transformation Rules: Syntax and Semantics
- **Ptera (Ptolemy Event Relationship Actor)**
- Model-Based Transformation
- Conclusion and Future Directions

Syntax

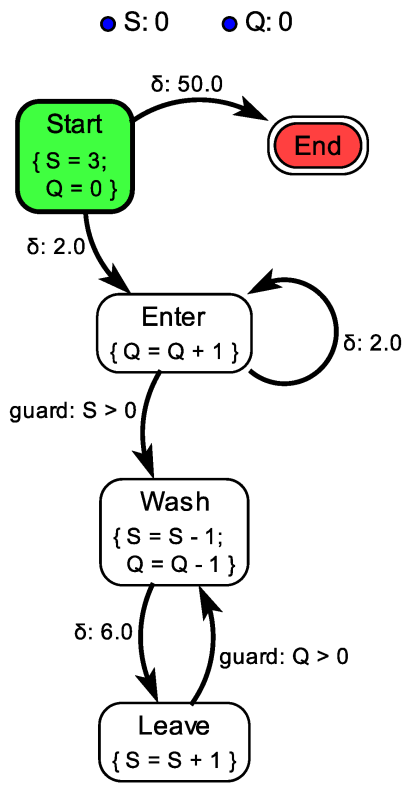
- Based on event graphs [Schruben 1983]
- Visual representation
  - Nodes are events
  - Edges are scheduling relations
- Compare to FSM
  - Actions on events
  - Can schedule multiple events
  - Timed

CarWash: single queue multiple servers

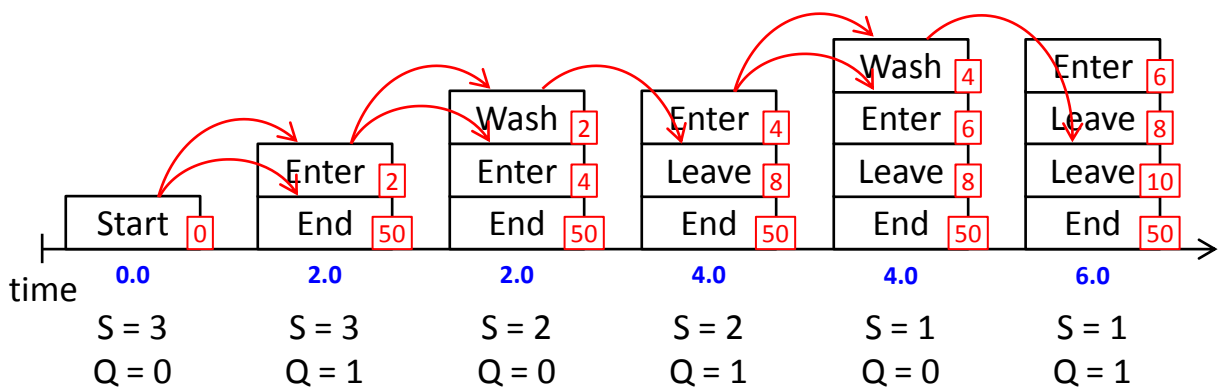


The CarWash model

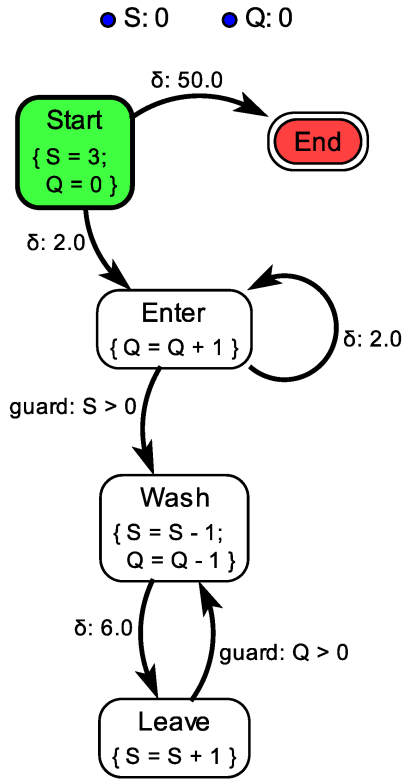
Execution



- During execution, the event queue stores *instances* of events
- Start by scheduling an instance of each initial events at time 0
- Remove and process the first instance in each *firing*
- Terminate when the event queue becomes empty



