Model-Based Diagnostics

Matthew Daigle (NASA Ames Research Center)
Indranil Roychoudhury (SGT Inc., NASA Ames Research Center)

Diagnostics & Prognostics Group, Intelligent Systems Division
NASA Ames Research Center

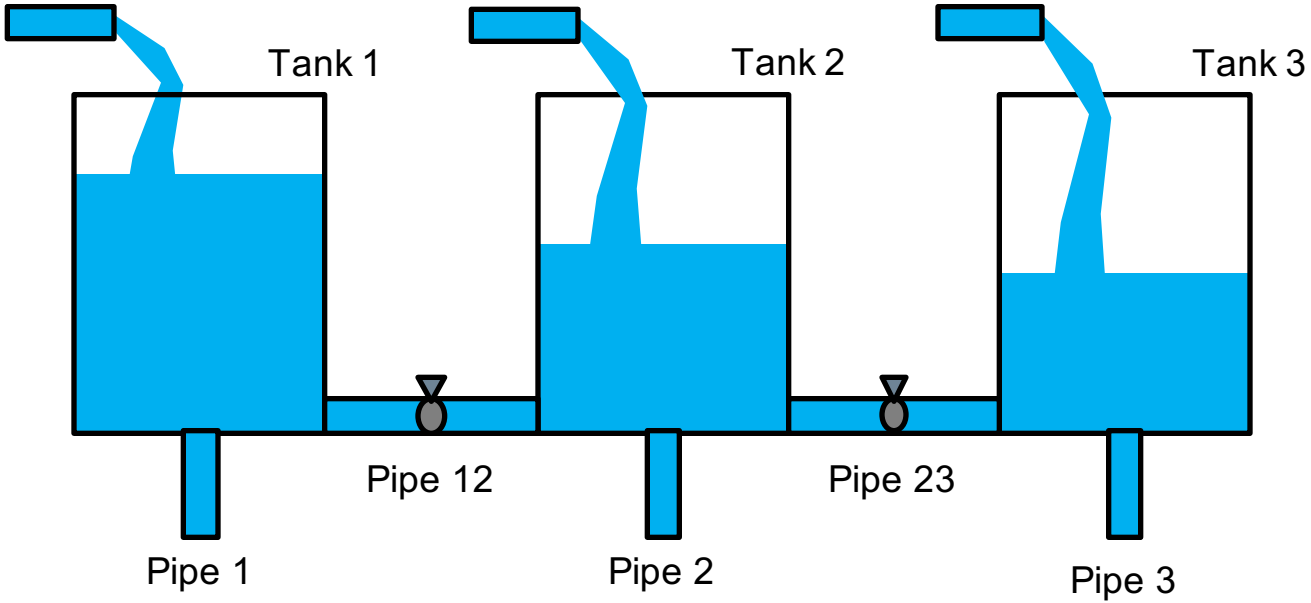
Objectives of This Tutorial

- What is meant by model-based diagnostics and why it is a preferred approach
- Explain the fundamentals of model-based diagnosis
- Overview different kinds of models and algorithms
- What are the constituent problems, and how do we solve them

Scope of This Tutorial

- The focus here is on defining the model-based diagnosis problem in a general way, and describing the types of models and algorithms used
  - Defining the constituent problems
  - Discrete-event systems, continuous systems, and hybrid systems models
  - Algorithms for different types of models
- For other material, see diagnosis tutorials from previous PHM conferences
  - Machine learning methods
  - Consistency-based diagnosis
  - Sensor selection
  - Test generation
  - Applications
  - Other perspectives on diagnostics

Running Example: Multi-Tank System



Outline

- Preliminaries
- Fundamentals of Model-based Diagnosis
- Discrete-event Systems Diagnosis
- Continuous Systems Diagnosis
- Hybrid Systems Diagnosis
- Distributed Diagnosis
- Conclusions

Preliminaries

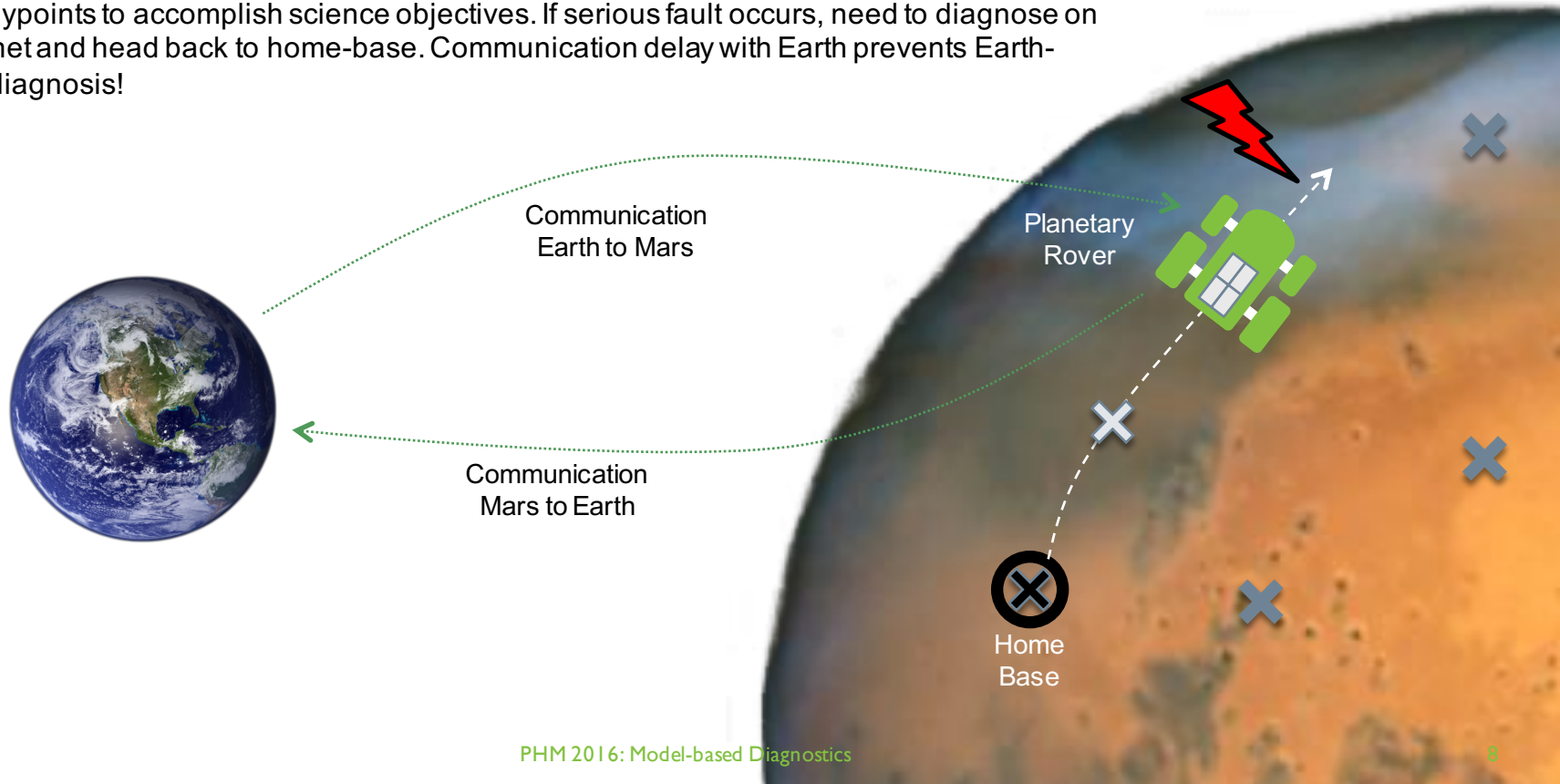
1. What is diagnostics?
2. What is model-based diagnostics?
3. Why model-based diagnostics?
4. What are the constituent problems?

What is Diagnostics?

- Diagnosis = determining the nature and cause of something
- In a health management context, this "something" is a fault
  - Fault = an unexpected change in the dynamics of a system
  - Fault ≠ Failure!
  - Failure = a condition of the system in which it does not meet functional specifications
  - Faults can grow and lead to failure, or a fault may be significant enough in magnitude, or a severe enough change in configuration, such that the system will have failed (due to the occurrence of the fault)

Why Diagnostics?

