Formal Verification of ECML using HyTech
(ECML: ETRI CPS Modeling Language)

JUNBEOM YOO
KONKUK University
2012.08.23
Contents

• Project Motivation

• CPS Modeling & Verification Techniques
  – ECML (ETRI CPS Modeling Language)
  – HyTech

• Formal Verification of ECML using HyTech
  – ECMLtoHyTech Translator
  – HyTech Analyzer

• Conclusion and Future Work
Project Motivation

• ETRI CPS Team
  – Developing a framework of CPS modeling, simulation and verification, from 2010
  – Proposed a new CPS modeling language – ECML
  – Not yet supporting formal verification/Analysis of ECML

• KONKUK University
  – Joined the ETRI CPS Project in 2011
  – Trying to develop a way to verify ECML models with existing CPS verification tools
  – Much troubled, since we didn’t know hybrid systems.
Our Effort for Finding a Research Starting Point

Chronology of Hybrid System Researchers from CS

- Timed Automata
- Hybrid Automata
- Linear Hybrid Automata
- HyTech
- Rajeev Alur
- Insup Lee
- Thomas A. Henzinger
- Wonhong Nam
- Moonzoo Kim
- CHARON
- [TMS’99]
- DEV
- DEV&DESS
- DEVSim++
- Tag Gon Kim

Excerpted from a presentation to ETRI in 2011.06
# CPS Modeling & Verification Techniques

<table>
<thead>
<tr>
<th>Name</th>
<th>Objective</th>
<th>Input front-end</th>
<th>Verification method</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARON[21]</td>
<td>modelling, simulation</td>
<td>CHARON language</td>
<td>none</td>
</tr>
<tr>
<td>CheckMate[22]a</td>
<td>verification</td>
<td>autonomous linear hybrid automata</td>
<td>rectangular polytopes automation</td>
</tr>
<tr>
<td>ddt[12]</td>
<td>verification</td>
<td>linear hybrid automata</td>
<td>over-approximation</td>
</tr>
<tr>
<td>Ellipsoidal ToolBox[23]a</td>
<td>verification</td>
<td>controlled linear hybrid system</td>
<td>polyhull method[24]</td>
</tr>
<tr>
<td>GBT[25]a</td>
<td>computation</td>
<td>polytope, ellipsoid</td>
<td>convex hull determination</td>
</tr>
<tr>
<td>HSIF[26]</td>
<td>modelling, simulation</td>
<td>network(collection of hybrid automata)</td>
<td>none</td>
</tr>
<tr>
<td>HSolver[27]</td>
<td>verification</td>
<td>input hybrid system</td>
<td>constraint propagationb</td>
</tr>
<tr>
<td>HyTech[10]</td>
<td>verification</td>
<td>linear hybrid automata</td>
<td>quantifier elimination, validity checking</td>
</tr>
<tr>
<td>HyVisual[28]</td>
<td>modelling</td>
<td>embedded systems</td>
<td>none</td>
</tr>
<tr>
<td>KeYmaera[29]</td>
<td>verification</td>
<td>differential dynamic logic</td>
<td>symbolic decompositionb</td>
</tr>
<tr>
<td>Level Set ToolBox[30]a</td>
<td>verification</td>
<td>partial differential equation</td>
<td>Hamilton-Jacobi equation solutions[31]</td>
</tr>
<tr>
<td>MATISSE[32]a</td>
<td>verification</td>
<td>transition system</td>
<td>bisimulation</td>
</tr>
<tr>
<td>MultiParametric ToolBox[33]a</td>
<td>verification</td>
<td>piecewise affine systems</td>
<td>linear/quadratic programming solver</td>
</tr>
<tr>
<td>PHAVer[13]</td>
<td>verification</td>
<td>linear I/O hybrid automata</td>
<td>on-the-fly over-approximation</td>
</tr>
<tr>
<td>Ptolemy II[17]</td>
<td>modeling, simulating</td>
<td>embedded system</td>
<td>non-hybrid system verifier</td>
</tr>
<tr>
<td>SHIFT[34]</td>
<td>modelling, translation</td>
<td>SHIFT language</td>
<td>none</td>
</tr>
<tr>
<td>SpaceEx[16]</td>
<td>verification</td>
<td>hybrid automata</td>
<td>time-step flowpipe computation</td>
</tr>
<tr>
<td>SteP[35]</td>
<td>verification</td>
<td>real-time system</td>
<td>invariant generationb</td>
</tr>
</tbody>
</table>