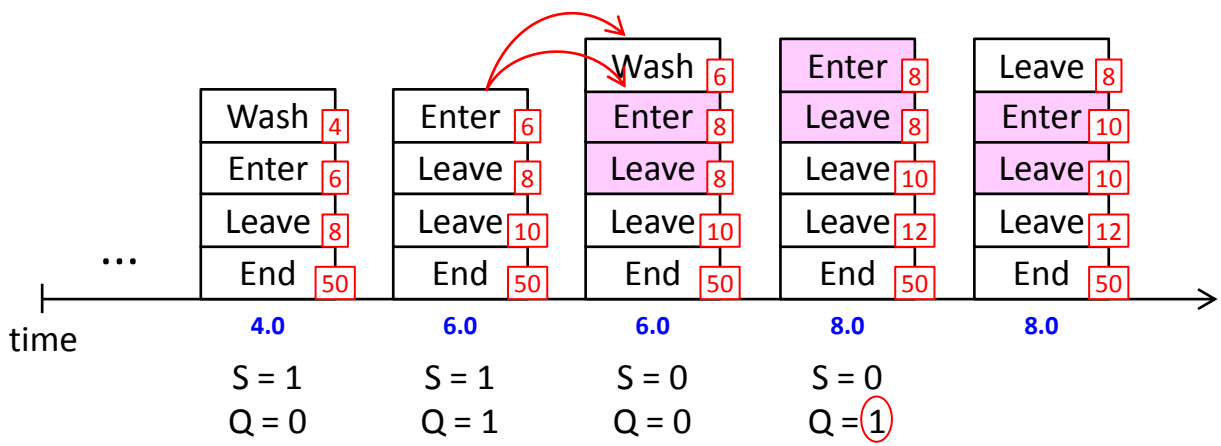
Simultaneous Events



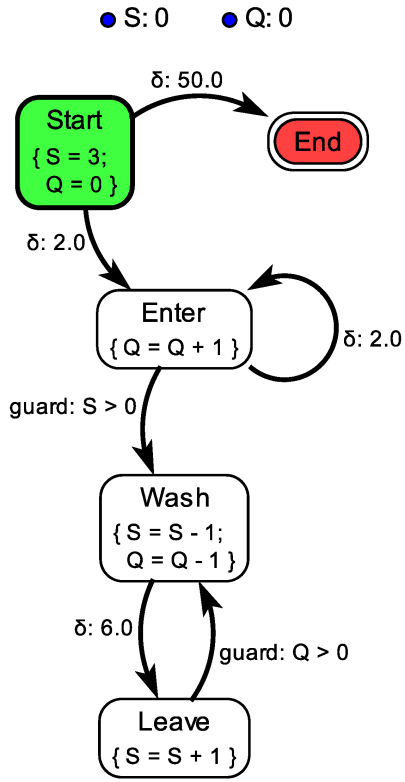
Events with instances that

1. coexist in the event queue, and
2. are scheduled to occur at the same time

E.g., Enter and Leave.