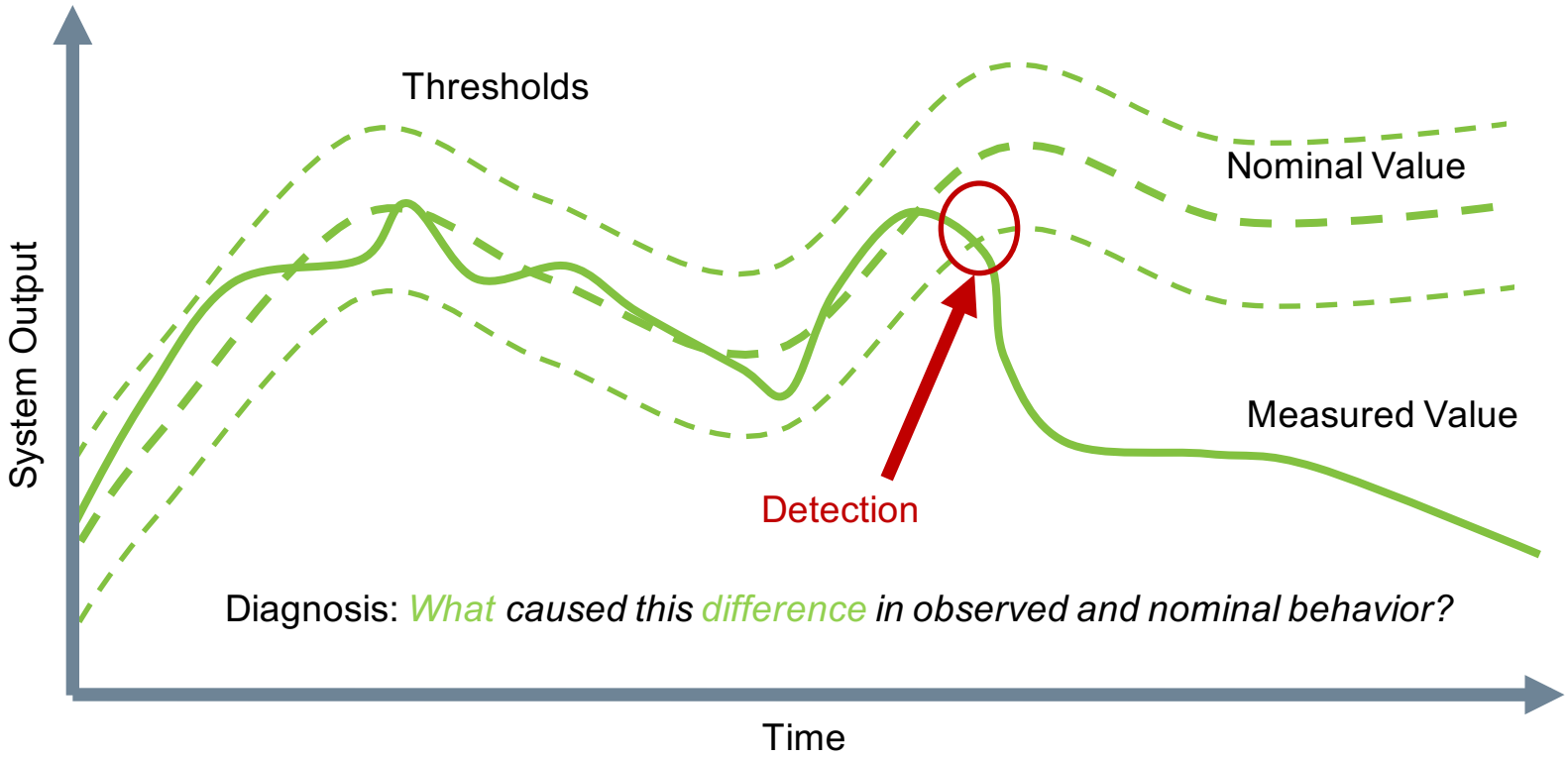
Example: Rover Mission
Visit waypoints to accomplish science objectives. If serious fault occurs, need to diagnose on the planet and head back to home-base. Communication delay with Earth prevents Earth-based diagnosis!



Why Diagnostics?

- Diagnostics informs decision-making
- Which components to repair/replace
- Inform fault mitigation
- Inform fault recovery
- Inform functional reallocation
- Diagnostics informs prognostics
  - What is (are) the dominant aging/degradation mode(s)
  - Can we complete operation(s)/mission in presence of fault?

The Basic Idea



Nominal vs Faulty Behavior

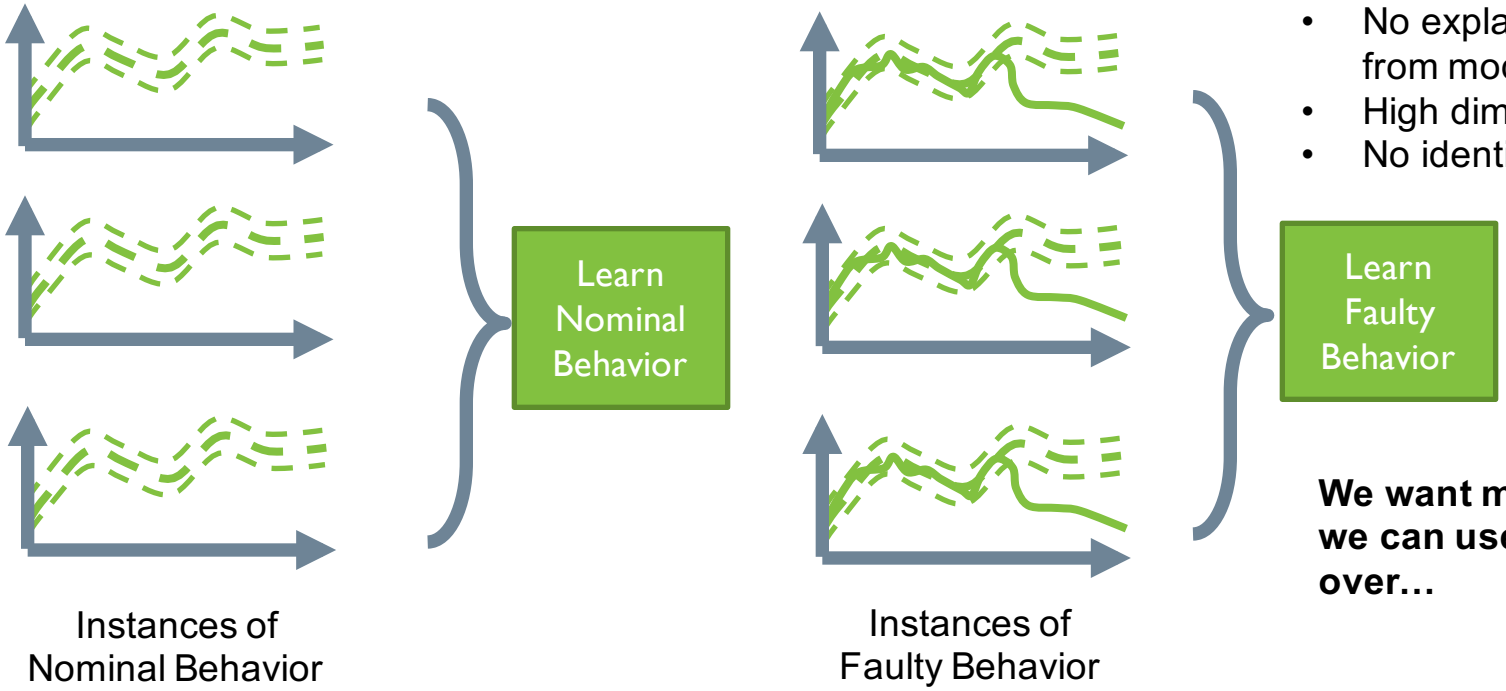
- What is "nominal" behavior?
  - Nominal is defined w/r/t some knowledge about the system, a reference behavior
  - Comes from expert knowledge, known operating limits, physics model, machine learning approaches, etc.
- In *model-based* diagnostics, the reference comes from a model that explicitly describes the nominal behavior
  - Models can be static or dynamic
  - Models can be used for simulation

Why Model-based Diagnostics?

Machine Learning Approach:

Detection: Classify between nominal and non-nominal behavior

Isolation: Classify between different classes of faulty behavior



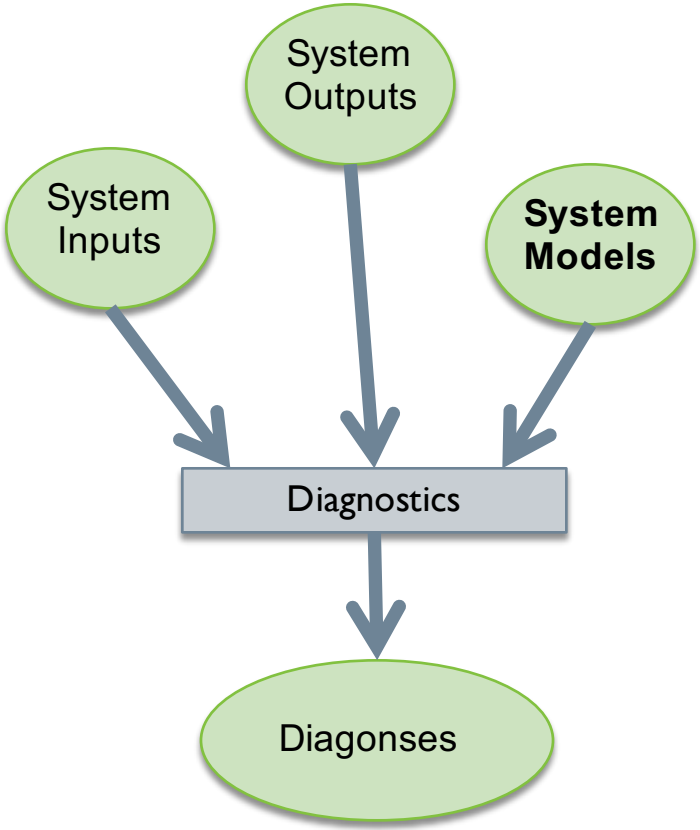
Problems:

- Lack of faulty data instances
- No explanatory power from models
- High dimensionality
- No identification

We want models that we can use to *reason* over…

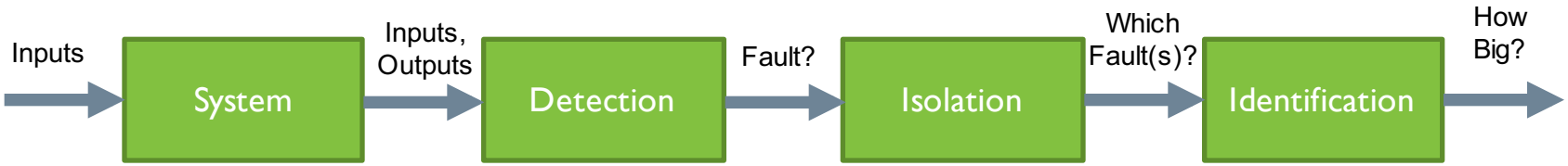
Why Model-based Diagnostics?

