

Conversión de modelos PowerDEVS al lenguaje Modelica

Tesinista: Luciano Andrade
Director: Federico Bergero, Co-Director: Ernesto Kofman

25 de enero de 2015

Índice general

1. Introducción	1
1.1. Motivación y Objetivos	1
1.2. Trabajo relacionado	2
1.3. Alcance	2
2. Conceptos Previos	3
2.1. Modelado y Simulación	3
2.1.1. Sistemas Continuos y Discretos	3
2.1.2. Métodos de Integración numérica	3
2.2. Formalismo DEVS	4
2.2.1. Atómicos	4
2.2.2. Acoplados	4
2.2.3. Modelos Vectoriales	4
2.3. Métodos de integración QSS	4
2.4. PowerDEVS	4
2.5. Modelica	4
2.6. QSS Stand Alone Solver	4
2.6.1. μ Modelica	4
3. Conversión de modelos DEVS	5
3.1. Modelos Atómicos	5
3.2. Modelos Vectoriales	5
3.3. Modelos Acoplados Planos	5
3.4. Equivalencia semántica de la conversión	5
3.5. Modelos Acoplados Jerárquicos	5
3.5.1. Algoritmo de aplanado	5
3.6. Comparación de performance	6
4. Detalles de Implementación	7
4.1. Ejemplos de Aplicación	7
4.1.1. Vector/airs	7
4.1.2. Vector/lcline	7
4.2. Conclusiones y Trabajo a futuro	7

Resumen

En este trabajo se describe la implementación de una aplicación para convertir modelos descritos en la herramienta PowerDEVS a modelos en el lenguaje Modelica, más específicamente en μ Modelica, con el fin de aprovechar la velocidad de simulación del 'QSS Solver', permitiendo describir las simulaciones en el entorno PowerDEVS y ejecutando las simulaciones en 'QSS Solver'

Capítulo 1

Introducción

1.1. Motivación y Objetivos

PowerDEVS es una herramienta de simulación de sistemas híbridos, basado en el formalismo DEVS, con una interfaz grafica orientada a bloques, donde los bloques pueden ser conectados entre si, modificado los parametros, posibilidad de conectarse con el entorno Scilab para poder utilizar expresiones y herramientas de cálculo provistas por este entorno.

La resolución de ecuaciones diferenciales ordinarias, requiere el uso de métodos de integración numérica. Todos los algoritmos tradicionales de integración se basan en la discretización de la variable independiente (que generalmente representa el tiempo). Las rutinas que implementan estos algoritmos, se denominan solvers y existen gran variedad de implementaciones de los mismos en diferentes lenguajes de programación. Los Métodos de Integración Numérica QSS (Quantized State System), a diferencia de los métodos de integración tradicionales, realizan la discretización sobre las variables de estado. En consecuencia, convierten los sistemas continuos en sistemas de eventos discretos, y tienen grandes ventajas para simular sistemas con discontinuidades. Si bien PowerDEVS, implementa la totalidad de los algoritmos de QSS, resultan ineficientes, dado que malgastan gran parte de la carga computacional en la transmisión de eventos entre submodelos.

Para solventar este hecho se desarrollo una familia de QSS stand-solver, el cual requiere un modelo descrito en lenguaje C el cual contiene las ecuaciones diferenciales, las funciones de cruce de cero así como la información estructural requerida por los algoritmos QSS. Estos solvers obtienen una mejora de performance de hasta un orden de magnitud comparado con otras implementaciones DEVS. Sobre este se desarrollo una herraminta la cual genera a partir de un modelo μ -Modelica (un subconjunto del lenguaje Modelica) el modelo requerido para el QSS solver.

Con el objetivo de utilizar las mejoras de velocidad y mantener un entorno amigable con el usuario, se creo una herramienta capas de convertir

un modelo PowerDEVS en un modelo μ -Modelica.

1.2. Trabajo relacionado

¿Cuales?

1.3. Alcance

DEVS, Discrete Event System Specification (Especificación de Sistemas de Eventos Discretos), es un formalismo modular y jerarquico para modelar y analizar sistemas que pueden ser de eventos de tiempo discreto mediante tablas de transición, y con estados continuos mediante que pueden ser descriptos mediante ecuaciones diferenciales.

En el formalismo clasico DEVS, los modelos atómicos capturan el comportamiento del sistema, mientras los modelos Acoplados describen la estructura del sistema.

En particular los modelos atómicos en PowerDEVS son descriptos en clases C++, mientras que la estructura se encuentra definida en archivos pds y pdm.

Modelica es un lenguaje de modelado, orientado a objetos, declarativo, para el modelado orientado a componentes de sistemas complejos.

Para poder realizar nuestro objetivo es necesario primero contar con un modelo en modelica para cada atómico PowerDEVS que deseemos convertir.

De esto se desprenden las siguientes limitaciones importantes:

- La semántica de los modelos convertidos depende de los modelos equivalentes a los DEVS atómicos
- Solamente podemos convertir modelos cuyos componentes atómicos estén o puedan ser convertidos a μ modelica.

Capítulo 2

Conceptos Previos

En este capítulo introducimos algunos conceptos básicos, necesarios para poder comprender este trabajo.