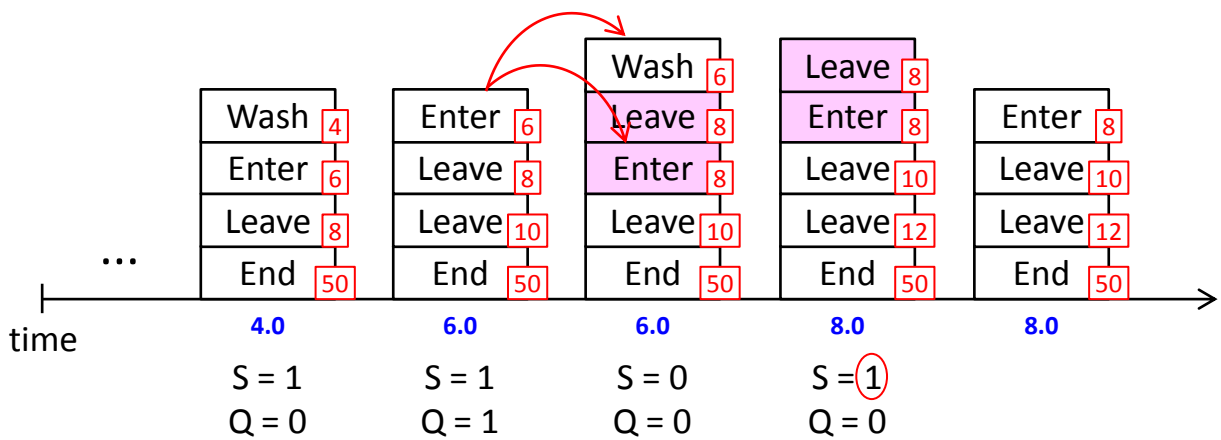
Simultaneous Events



Events with instances that

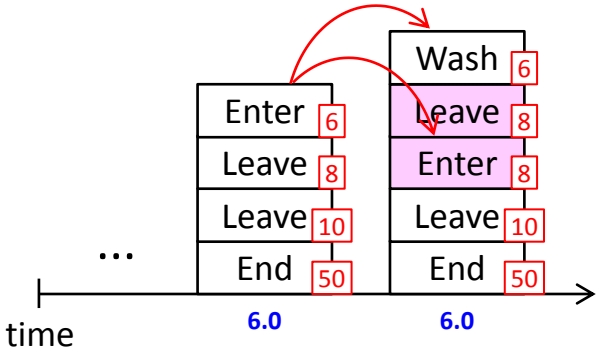
1. coexist in the event queue, and
2. are scheduled to occur at the same time

E.g., Enter and Leave.