- Models have explanatory power
  - Causal reasoning
  - Explicit representation of faults
- Develop general model-based algorithms
  - Models used for diagnosis are *inputs*
  - Algorithms do not change for a new system, only the model changes



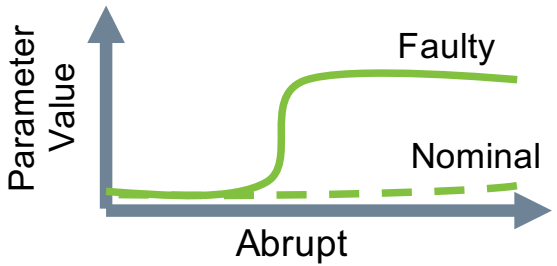
Constituent Problems

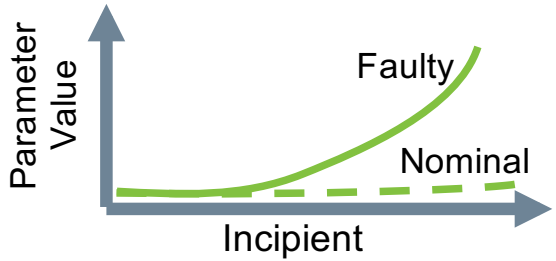
- Fault detection = determination of whether the system is not operating nominally
- Fault isolation = determination of the root cause(s) of the unexpected system behavior
- Fault identification = determination of the magnitude of the fault (if applicable)



Characterizing Faults

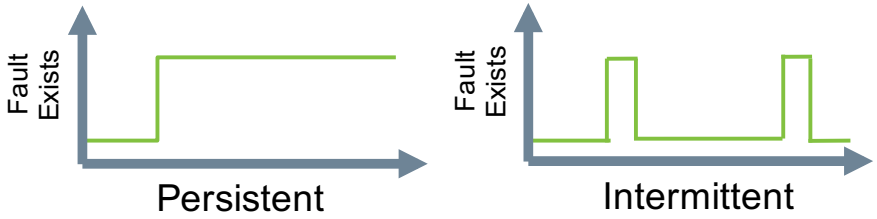
- Abrupt: Change in parameter value faster than the sampling frequency
- Incipient: Change in parameter value slower than the sampling frequency
  - Can be linear, exponential, or arbitrary degradation
  - Prognostics usually pertains to incipient faults
- Abrupt faults can be easier/faster to detect compared to incipient faults
- Dynamics of fault is different from dynamics of measurements/observations
  - E.g., abrupt fault can present incipient change in measurements

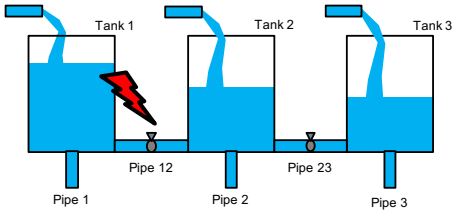




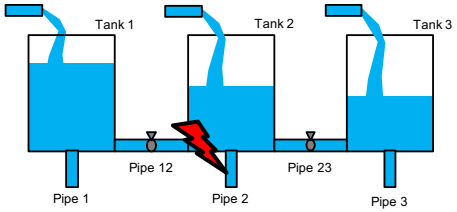
Characterizing Faults

- Persistent vs Intermittent
  - Persistent: Once manifested, the fault persists
  - Intermittent: Fault manifests intermittently
- Discrete vs Parametric
  - Discrete faults involve undesired change in system or model structure, e.g., valve on Pipe 12 stuck closed
  - Parametric faults involve undesired change in system or model parameter, e.g., Pipe 2 is clogged





Discrete



Parametric

Models

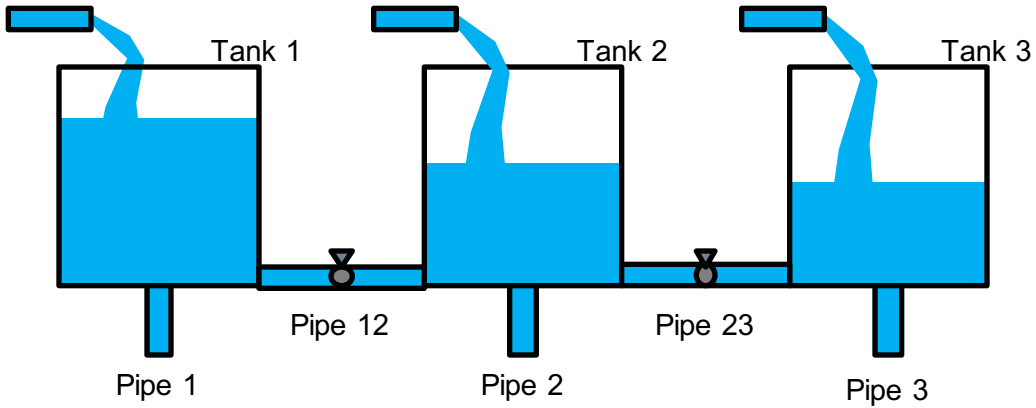
- Discrete Event System Model
  - System behavior abstracted into discrete states
  - Dynamics represented through events that define state transitions
  - Example models: Finite State Machines, Petri Nets, etc.
- Continuous System Model
  - Tries to capture continuous time evolution of system behavior
  - Using computers, these systems are modeled and simulated using differential or difference equations, and in discrete time
  - Example models: Ordinary Differential Equations, Partial Differential Equations, Bond Graphs, Bayes Nets, etc.
- Hybrid System Model
  - Combines both continuous time and discrete dynamics
  - Has discrete states, with continuous behavior defined for each discrete state
  - Example models: Hybrid Automata, Hybrid Bond Graphs, etc.

Fundamentals of Model-Based Diagnosis

1. Logical foundations of diagnosis
2. How to perform diagnostic reasoning?
3. Practical considerations

Specifying Behavior

- First-order logic formulation
  - SD = system description, set of first-order sentences
  - COMPS = components, set of constants



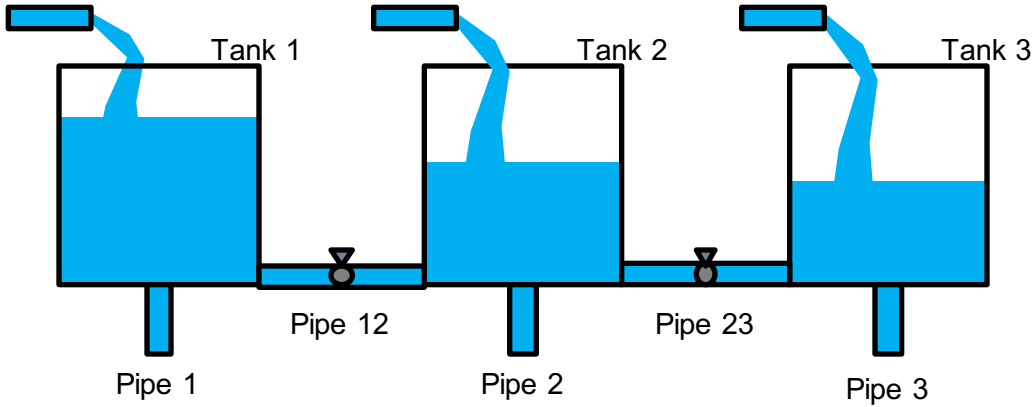
(Assume steady-state behavior)

SD
- In(Pipe12) = Out(Pipe12)
- In(Tank1) = Out(Pipe1) + Out(Pipe12)
- etc…

COMPS
- Tank1, Tank2, Tank3
- Pipe1, Pipe2, Pipe3
- Pipe12, Pipe23

Observations

- Diagnosis requires observations from the system
  - Without observations, no way to determine if something is wrong
  - OBS = observations, set of first-order sentences

