Template based code generation for networks of hybrid systems

Cambdridge CodeGen Workshop

Boris Marin

Silver Lab
University College London

January 7, 2014

UCL

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

Requirements

(in the context of Computational Neuroscience)

- discrete *units*, modeled as dynamical systems
  - ODEs / maps
  - SDEs / kinetic schemes
- discontinuous state jumps (e.g. synaptic coupling)
  - event detection (rootfinding)
- networks of *units*
  - event propagation

In summary, *networks of hybrid systems*

LEMS: a general purpose model specifiation language

SPOILER ALERT!

LEMS: a general purpose model specifiation language

```
<ComponentType name="iafTauRefCell"
               extends="iafTauCell"
               description="Integrate and fire cell">

  <Parameter name="refract" dimension="time"/>

  <Dynamics>
    <StateVariable name="v" exposure="v" dimension="voltage"/>
    <StateVariable name="lastSpikeTime" dimension="time"/>

    <OnStart>
      <StateAssignment variable="v" value="reset"/>
    </OnStart>

    <Regime name="integrating" initial="true">
      <TimeDerivative variable="v" value="(leakReversal - v)/tau" />
      <OnCondition test="v .gt. thresh">
        <EventOut port="spike"/>
        <Transition regime="refractory" />
      </OnCondition>
    </Regime>

    <Regime name="refractory">
      <OnEntry>
        <StateAssignment variable="lastSpikeTime" value="t" />
        <StateAssignment variable="v" value="reset" />
      </OnEntry>
      <OnCondition test="t .gt. lastSpikeTime + refract">
        <Transition regime="integrating" />
      </OnCondition>
    </Regime>

  </Dynamics>

</ComponentType>
```

From MSL to simulation

Bottom line: Simulation of hybrid systems is a well established field.
Why not using state of the art numerical libraries?

- numerical stability
  - stiffness
  - Zeno
- error control (variable stepsize)
- accurate event detection

From MSL to simulation

Bottom line: Simulation of hybrid systems is a well established field.
Why not using state of the art numerical libraries?

- numerical stability
  - stiffness
  - Zeno
- error control (variable stepsize)
- accurate event detection

From MSL to simulation

Bottom line: Simulation of hybrid systems is a well established field.
Why not using state of the art numerical libraries?

- numerical stability
  - stiffness
  - Zeno
- error control (variable stepsize)
- accurate event detection

From MSL to simulation

Bottom line: Simulation of hybrid systems is a well established field.
Why not using state of the art numerical libraries?

- numerical stability
  - stiffness
  - Zeno
- error control (variable stepsize)
- accurate event detection

From MSL to simulation

Bottom line: Simulation of hybrid systems is a well established field.
Why not using state of the art numerical libraries?

- numerical stability
  - stiffness
  - Zeno
- error control (variable stepsize)
- accurate event detection

The need for an intermediate format

- template-based code generation: attract expert users
  - close correspondence to target structure
  - most targets use a similar format
- alternative approach: develop a compiler

The need for an intermediate format

- template-based code generation: attract expert users
  - close correspondence to target structure
  - most targets use a similar format
- alternative approach: develop a compiler

The need for an intermediate format

- template-based code generation: attract expert users
  - close correspondence to target structure
  - most targets use a similar format
- alternative approach: develop a compiler

The need for an intermediate format

- template-based code generation: attract expert users
  - close correspondence to target structure
  - most targets use a similar format
- alternative approach: develop a compiler

The distilled LEMS format

Goal: direct mapping to widely used ODE steppers with event handling

The distilled LEMS format

```
{
    "name":"izhikevich_burster",
    "state":{
        "v":"v0",
        "u":"b*v0"
    },
    "state_functions":{
        "phi":"0.04 * v**2 + 5*v + 140"
    },
    "dynamics":{
        "v":"phi - u + I",
        "u":"a * (b * v - u)"
    },
    "parameters":{
        "a":"0.02",
        "b":"0.2",
        "c":"-50",
        "d":"2",
        "I":"0",
        "v0":"-70"
    },
    "t_start":"0",
    "t_end":"300",
    "dt":"0.01",
```

```
"events":[
    {
        "name":"spike",
        "condition":"v - 30",
        "direction":"+",
        "effect":{
            "state":{
                "v":"c",
                "u":"u + d"
            }
        }
    },
    {
        "name":"start_inj",
        "condition":"t - 30",
        "direction":"+",
        "effect":{
            "parameters":{
                "I":"5"
            }
        }
    },
    {
        "name":"end_inj",
        "condition":"t - 150",
        "direction":"+",
        "effect":{
            "parameters":{
                "I":"0"
            }
        }
    }
],
```

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)

Current status

LEMS $\mapsto$ dLEMS $\mapsto$

- LLNL Sundials (CVODE/IDA)
- matlab ODEsuite
- XPPAUT
- modelica
- (nearly) effortless accomodation of addditional formats
- (nodes only: no event routing)