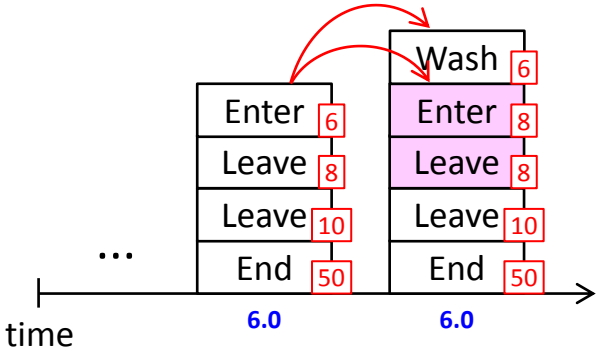


FIFO and LIFO Policies

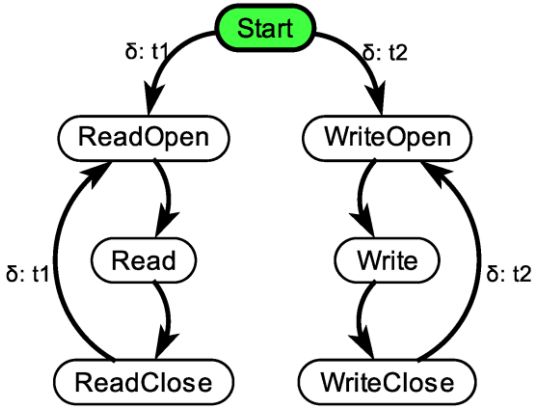
- With FIFO (First In First Out) policy



- With LIFO (Last In First Out) policy



Rationale for FIFO and LIFO

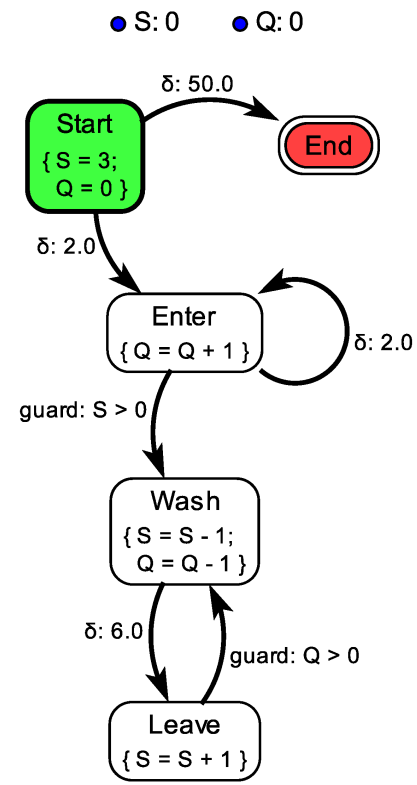


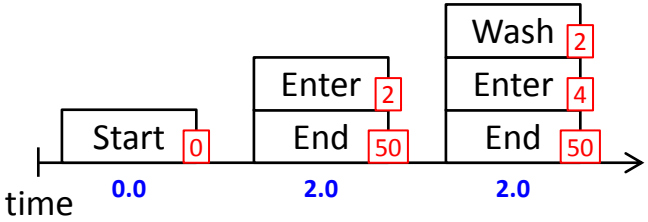
1. With FIFO, when x*t1 = y*t2 ∧ t1 > t2
   ReadOpen → WriteOpen → Read → Write → ReadClose → WriteClose

2. With LIFO, always
   (ReadOpen → Read → ReadClose), (WriteOpen → Write → WriteClose)

Execution Algorithm

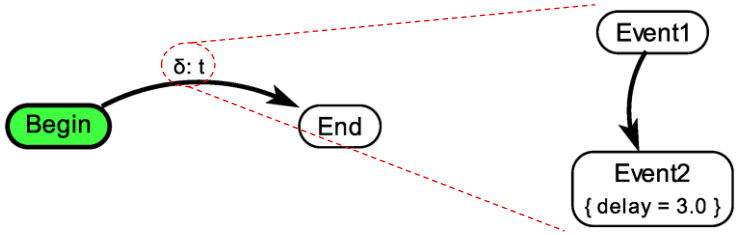
1. Initialize *E* to contain all initial events
2. While *E* is not empty
   a. Remove the top instance *t* from *E*
   b. Execute *t*'s actions
   c. Terminate if *t* is a final event
   d. Schedule events in *E* in the order of
      1. Time stamp
      2. FIFO or LIFO policy
      3. Event name
      4. Scheduling relation name



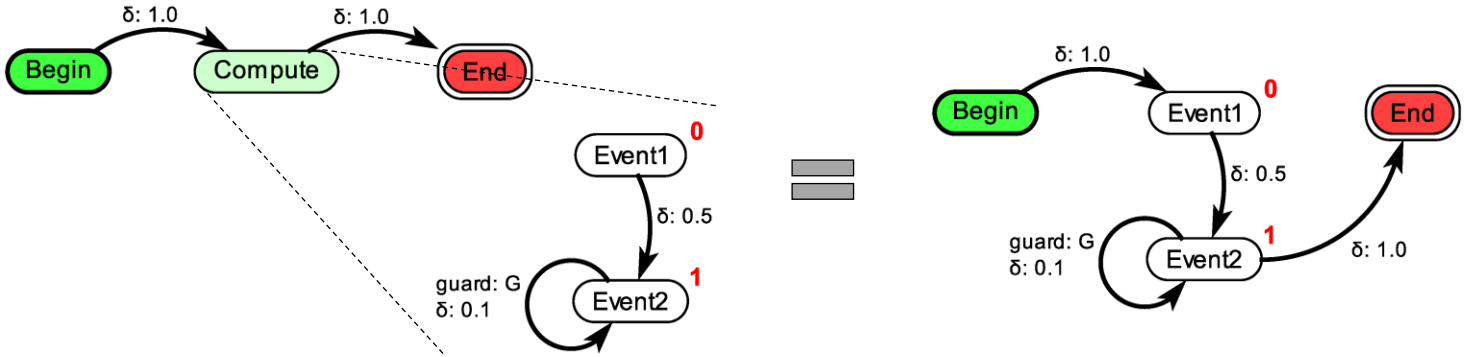


Model Hierarchy: Previous Attempts

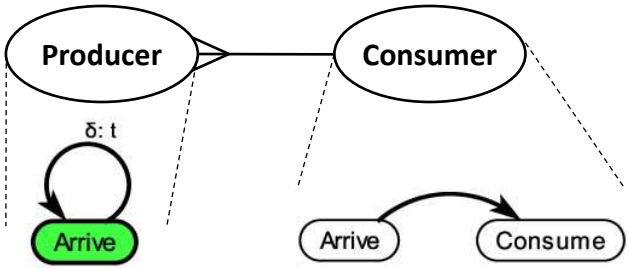
- Submodel associated with scheduling relation [Schruben 1995]



