CellML SBGN SBO BioPAX MIASE Workshop 2009 Waiheke Island

CellML time delays and events

Poul Nielsen

Declarative descriptions

- CellML and SBML are declarative model specification languages.
- i.e. relationships and dynamics are specified as algebraic and differential/integral equations.
- All relationships are assumed to hold simultaneously.
- This approach works well for systems that vary continuously.
- However, it is difficult to express, using only algebra and calculus, systems that involve discontinuities such as state transitions and/or changes in topology.
- Cell cycle models and state transition models are examples where such processes occur.

Problem

- How can a declarative specification of a model accommodate discontinuous descriptions?
- One approach is to introduce the notion of and ‘event’ that describes both the trigger and nature of a discontinuous transition.
- SBML uses this approach, where an event consists of:
  - one *trigger*, a single logical expression that fires an event when transitioning from ‘false’ to ‘true’;
  - zero or one *delay*, specifying the time between when the trigger is fired and the event assignment is *executed*;
  - one or more *event assignments* that are effected when the event is executed.
- This is a pragmatic approach that seems to fit well with implicit time descriptions.

Issues

- Why should a delay element have special significance in events? Would it not be better to define a general delay function that could be used anywhere mathematical expressions are permitted?
- CellML does not give any special status to time, so the *delay* element would probably be better described as a *shift* operation where the variable that is shifted with respect to must be specified.
- Defining variables that are shifted with respect to other variables raises an issue for initial value problems – what happens when a variable is defined with respect to second variable outside the domain of the second variable? My inclination is to make such definitions illegal.

Issues

- The event element introduces a procedural flavour to an otherwise declarative description of the model.
- Is this the only way to specify discontinuities?
  - Can we define events in terms of the MathML *piecewise* element?
  - Can step changes be achieved by using functions of inherently discontinuous functions (such as the **rem**ainder operator)?
  - Can we use indexed variables to define variables after an event (index is $n + 1$) in terms of variables before an event (index is n)?
  - How do other purely declarative languages deal with this issue?