Toward an Equation-Oriented Framework for Diagnosis of Complex Systems

Gregory Provan and Alexander Feldman

Apr 19, 2013, Nottingham
Overview

• Motivation
• Related Work
  – LNG vs. Modelica
• Lydia-NG Overview
• Examples
  – Circuits
  – Thermo-Fluid Systems
• Summary
Motivation

• How can Modelica libraries be used for Model-Based Diagnosis (MBD)?

• Significant value in Modelica libraries
  – Can this be leveraged for MBD?

• Must understand difference between
  – Simulation (via Modelica)
  – MBD
Contributions

• Equation-based framework for Model-Based Diagnosis
  – Generalisation of Modelica (inference)
  – Uses multiple simulation tools, as well as diagnosis inference tools

• Application to several systems
  – Circuits, thermo-fluid systems
• Assume component-based framework
• Each component has operating mode
  – Mode defines a set of dynamical equations
  – Example
    • \( y = f(x, h) \)
    • \( y \): output; \( x \): state vector; \( h \): mode
Example: Boolean XOR

```c
system xor2(bool o, i1, i2) {
    bool h; // mode variable
    attribute health(h) = h; // like annotations

    h => (o = (i1 != i2));
    !h => (o = !(i1 != i2));
}
```
Simulation

• Given inputs and a health state, compute outputs

\[ y = f(x) \]

\[ x: \text{input}, \quad y: \text{output}, \quad f: \text{system function}, \quad f_i: \text{component functions} \]

• Simulate: solve problem \( y = f(x) \)
Fault Diagnosis

- Identify component(s) that are **root cause of failure** (error)

  ![Diagram](image)

  - $x, y$: observation vectors
  - $f$: system function, $f_i$: component functions
  - $h$: system health state vector, $h_i$: component health variables

- Diagnose failure: solve inverse problem $\underline{h} = f^{-1}(x, y)$

- Diagnosis: $\underline{h}_2 = \text{fault state}$, or $\underline{h}_4$ and $\underline{h}_5 = \text{fault state}$
Simulation vs. Diagnosis

• Simulation
  – solving a system of equations for some output variables

• Diagnosis
  – solving a system of equations for some set of health variables,

  – However:
    • Diagnostic systems typically under-constrained due to ignorance of abnormal behavior, etc.
    • Simulation systems often constructed to have one solution,
      – in diagnosis we want to compute multiple solutions (hypotheses/diagnostic candidates)
Overview

• Motivation

• Related Work
  – Lydia-NG vs. Modelica

• Lydia-NG Overview

• Examples
  – Circuits
  – Thermo-Fluid Systems

• Summary
Related Work

• Rodelica
  – Atemporal, interval-based approach
• Modelica
• Bond graphs
LYDIA-NG and MODELICA

• **LYDIA-NG** is very similar to Modelica
  – solves Modelica models for some parameters
  – we plan a Modelica translation tool for easy modeling
    • problem is some Modelica component libraries include non-declarative model entities
**LYDIA-NG vs. MODELICA**

**LYDIA-NG**
- Braces (C/C++/Verilog)
- Not object-oriented (may become in the future)
- Strongly-typed
- Quantified (existential, universal), static expansion

**MODELICA**
- begin/end (Pascal/VHDL)
- Object-oriented
- Strongly-typed
- Less static
LYDIA-NG vs. MODELICA (cont.)

- **LYDIA-NG**
  - no connectors (only variables)
  - no flow variables (a.k.a. write your own “global” equations)
  - fairly complex data types: structures, arrays, array of structures, etc.
  - special treatment of Boolean systems (legacy)

- **MODELICA**
  - uses connectors
  - flow variables (equation sugar)
Compilation and Inference

• Lydia-NG
  – Compilation not a key aspect
    • Output C code can be used by any inference system

• Modelica
  – Compilation key aspect for efficiency
  – Compilation can lead to incompatibility among Modelica inference systems
system HeatingCoil(PneumaticPort pneumaticCold, pneumaticHot, HydraulicPort hydraulicCold, hydraulicHot)
{
    float c = 4180.0; // water specific heat

    float coldSideCapacitanceRate = pneumaticCold.mflow * c;
    float hotSideCapacitanceRate = hydraulicCold.mflow * c;

    float eff = 0.6;

    float minCapacitanceRate = min(coldSideCapacitanceRate, hotSideCapacitanceRate);
    float maxCapacitanceRate = max(coldSideCapacitanceRate, hotSideCapacitanceRate);

    float heatRate = eff * minCapacitanceRate * (hydraulicHot.T - pneumaticCold.T);

    hydraulicHot.T = hydraulicCold.T - heatRate / (hydraulicCold.mflow * c);
    pneumaticHot.T = pneumaticCold.T + heatRate / (pneumaticCold.mflow * c);
}
• **LYDIA-NG is a diagnostic framework**
  – Use cases:
    • modeling of diagnostic systems (also an IDE)
    • running of diagnostic scenarios in batch mode (computation of diagnostic metrics)
    • embedding in a SCADA system, e.g., a BMS

• **Our view on the development of LYDIA-NG**
  – collection of tools (simulators, translators, etc.)
  – simple interfaces (our users are not assumed to be expert MBD users)
  – higher coding/documentation/knowledge dissemination standards than typical projects
LYDIA-NG Overview