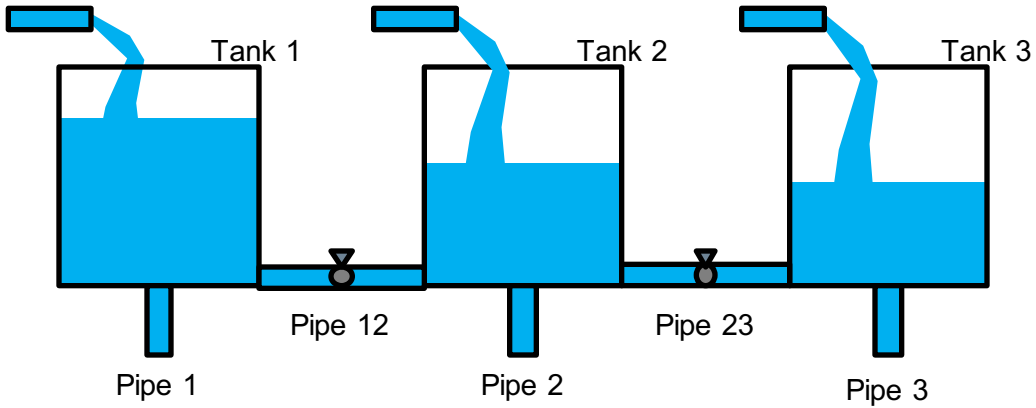


(Assume steady-state behavior)

OBS
- In(Tank1) = 5
- Out(Pipe1) = 10
- Out(Pipe12) = 6
- …

AB(.) Predicate

- SD describes *nominal* system behavior
- AB(C) means component C is *abnormal*, i.e., not nominal
- Extend SD with AB predicates



(Assume steady-state behavior)

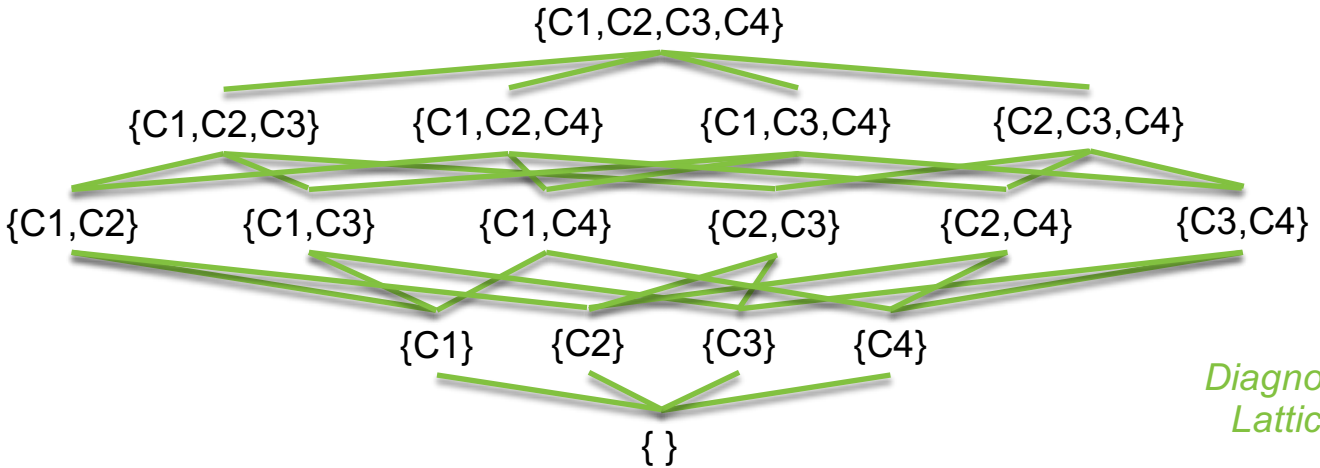
SD

- ~AB(Pipe12) → In(Pipe12) = Out(Pipe12)
- ~AB(Tank1) and ~AB(Pipe1) and ~AB(Pipe12) → In(Tank1) = Out(Pipe1) + Out(Pipe12)
- etc…

Diagnosis

- If operating nominally, then
  - SD and OBS and {~AB(C1), ~AB(C2), …} will be consistent
- If not operating nominally, a contradiction is derived
  - This is how fault detection is performed
- Diagnosis = set of components D such that:
  - SD and OBS and {AB(c) | c in D} and {~AB(c) | c in COMPS-D} is consistent
  - Isolation is performed by finding D
  - Identification is not applicable

Minimality:
A diagnosis is minimal if there is no superset that is also a diagnosis. Working with minimal diagnoses is more efficient and captures the principle of parsimony.

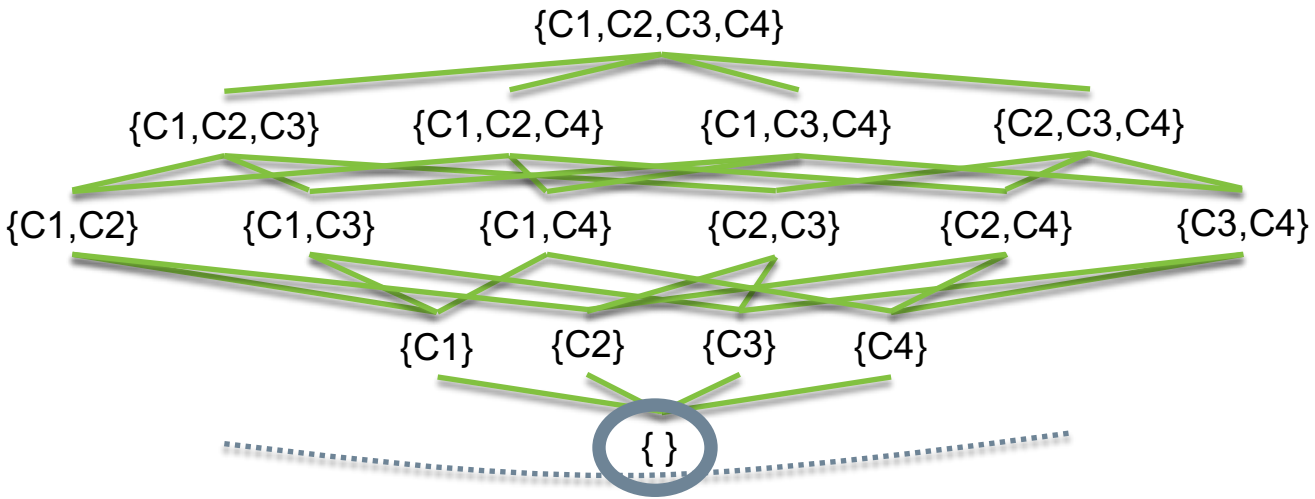


Diagnosis Lattice

Reasoning with Conflicts

- Conflict = set of components that cannot all be faulty
- Given an observation, generate a conflict
  - Update diagnosis based on conflict
  - Repeat with new observations

Example

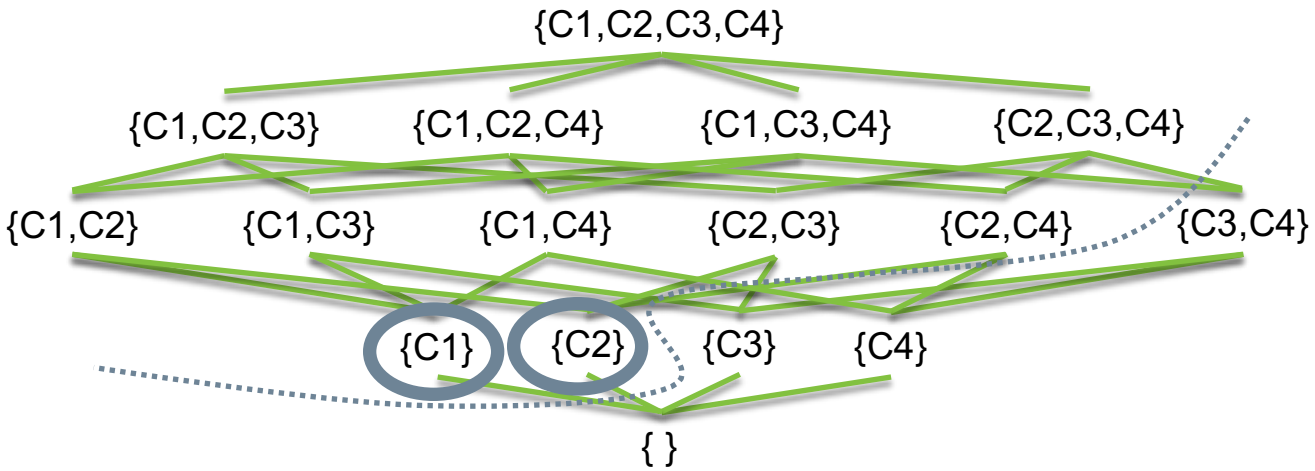


Reasoning with Conflicts

- Conflict = set of components that cannot all be faulty
- Given an observation, generate a conflict
  - Update diagnosis based on conflict
  - Repeat with new observations