- Submodel associated with event [Schruben 1995]

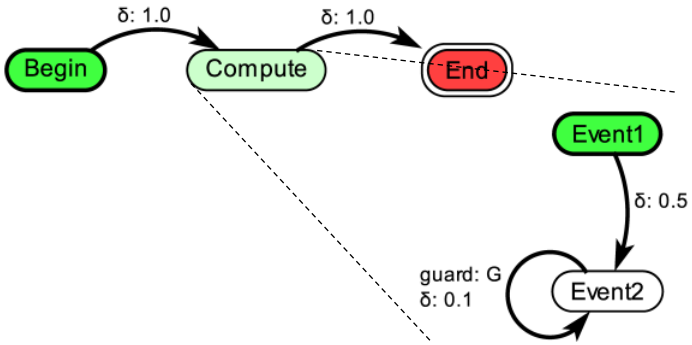


- LEGOs (Listener Event Graph Objects) [Buss & Sánchez 2002]



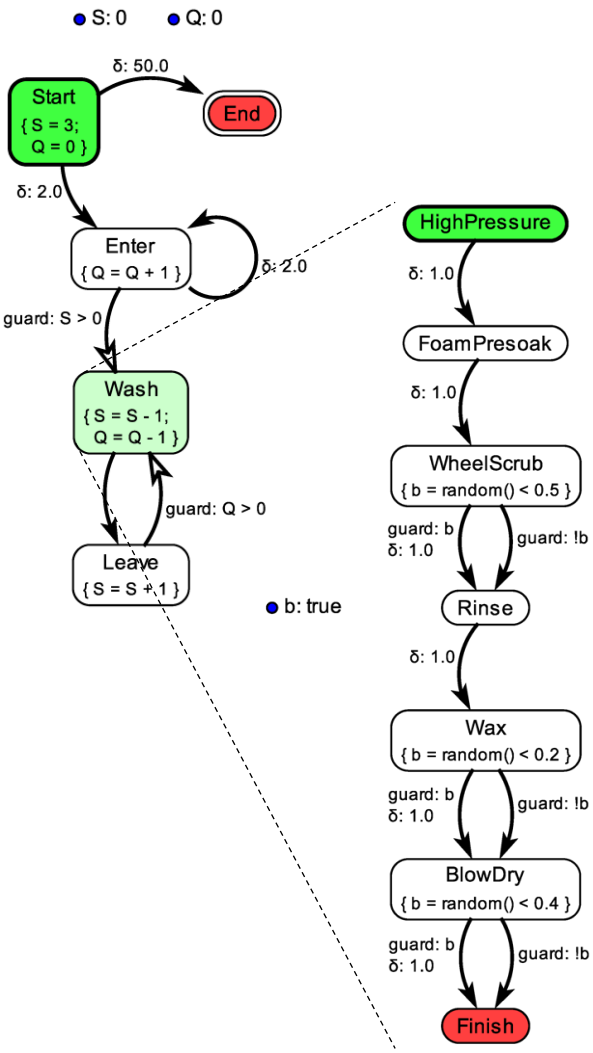
Model Hierarchy: The Ptera Approach

- A submodel is itself a model
  - No difference in syntax
  - Conceptually equipped with an isolated event queue
  - A global notion of model time