• Core libraries
  – simulation
    • SPICE
    • symbolic
    • ODEs/DAEs
    • Boolean circuits (old LYDIA heritage)
  – diagnosis
    • forward reasoning (simulation for various health/fault states)
    • backward reasoning (residual analysis)
  – disambiguation
    • entropy-based selection of tests
    • virtual sensors (by-product)
LYDIA-NG Approach to Diagnosis

- The main idea is to run multiple simulations simultaneously (each simulation reflects different health state)
- Choose those simulations that minimize some residual function
- Report diagnosis as a probability of each component being healthy/faulty
State of LYDIA-NG Development

- **LYDIA-NG** has reached a milestone in diagnosing an electrical system
  - results show that the software will be useful in practice
  - releasing version 1.0 after fixing some bugs, documentation, and testing
  - preview versions of the software continuously made available to EMWiNS team members for purpose of progress-tracking, collaboration, and planning
Validation of Diagnostics

- Validation of diagnostics is in general more difficult than that of simulation
  - simulation accuracy metrics such as mean difference from measured values
  - diagnostic accuracy metrics are interrelated (false positives vs. false negatives, classification errors, etc.)
  - diagnostic metrics are often domain-dependent (e.g., energy)
  - diagnostic world is “less closed”, some metrics can be computed only after, e.g., “repair”
Overview

• Motivation
• Related Work
  – LNG vs. Modelica
• Lydia-NG Overview
• Examples
  – Circuits
  – Thermo-Fluid Systems
• Summary
Use-Cases

• Many Boolean circuits (ISCAS-85)
  – show properties of complex systems, easier to analyze complexity
  – check correctness of some diagnostic algorithms
• Analog electrical circuits (GOCE satellite EPS, NASA’s ADAPT)
• Thermal-fluid systems (UCC’s AHU-9)
Use-Case AHU System
AHU Modeling
LYDIA-NG and AHU-9

 Modeling effort includes the following:
  – Top-level topology (100%)
  – Component fault-modes and user commands (100%)
  – Component equations (0%)

 Model parametrization and calibration
AHU Simulation

- ODE
  - not stiff
  - RK4 will do the job
  - error is function of the step-size
- We need to maintain multiple simulations that can be stopped/continued whenever sensor data arrives
- Some ODEs reduce to algebraic equations – we also saw from GOCE that fault simulations are cheaper in terms of CPU time, memory
Overview

• Motivation
• Related Work
  – LNG vs. Modelica
• Lydia-NG Overview
• Examples
  – Circuits
  – Thermo-Fluid Systems
• Summary
Summary

• Lydia-NG: MBD framework
  – Accepts multiple equation types
  – Generalises Modelica
    • mode-based equations
    • Wider range of inference algorithms
Future Work

• Integrate control
• Extend language to wider range of dynamical systems
• Examine other real-world applications
  – thermal systems
  – Mechanical systems (drive trains)
• DXC-2013
Call for Participation

• Want to play with/test/develop LYDIA-NG?
  – LYDIA-NG is free for academic use/open-source
  – send an email to alex@general-diagnostics.com

• Want to apply LYDIA-NG to your research/write a paper?

• Want to extend LYDIA-NG to solve #@! equations?

4/23/2013
Come to DX-2013

• Come to DX-2013, Oct 1-4, Dan Panorama Hotel, Jerusalem (http://dx-2013.org/)

• DXC-2013
  • synthetic track (ISCAS)
  • electrical system (ADAPT)
  • thermal fluid system (AHU-9)
  • software track
Thank You
Backup
Introduction

• History
  – **LYDIA** – diagnosis of Boolean circuits
    • **SAFARI** – stochastic diagnosis
    • **FRACTAL** – disambiguation of diagnoses (reduce diagnostic uncertainty – entropy based methods)
    • Beyond diagnosis – worst case sensor data – diagnosability (**MIRANDA**)
  – A resistor network can be modeled with Boolean variables but that is very difficult – e.g., 4-bit multiplication tables
  – **LYDIA-NG** – generalization to continuous variables
    • **SAFARI-NG** – greedy stochastic reasoning, but also other candidate generation policies
    • **FRACTAL-NG** – disambiguation

• The insight that allows generalization of Lydia to Lydia-NG is that simulation is a key-component in model-based diagnosis
  – What is the simulation step in the MBD circuit problem shown in the next slide?
• Entropy-based methods for computing uncertainty of a component:

\[ U \sum_{x \in C^*} - \Pr \sum_{x \in C} \log \Pr \sum_{x \in C} \Pr = x \]

• Per system:

\[ \bar{U} = \frac{1}{|COMPS|} \sum_{C \in COMPS} U \sum_{x \in C} \Pr \]

![Diagram of electrical circuit](image-url)
- AHU systems are typically sensor-lean to reduce cost
- AHU models are imprecise due to:
  - fine-grained CFD modeling is too complex
  - the AHU model cannot include a model of the local weather
- Sensors may drift/fail
- As a result diagnosis may be inaccurate
- We propose an algorithm that can increase the diagnostic accuracy by “playing” with the system
  - for example the system can reconfigure the mixing box to confirm/disprove a hypothesis about a failing heating coil