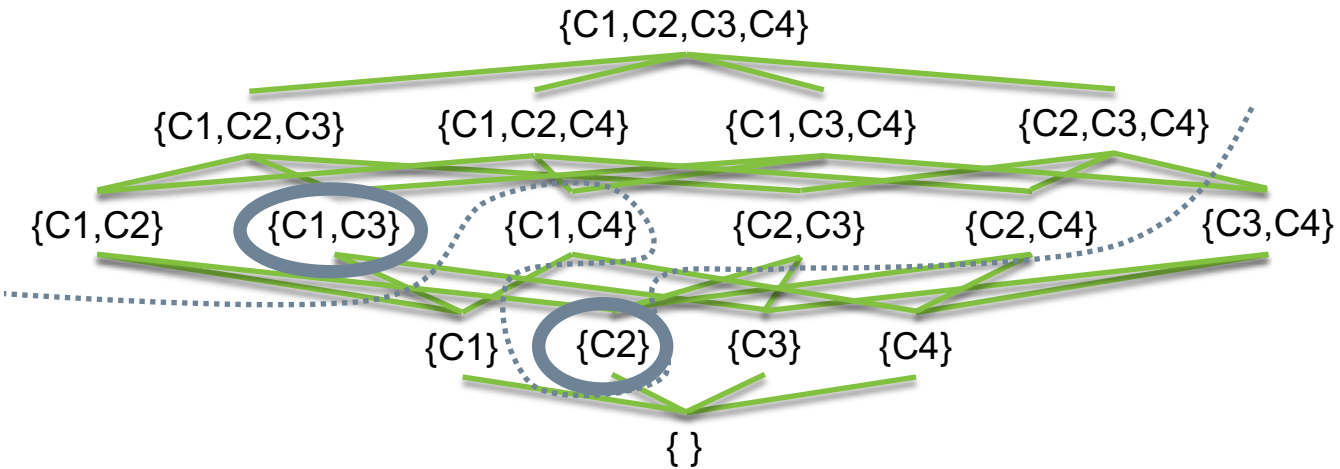
Example

1. Conflict {C1,C2}
   Means C1 or C2 is faulty.
   Previous diagnosis is { }.
   New diagnoses are {C1,C2}.



Reasoning with Conflicts

- Conflict = set of components that cannot all be faulty
- Given an observation, generate a conflict
  - Update diagnosis based on conflict
  - Repeat with new observations

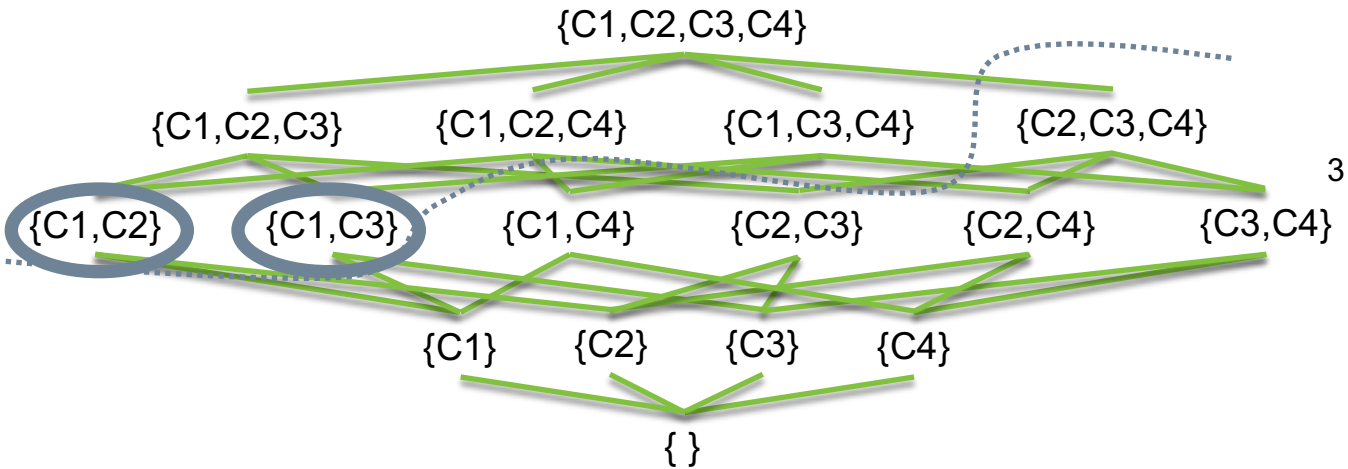


Example

1. Conflict {C1,C2}
   Means C1 or C2 is faulty.
   Previous diagnosis is { }.
   New diagnoses are {C1,C2}.
2. Conflict {C2,C3}
   Means C2 or C3 is faulty.
   New diagnoses are (C1 or C2) and (C2 or C3) = {C2,C1C2,C1C3,C2C3}.
   Minimal diagnoses are {C2,C1C3}.

Reasoning with Conflicts

- Conflict = set of components that cannot all be faulty
- Given an observation, generate a conflict
  - Update diagnosis based on conflict
  - Repeat with new observations



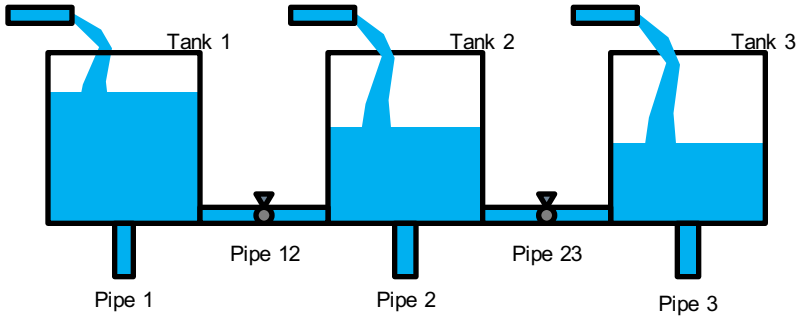
Example

1. Conflict {C1,C2}
   Means C1 or C2 is faulty.
   Previous diagnosis is { }.
   New diagnoses are {C1,C2}.
2. Conflict {C2,C3}
   Means C2 or C3 is faulty.
   New diagnoses are (C1 or C2) and (C2 or C3) = {C2,C1C2,C1C3,C2C3}.
   Minimal diagnoses are {C2,C1C3}.
3. Conflict {C1}
   Means C1 must be faulty.
   New diagnoses are ((C2) or (C1 and C3)) and (C1) = {C1C2,C1C3}.
   Minimal diagnoses are {C1C2,C1C3}.

Example: Steady-State Tank System

COMPS:
Tank1, Tank2, Tank3, Pipe1, Pipe2, Pipe3, Pipe12, Pipe23

SD:
~AB(Tank1) → In(Tank1) = In(Pipe1) + Left(Pipe12)
~AB(Tank2) → In(Tank2) + Right(Pipe12) = In(Pipe2) + Left(Pipe23)
~AB(Tank3) → In(Tank3) + Right(Pipe23) = In(Pipe3)
~AB(Pipe1) → In(Pipe1) = Out(Pipe1)
~AB(Pipe2) → In(Pipe2) = Out(Pipe2)
~AB(Pipe3) → In(Pipe3) = Out(Pipe3)
~AB(Pipe12) → Left(Pipe12) = Right(Pipe23)
~AB(Pipe23) → Left(Pipe12) = Right(Pipe23)



OBS:
Left(Pipe12) = 2
In(Tank1) = 5
In(Pipe1) = 2

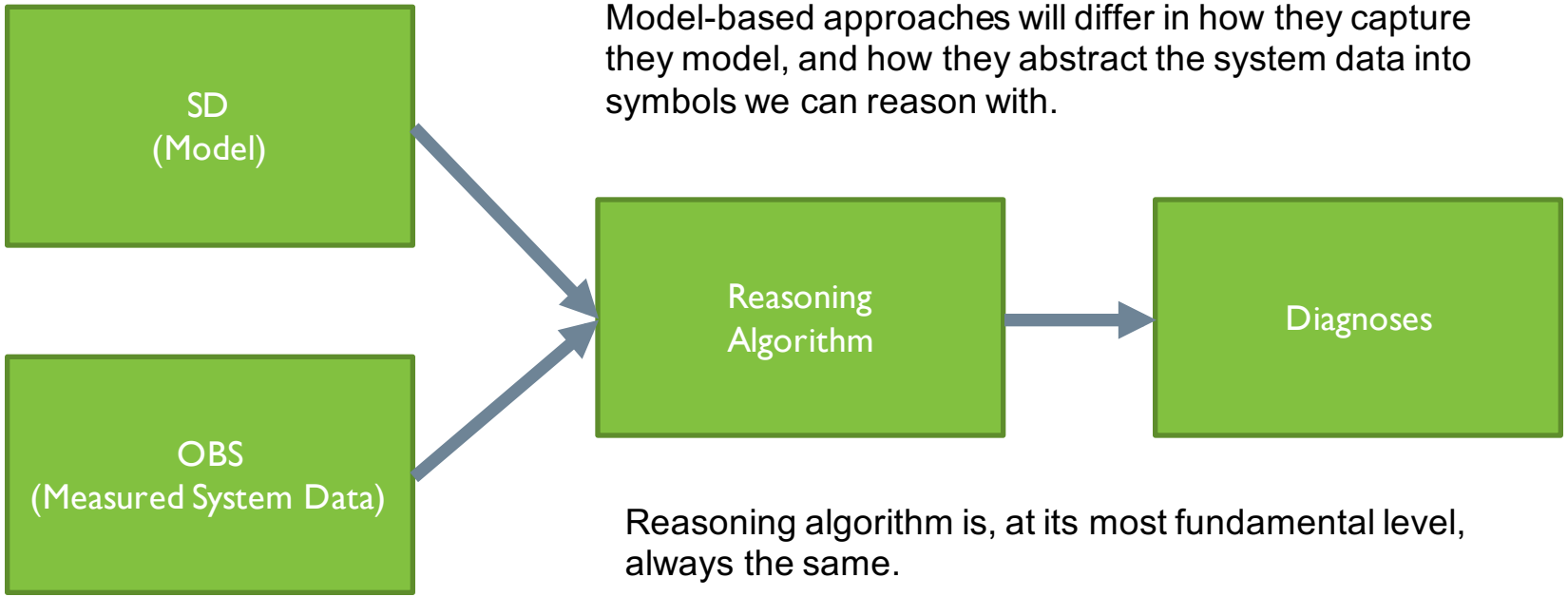
SD and OBS and {~AB(Tank1), ~AB(Tank2),…~AB(Pipe23)} → Contradiction!