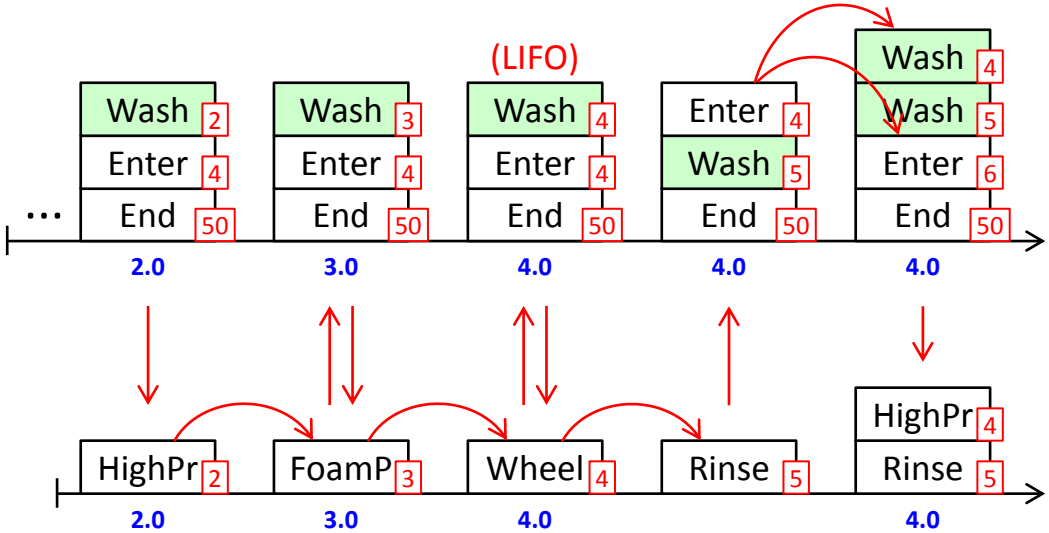


- Implication: events (or tasks) are no longer instantaneous
  - Start of an event causes start of its submodel
  - End of the submodel causes end of the event

Hierarchical CarWash



- Hierarchical workflow:
  - Global time, separate event queues
  - A composite task (CT) is composed of smaller tasks
  - Execution of CTs interleave
  - The whole workflow is a top-level CT

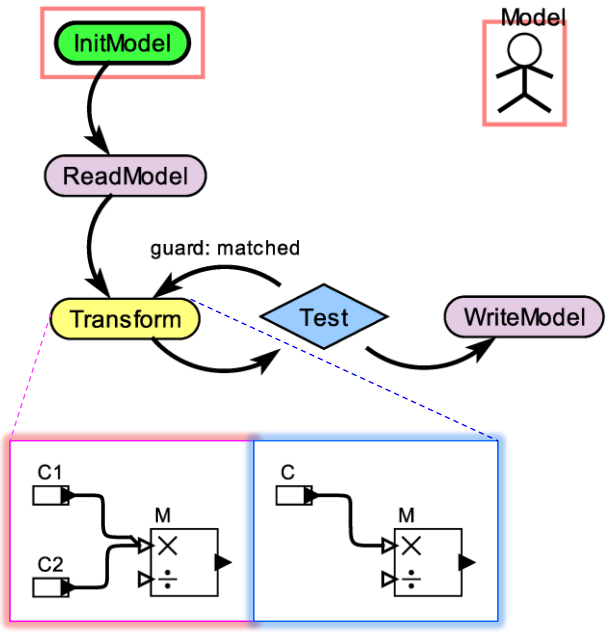


Outline

- Motivation and Contribution
- Transformation Rules: Syntax and Semantics
- Ptera (Ptolemy Event Relationship Actor)
- **Model-Based Transformation**
- Conclusion and Future Directions