*a* Requiring Matlab  
*b* Theorem proving
<table>
<thead>
<tr>
<th>Name</th>
<th>Year (Update)</th>
<th>Tool Support</th>
<th>Execution Environment</th>
<th>Functions (M/S/A/V/Tr)</th>
<th>Verifiability</th>
<th>Input Front-End</th>
<th>Verification Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARON</td>
<td>2001</td>
<td>Yes</td>
<td>JAVA</td>
<td>M / S</td>
<td>N/A</td>
<td>Automata</td>
<td>N/A</td>
</tr>
<tr>
<td>CheckMate</td>
<td></td>
<td>No</td>
<td>MATLAB</td>
<td>V</td>
<td>MATLAB</td>
<td>MATLAB</td>
<td>Approximate quotient transition systems</td>
</tr>
<tr>
<td>d/dt</td>
<td>2001</td>
<td>Yes</td>
<td>Linux</td>
<td>M / S</td>
<td></td>
<td>d/dt input language</td>
<td>Forward reachability analysis</td>
</tr>
<tr>
<td>Ellipsoidal Toolbox</td>
<td>2006</td>
<td>Yes</td>
<td>MATLAB</td>
<td>V</td>
<td>MATLAB</td>
<td>MATLAB</td>
<td>Forward and backward reachability analysis</td>
</tr>
<tr>
<td>GBT</td>
<td>2004</td>
<td>Yes (Commercial)</td>
<td>MATLAB</td>
<td>A</td>
<td>N/A</td>
<td>MATLAB</td>
<td>Convex hull</td>
</tr>
<tr>
<td>HSIF</td>
<td>2002</td>
<td>Yes</td>
<td>Windows</td>
<td>M / S</td>
<td>N/A</td>
<td>GME model</td>
<td>N/A</td>
</tr>
<tr>
<td>HSolver</td>
<td>2005</td>
<td>Yes</td>
<td>Linux</td>
<td>V</td>
<td>Manual</td>
<td>Input program</td>
<td>Theorem prover (Rsolver)</td>
</tr>
<tr>
<td>HyTech</td>
<td>2000</td>
<td>Yes</td>
<td>Linux</td>
<td>V</td>
<td>Automatic</td>
<td>Linear hybrid automata</td>
<td>Polyhedral model checking</td>
</tr>
<tr>
<td>HyVisual</td>
<td>2000 (2005)</td>
<td>Yes</td>
<td>JAVA</td>
<td>M / S</td>
<td>N/A</td>
<td>Ptolemy plug-in</td>
<td>N/A</td>
</tr>
<tr>
<td>Hybrid ToolBox</td>
<td>2004 (2011)</td>
<td>Yes</td>
<td>MATLAB</td>
<td>M / S / V</td>
<td>MATLAB</td>
<td>HYSDEL language, MATLAB</td>
<td>LP/QP Solver</td>
</tr>
<tr>
<td>HYSDEL</td>
<td>2002 (2011)</td>
<td>Yes</td>
<td>Windows, Linux, Solaris</td>
<td>Tr</td>
<td>N/A</td>
<td>HYSDEL language</td>
<td>N/A</td>
</tr>
<tr>
<td>Level Set Toolbox</td>
<td>2004 (2011)</td>
<td>Yes</td>
<td>MATLAB</td>
<td>S / V</td>
<td>MATLAB</td>
<td>MATLAB</td>
<td>Set of Algorithms</td>
</tr>
<tr>
<td>MATISSE</td>
<td>2005</td>
<td>Yes</td>
<td>MATLAB</td>
<td>V</td>
<td>MATLAB</td>
<td>MATLAB</td>
<td>Bi-simulation, reachable analysis</td>
</tr>
<tr>
<td>MultiParametric Toolbox</td>
<td>2004 (2006)</td>
<td>Yes</td>
<td>MATLAB</td>
<td>M / A / V</td>
<td>MATLAB</td>
<td>MATLAB</td>
<td>Forward and backward reachability analysis</td>
</tr>
<tr>
<td>PHAVer</td>
<td>2004 (2007)</td>
<td>Yes</td>
<td>Windows, Linux, Mac</td>
<td>V</td>
<td>Automatic</td>
<td>Linear hybrid automata</td>
<td>Forward and backward reachability analysis</td>
</tr>
<tr>
<td>SHIFT</td>
<td>1999</td>
<td>Yes</td>
<td>Linux</td>
<td>M / Tr</td>
<td>N/A</td>
<td>Shift language</td>
<td>N/A</td>
</tr>
<tr>
<td>SpaceEx</td>
<td>2010 (2011)</td>
<td>Yes</td>
<td>Linux</td>
<td>V</td>
<td>Automatic</td>
<td>SX language</td>
<td>LeGuernic-Girard Algorithm</td>
</tr>
</tbody>
</table>
ECML