Minimal diagnosis is AB(Tank1).

But what is "abnormal" about it?

- Use the concept of behavioral modes
- For each possible fault for each component, specify behavior for that fault mode
- Instead of determining AB(C), determine F1(C) or F2(C), etc.

Summary of Approach



Model-based approaches will differ in how they capture they model, and how they abstract the system data into symbols we can reason with.

Reasoning algorithm is, at its most fundamental level, always the same.

Practical Considerations

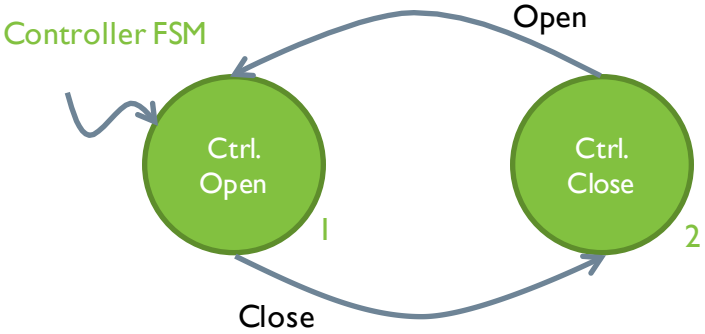
- Such an approach doesn't work well for dynamic systems, and hides many issues
  - What about sensor noise?
  - How to represent dynamic behavior in this framework?
  - How to reason over time?
  - Computational complexity?
- However, the algorithmic approach is sound and forms the basis for most model-based diagnostic reasoning algorithms
  - Describe nominal and faulty behavior
  - Reason over discrepancies between nominal and observed behavior
  - Determine which faults would be consistent with the observations
  - This is always the approach in model-based diagnosis!

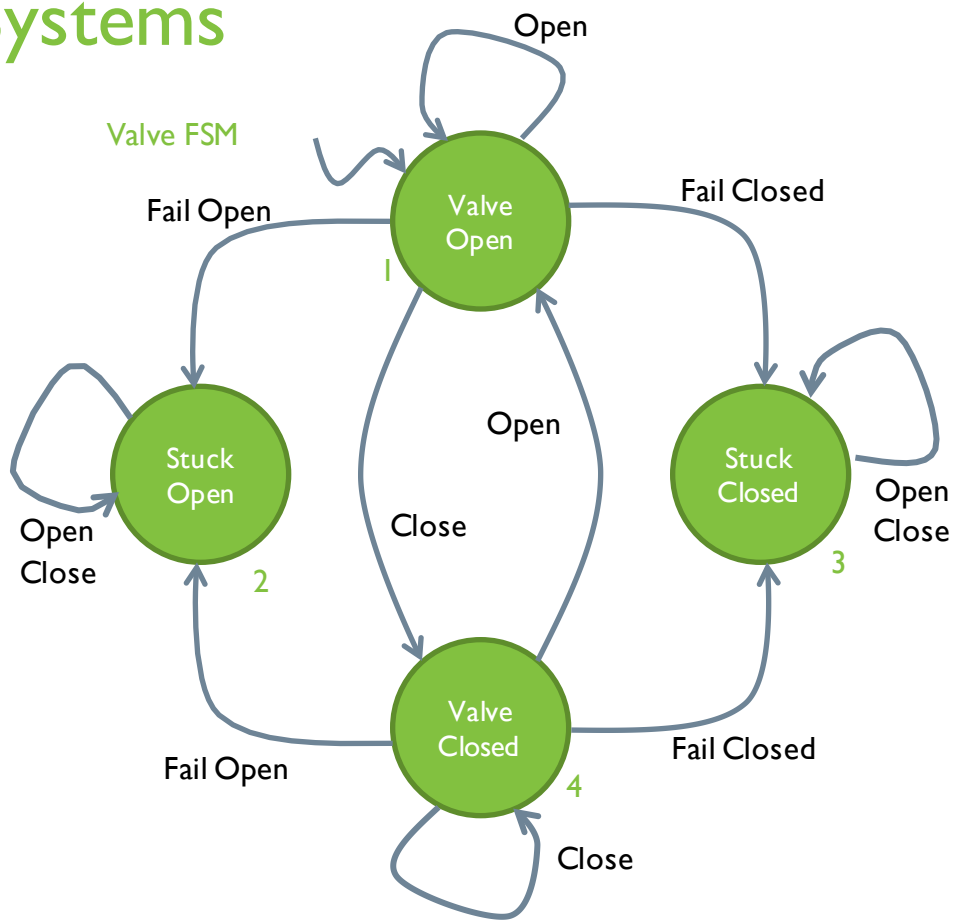
Discrete-Event Systems Diagnosis

1. How do you model discrete event systems?
2. How do you diagnose faults in discrete event systems?
3. What are some practical considerations while diagnosing discrete event systems?

Modeling Discrete-Event Systems

- Finite state machine: $G = (X, S, d, x_0)$
  - X is the set of states
  - S is the set of events
  - d is the transition function
  - x_0 is the initial state
- Faults are modeled as *unobservable events*



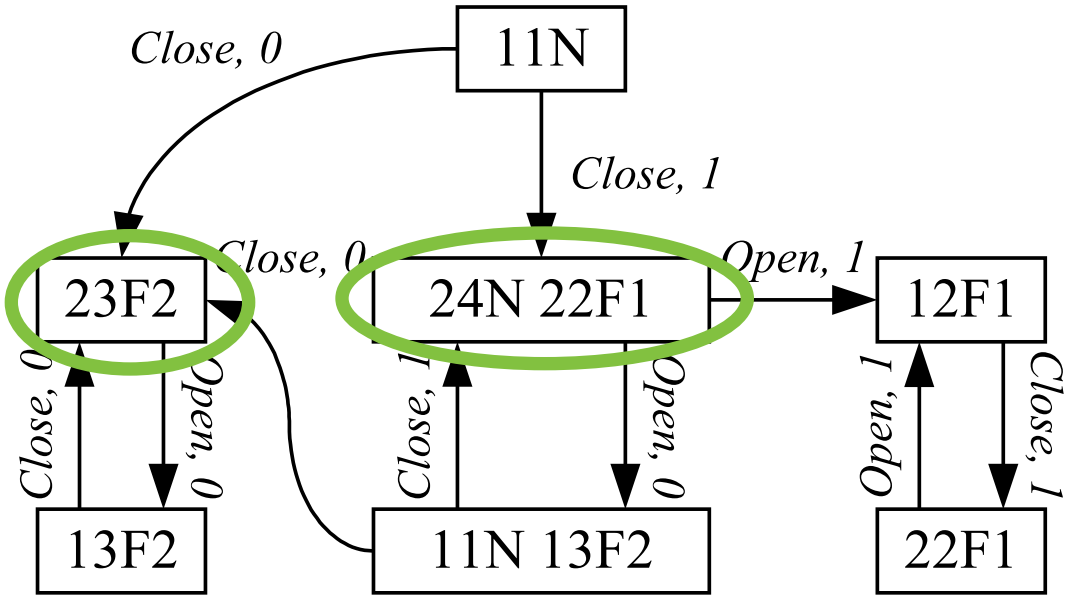


Diagnosis

- Some events are observable, and some are not
- We need to estimate what the possible state is, as that determines whether an unobservable fault event may or may not have occurred
- In the valve example:
  - We have a sensor that reports the position of the valve at a regular interval, 0 for open and 1 for closed
  - Say that we observe the event sequence: Open, 0, 0, 0, Close, 1, 1
    - Is this nominal? Maybe
  - Say that we observe: Open, 0, 0, Close, 0, 0
    - Is this nominal? No
  - How do we capture this reasoning?