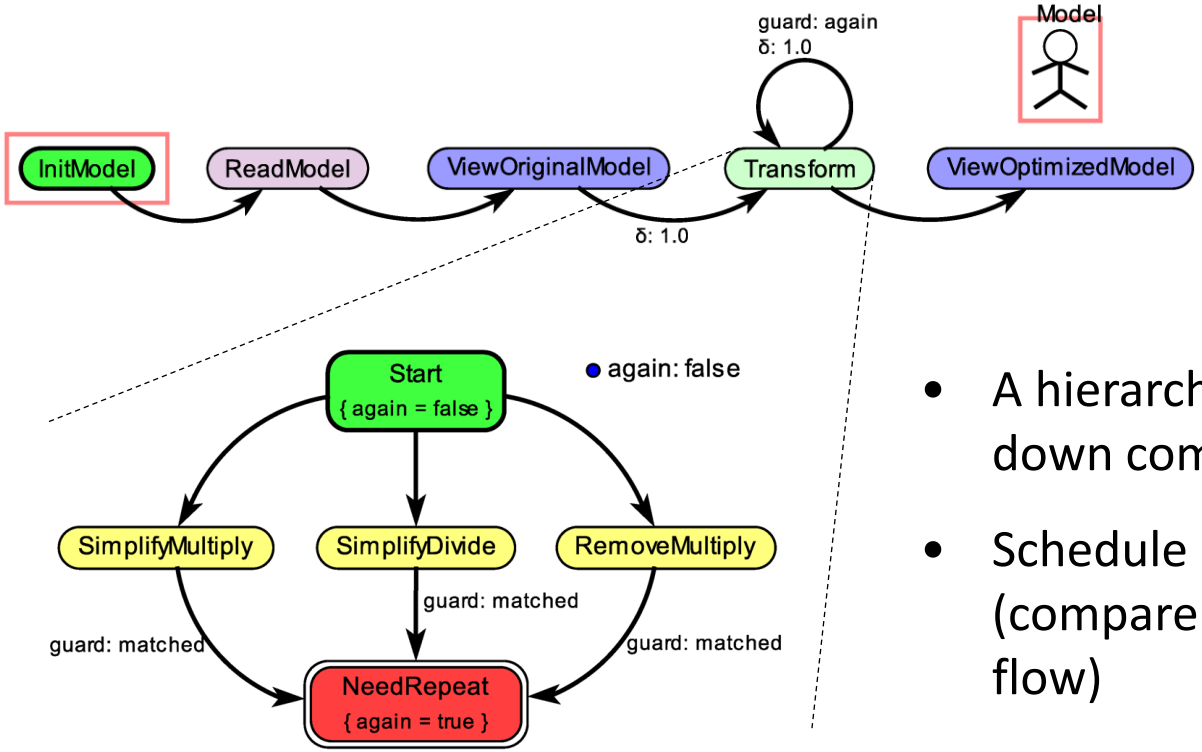
Transformation Workflow

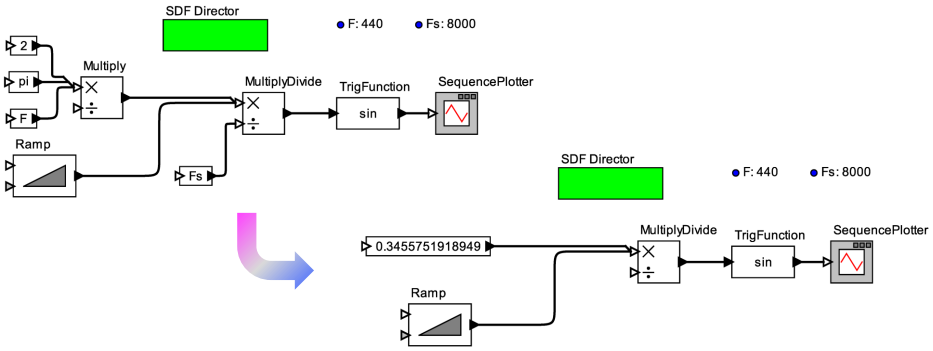
Parameter	Usage
Model	Temporarily store the model to be transformed
Event	**Action**
Configure	Pop up a dialog for the user to input values of some variables
Execute	Execute the model in the Model variable to completion
InitModel	Set the Model variable to contain an empty model
InitModelWithContainer	Set the Model variable to contain the container of this workflow
ListDirectory	List the names of all files in a directory
Match	Match a pattern with the model in the Model variable
ReadModel	Read the model stored in a file into the Model variable
Report	Report a message or an error to the user
Sleep	Wait for a certain amount of real time (in seconds)
Stop	Clear the event queue
Test	Perform pure testing (on the outgoing edges) with no side effect
Transform	Transform the model in the Model variable with a basic transformation
View	Show the model in the Model variable in a separate window
WriteModel	Output the model in the Model variable into a file