2.1. Modelado y Simulación

Modelado y Simulación de un Sistema es el proceso por el cual se desarrolla un modelo, el cual es luego simulado, de forma de obtener datos sobre el sistema. El modelo debe conservar las principales características del sistema, pero al mismo tiempo ser significativamente más simple, de forma que al momento de simularlo sea más eficaz utilizar la simulación que el sistema en si.

2.1.1. Sistemas Continuos y Discretos

Se considera un sistema continuo si las variables de este son conocidas en cada instante de tiempo, mientras que se considera discreto si las variables son conocidas en instantes de tiempo determinados.

En general los sistemas en estudio serán continuos, pero deberemos utilizar sistemas discretos puesto que la simulación en computadora así lo requiere, puesto que la misma computadora es un sistema discreto.

2.1.2. Métodos de Integración numérica

Un sistema continuo puede ser descripto por un modelo en espacios de estados de la forma:

$$\dot{x}(t) = f(x(t), u(t)) \quad (2.1)$$

donde $x \in \mathbb{R}^n$ es el vector de estados, $u \in \mathbb{R}^m$ es una función de entradas conocidas, t representa el tiempo y con sus condiciones iniciales:

$$x(t = t_0) = x_0 \quad (2.2)$$

Sea $x_i(t)$ la trayectoria del estado i -ésimo expresada como función de tiempo simulado. Mientras que la ecuación (2.1) no contenga discontinuidades $x_i(t)$ será una función continua con derivada continua. Esta puede ser aproximada con la precisión deseada mediante series de Taylor en cualquier punto de su trayectoria

Denominando t^* al instante de tiempo en torno al cual se aproxima la trayectoria mediante una serie de Taylor, y siendo $t^* + h$ el instante de tiempo en el cual se quiere evaluar la aproximación, entonces, la trayectoria en dicho punto puede expresarse como sigue:

$$x_i(t^* + h) = x_i(t^*) + \frac{dx_i(t^*)}{dt} \cdot h + \frac{d^2x_i(t^*)}{dt^2} \cdot \frac{h^2}{2!} + \dots \quad (2.3)$$

Reemplazando con la ecuación de estado (2.1), la serie (2.3) queda:

$$x_i(t^* + h) = x_i(t^*) + f_i(t^*) \cdot h + \frac{d^2x_i(t^*)}{dt^2} \cdot \frac{h^2}{2!} + \dots \quad (2.4)$$

2.2. Formalismo DEVS

Ber12 pag. 20

2.2.1. Atómicos

2.2.2. Acoplados

2.2.3. Modelos Vectoriales

BK11

2.3. Métodos de integración QSS

QSS paper

2.4. PowerDEVS

Powerdevs Paper

2.5. Modelica

2.6. QSS Stand Alone Solver

QSSPaper

2.6.1. μ Modelica

Capítulo 3

Conversión de modelos DEVS

3.1. Modelos Atómicos

Cómo se traducen (es conocimiento del modelador, no automático)

3.2. Modelos Vectoriales

Consideraciones y anotaciones en modelica, modificaciones

3.3. Modelos Acoplados Planos

Modelos acoplados solo con modelos atómicos adentro.

Mencionar el algoritmo (traducción de conexiones y “aplanado” de cada uno de los atómicos hijos“)

3.4. Equivalencia semántica de la conversión

3.5. Modelos Acoplados Jerárquicos

Explicar cuándo se utilizan y que resolvemos el problema aplanando los acoplados

3.5.1. Algoritmo de aplanado

Describir el algoritmo para PDS

3.6. Comparación de performance

¿No debería estar despues de .Ejemplos de Aplicación¿

Capítulo 4

Detalles de Implementación

API Powerdevs, AST Modelica Traverser Modelica Transformer

4.1. Ejemplos de Aplicación

tamaños de los vectores y comparativas de

4.1.1. Vector/airs

4.1.2. Vector/lcline

4.2. Conclusiones y Trabajo a futuro

Bibliografía

- [1] Federico Bergero, Xenofon Floros, Joaquín Fernández, Ernesto Kofman, and François E. Cellier. Simulating Modelica models with a Stand-Alone Quantized State Systems Solver. In *9th International Modelica Conference*, 2012. Aceptado.
- [2] Federico Bergero and Ernesto Kofman. PowerDEVS: a tool for hybrid system modeling and real-time simulation. *Simulation*, 87(1–2):113–132, 2011.
- [3] François Cellier and Ernesto Kofman. *Continuous System Simulation*. Springer, New York, 2006.
- [4] Joaquín Fernández and Ernesto Kofman. Implementación autónoma de métodos de integración numérica QSS. Technical report, FCEIA - UNR, Rosario, Argentina, 2012.
- [5] Xenofon Floros, Federico Bergero, Francois E. Cellier, and Ernesto Kofman. Automated Simulation of Modelica Models with QSS Methods - The Discontinuous Case. In *8th International Modelica Conference*, March 2011.
- [6] Peter Fritzson and Vadim Engelson. Modelica - A Unified Object-Oriented Language for System Modelling and Simulation. In *ECOOP*, pages 67–90, 1998.