Diagnosers

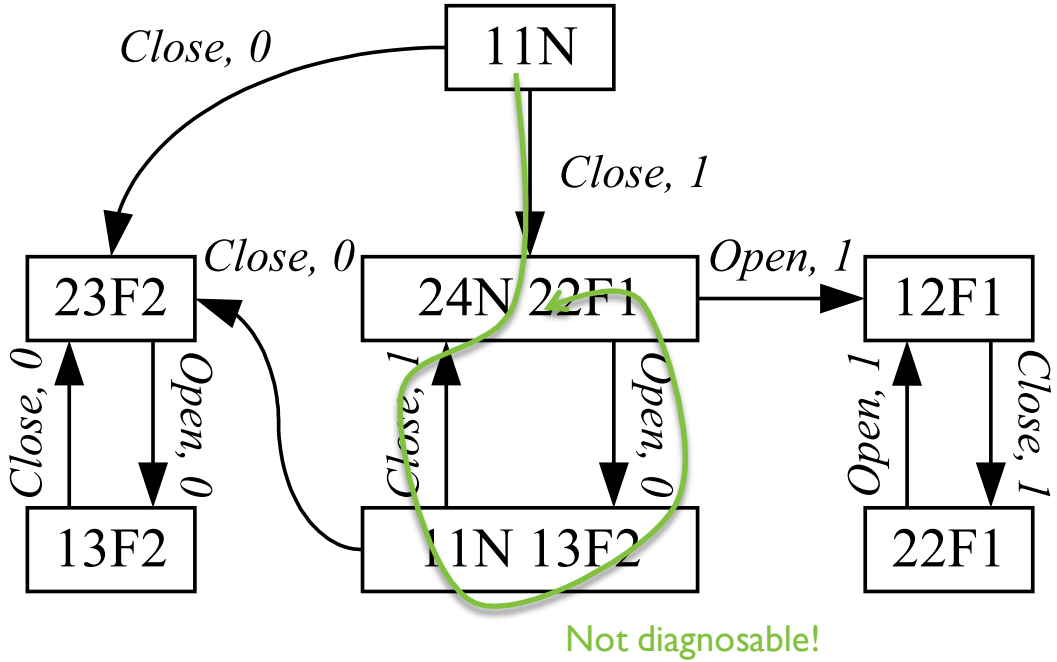
- Build diagnosers, which are labeled FSMs, that track the system state along with observed events
  - Labels are possible states and diagnoses (fault events)
  - There is a procedure to build from given FSMs
- For example
  - Start in initial states of both automata (11N) where system known to be nominal
  - If we command to close and measure as open, it is definitely faulty (fail open has occurred)
  - If we command to close and measure as closed, may or may not be faulty (fail closed might have occurred)



A reasoning algorithm would map events to diagnoses. This is basically "hard-coding" the reasoning algorithm.

Diagnosability

- Want to be able to guarantee that, eventually, we will know if a fault event has occurred
  - A system is diagnosable if, after finite time, we can determine that a fault has occurred from the observed events
- Procedure exists to, given FSMs, determine diagnosability w/r/t different faults



Practical Considerations

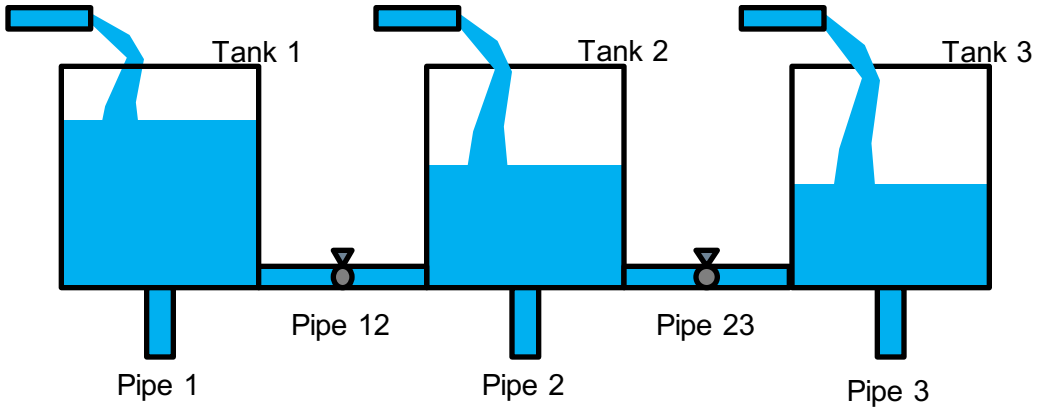
- Following the diagnoser approach
  - A fault is *detected* once we reach a diagnoser state where the label contains only faults
  - A fault is *isolated* once we reach a diagnoser state where the label contains only a specific fault
  - Identification is not applicable here
- Need to abstract system behavior to FSM
  - For continuous-time systems, there is a significant loss of information
  - Still have to deal with sensor noise, and abstract sensor signals to events
  - FSMs can get messy quickly – but the diagnosers are very time-efficient

Continuous Systems Diagnosis

1. How do you model continuous time systems?
2. Fault Detection in Continuous Systems
3. Fault Isolation in Continuous Systems
4. Fault Identification in Continuous Systems

Modeling

- Continuous system behavior captured using "equations"
  - Generally modeled using Ordinary Differential Equations
- Faults are parametric only, i.e., modeled as changes in system parameters
  - E.g., R_1^+ = increase in resistance R_1 of Pipe 1
  - Since continuous system, there are no discrete faults
  - Faults can be abrupt or incipient, and persistent or intermittent



$$dm_1/dt = u_1 - q_1 - q_{12}$$
$$p_1 = m_1/K_1$$
$$m_1 = \int dm_1/dt$$

Tank1

$$q_1 = p_1/R_1$$
$$q_1^* = q_1$$

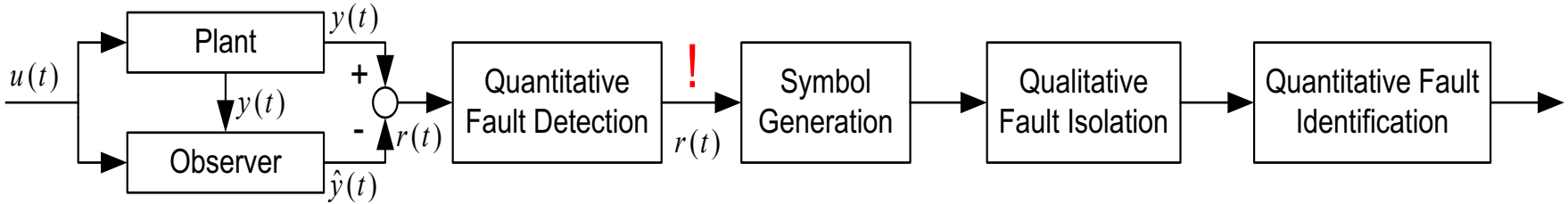
Pipe1

$$q_{12} = (p_1 - p_2)/R_{12}$$

Pipe12

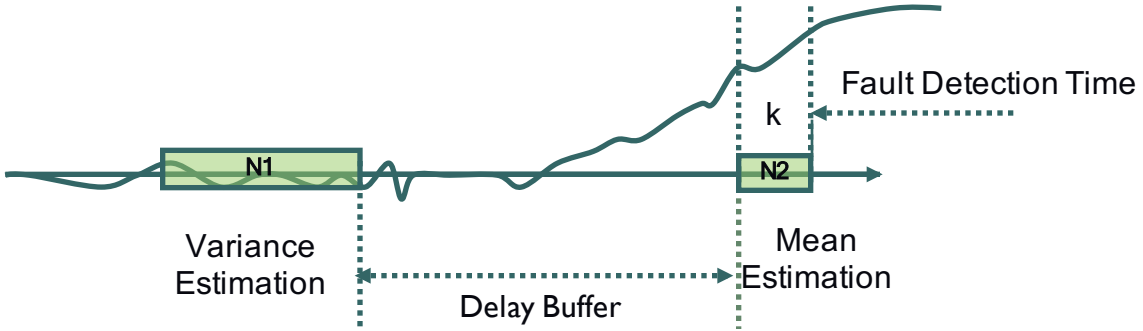
Continuous System Diagnosis: One Approach

- Tackles fault detection, isolation, and identification problems
- Assumes single, persistent parametric faults
  - Can be either abrupt or incipient
- Changes in parameter produces changes in system outputs w/r/t no parameter change
  - Eventually all (causally related) sensors effected
- Need to reason over those changes
  - Have a model of how measurements should deviate given different possible faults
  - Noting the order in which different measurement deviations are observed can also give us clues about the fault
  - Compare observed deviations to expected deviations for each fault candidate to diagnose true fault