A simple workflow for repeated transformation

Hierarchical Workflow

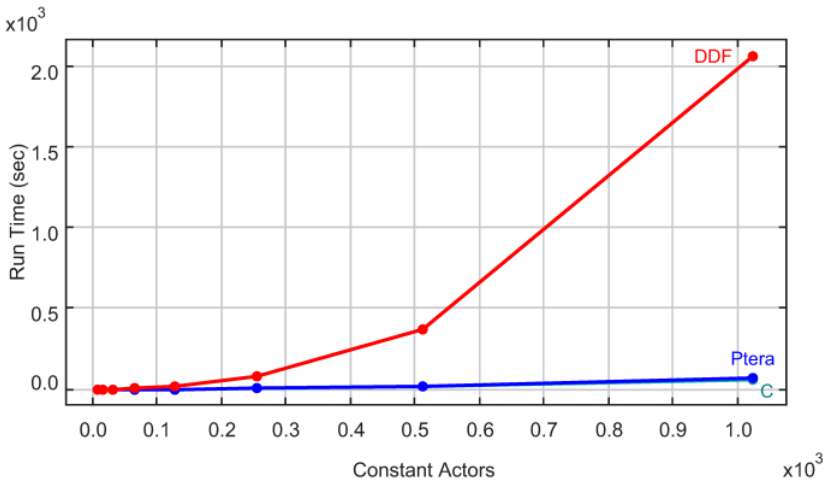




- A hierarchical workflow to break down complexity
- Schedule multiple events (compare to FSM and control flow)
- Model time is useful for
  - animation
  - specifying time or cost
  - simulation
  - …

Comparison

- Priority: not expressive enough
- Control flow, state machine: bounded state, not hierarchical, not timed
- Imperative program: lack of abstraction
- Dataflow: overhead of message passing

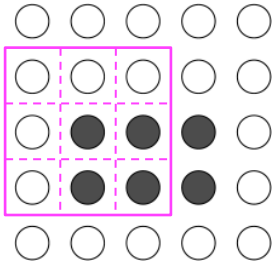


Const Actors	Run Time (sec)		
	Ptera	DDF	C (from DDF)
8	0.016	0.113	0.015
16	0.032	0.250	0.031
32	0.063	0.875	0.078
64	0.188	4.088	0.218
128	0.750	17.254	0.717
256	3.261	74.850	3.276
512	14.431	366.679	12.386
1024	72.736	2054.172	59.764

Shared variables are introduced in GReAT to improve performance, but they cause new problems.

Applications

- Model optimization
  E.g., partial evaluation, removal of nonfunctional parts
- Model construction
  E.g., generating a family of products
- Model refactoring
  E.g., flattening hierarchy, introducing hierarchy
- Simulation
  E.g., Conway's game of life



Outline

- Motivation and Contribution
- Transformation Rules: Syntax and Semantics
- Ptera (Ptolemy Event Relationship Actor)
- Model-Based Transformation
- **Conclusion and Future Directions**

Conclusion and Future Directions

Conclusion:

- Transformation rule in the same modeling language
- Model-based transformation is essentially a model
- Ptera
  - Convenient and expressive
  - Compositional
  - Timed
  - Avoids message passing

Future Directions:

- Analyze properties of transformations
- Compose Ptera with other MoCs (DE and FSM studied)
- Model checking (bound of event queue, simultaneous events, termination condition, …)
- Behavior-preserving concurrent and distributed execution

Acknowledgments

- Dissertation committee
  - Prof. Edward A. Lee (chair, advisor)
  - Prof. Rastislav Bodik
  - Prof. Lee W. Schruben
- Students, researchers and staff in Cory (545.*)
- Christopher Brooks and Mary Steward
- Industrial collaborators
  - Naomi team at Lockheed Martin ATL
  - Thomas Mandl at Bosch research
- Prof. Hans Vangheluwe (former advisor at McGill)
- Friends in EECS
- Family