- ETRI CPS Modeling Language
  - Proposed by ETRI (Electronics and Telecommunication Research Institute) in Korea, 2011
  - Supporting ECML Modeling & Simulation
    - EcoPOD
    - EcoSIM
  - Refers to CHARON
  - Extends DEV&DESS formalism
  - Includes several syntactic sugar
    - 3 types of I/O: Discrete / Continuous / Event
    - Easy to model discrete systems as well as continuous systems
      - Allows to use ‘phases’ in addition to states $S = S^C \times S^D$
      - Allows to produce outputs by discrete transitions in addition to continuous/internal transitions
    - Not allow hierarchical state modeling as CHARON and Statecharts
EcoPOD

ECML CMD (Coupled Model Diagram)
ECML BMD (Basic Model Diagram)
HyTech

- A basic verification tool for hybrid systems
  - Model checker
    - Safety verification, Parametric analysis
    - Simulation
  - Input-front-end: linear hybrid automata
  - No the concept of I/O variables
  - No GUI
  - No graphical editor for input programs

- We chose HyTech since it is the most fundamental model checker for hybrid automata.
  - Planning to use PHAVer and SpaceEX as well as HyTech
Formal Verification of ECML using HyTech

ETRI EcoPOD

ECML Model → ECMLtoHyTech*

LHA Model → HyTech Analyzer*

HyTech

LHA: Linear Hybrid Automata
ECMLtoHyTech

• A mechanical translator from ECML to LHA
  – Defined translation rules semi-formally
  – Resolved semantic gap between ECML and LHA of HyTech
    • Uses I/O automaton additionally
    • Uses invariant conditions of LHA to enforce state transition
Linear Hybrid Automata

(I/O Automaton)
Translation Rules

Discrete input automaton

Translation of Transitions

Continuous input automaton
Use synchronized labels to model immediate coming of discrete inputs
Use of invariant to guarantee immediate transitions
HyTech Analyzer

• A visual assistant of HyTech
  – Eclipse plug-in
  – Read LHA, execute HyTech, and visualize verification results

  – Supporting
    • RegionTableViewer
    • RegionAnalyzer
    • TraceTableViewer
    • TraceChart
HyTech Outputs

Region

Location: closing.active
  on_off - 1 & contents - 9 & e = 0 & switch - 1 & 3limit = 1000
  on_off = 2 & contents = 9 & e = 0 & switch = 1 & 3limit = 1000
  on_off - 1 & 3limit = 1000 & contents - e + 9 & switch - 1 & contents <= 10 & contents >= 9
  on_off = 2 & 3limit = 1000 & contents = e + 9 & switch = 1 & contents <= 10 & contents >= 9

Trace

======== Generating trace to specified target region ========
Time: 0.000000
Location: closed.idle
  on_off = 0 & contents = 0 & barrel = 0 & switch = 0 & 3limit = 1000 & e = 0
--------------
VIA 332.333344 time units
--------------
Time: 332.333344
Location: closed.idle
  on_off = 0 & contents = 0 & barrel = 0 & switch = 0 & 3limit = 1000 & 3e = 997
Console result
Conclusion and Future Work

- We have been trying to verify ECML models using HyTech
  - Have developed
    - Translation rules from ECML to linear hybrid automata
    - A mechanical translator – ECMLtoHyTech
    - A visual assistant of HyTech – HyTech Analyzer
  - Also found problems
    - Semantic gap between ECML and LHA
    - Limitation of the HyTech verification
    - Restriction on modeling by linear hybrid automata

- We are now trying SpaceEx.