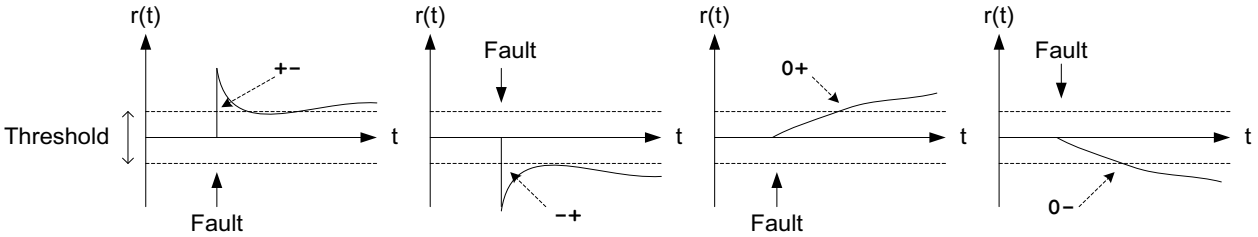
Residual Generation and Fault Detection

- Residual Generation
  - Observer (eg, Kalman filter, unscented Kalman filter, particle filter) based on nominal local submodel computes nominal behavior as a reference
  - Residual computed as measured value minus reference value
- Fault Detection
  - Nominally residual is approximately zero
  - Fault detected when residual deviation from zero is statistically significant
  - Usually there is a delay between fault occurrence and fault detection – cannot be avoided



Fault Signatures

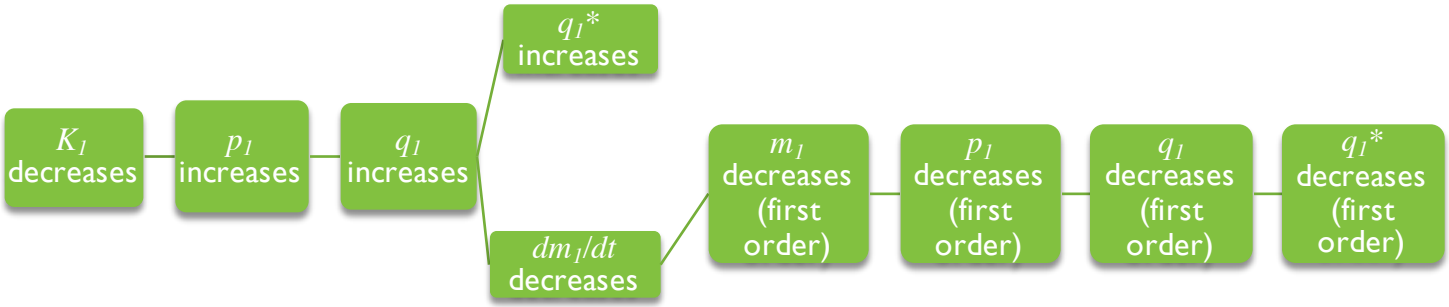
- Once a fault is detected, each measurement is qualitatively represented as symbols
  - 0 (at nominal), + (above nominal), and - (below nominal)
- Fault Signatures qualitatively capture the predicted effect of a fault on a measurement using the above symbols
  - All discriminatory evidence for fault isolation is provided by the first change in residual from time of fault detection
  - This reduces the possible fault signatures to {(+ -), (- +), (0 +), (0 -)}
  - {(++), (--)} imply positive feedback and hence, unstable systems



Fault	p_1	p_3
C_1^-	+-	0+
R_2^+	0+	0+
C_2^-	0+	+-

Deriving Fault Signatures

- Fault signatures are *predictions* of what the residuals will do in response to a fault
  - This information is already captured in the model!
  - Need to extract it
- Start with a causal model
- Example: For fault K_1^-, the fault signature of q_1^* is +-.



$$dm_1/dt = u_1 - q_1 - q_{12}$$
$$p_1 = m_1/K_1$$
$$m_1 = \int dm_1/dt$$
$$q_1 = p_1/R_1$$
$$q_1^* = q_1$$
$$q_{12} = (p_1 - p_2)/R_{12}$$

Fault Isolation

- Faults are isolated by comparing the qualitative deviation in measurements with the predicted fault signatures
  - Example: Consider fault set F = $\{C_1^-, R_2^+, C_2^-\}$ and measurement set M = $\{p_1, p_3\}$ all faults can be uniquely isolated
- Therefore, a system with faults F = $\{f_1, \ldots, f_l\}$, and measurements M = $\{m_1, \ldots, m_n\}$, is **diagnosable** if all single faults in F can be uniquely isolated using M
  - I.e., there is at least one distinguishing fault signature between f_i and all other faults in the system.

Fault	p_1	p_3
C_1^-	+-	0+
R_2^+	0+	0+
C_2^-	0+	+-

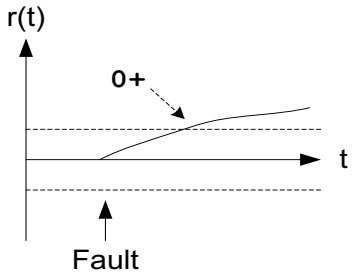
Fault Signature Matrix

Fault Isolation

- Faults are isolated by comparing the qualitative deviation in measurements with the predicted fault signatures
  - Example: Consider fault set F = $\{C_1^-, R_2^+, C_2^-\}$ and measurement set M = $\{p_1, p_3\}$ all faults can be uniquely isolated
- Therefore, a system with faults F = $\{f_1, \ldots, f_l\}$, and measurements M = $\{m_1, \ldots, m_n\}$, is **diagnosable** if all single faults in F can be uniquely isolated using M
  - I.e., there is at least one distinguishing fault signature between f_i and all other faults in the system.

Fault	p_1	p_3
~~C_1^-~~	~~+-~~	~~0+~~
R_2^+	0+	0+
C_2^-	0+	+-

Fault Signature Matrix



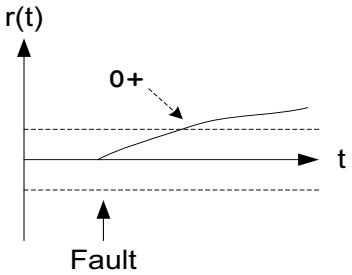
Pressure p_1

Fault Isolation

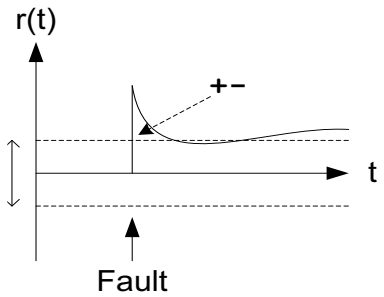
- Faults are isolated by comparing the qualitative deviation in measurements with the predicted fault signatures
  - Example: Consider fault set F = {C_1^-,R_2^+,C_2^-} and measurement set M = {p_1, p_3} all faults can be uniquely isolated
- Therefore, a system with faults F = {f_1, … ,f_l}, and measurements M = {m_1, … ,m_n}, is **diagnosable** if all single faults in F can be uniquely isolated using M
  - I.e., there is at least one distinguishing fault signature between f_i and all other faults in the system.

Fault	p_1	p_3
~~C_1^-~~	~~+-~~	~~0+~~
~~R_2^+~~	~~0+~~	~~0+~~
C_2^-	0+	+-

Fault Signature Matrix



Pressure p_1



Pressure p_3

Fault Identification

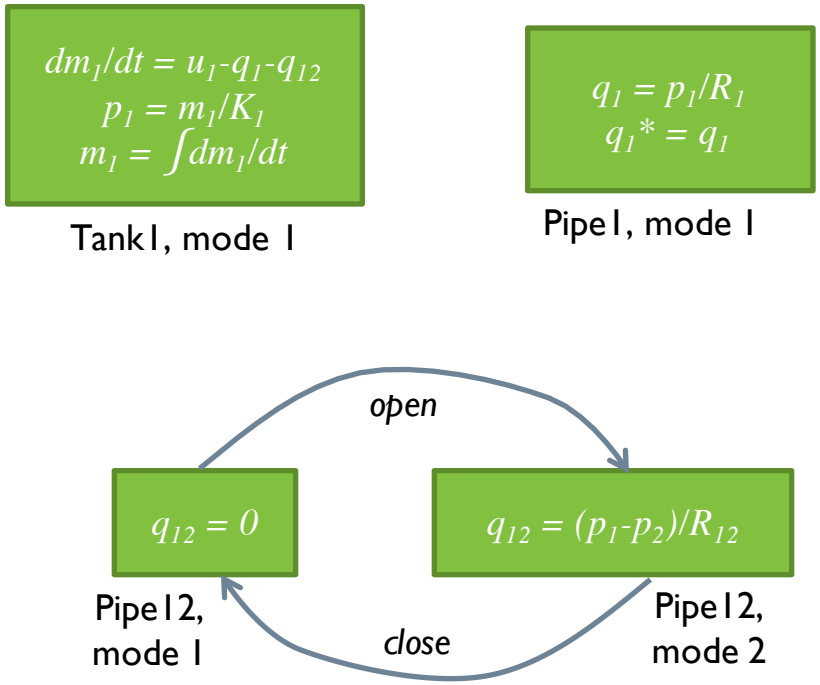
- Parameter estimation problem
  - Identify new (fault) parameter value, given observed faulty behavior
  - Several algorithms solve this problem
- One approach:
  - Determine an estimation window
    - Use data from *before* t_d (detection time) to t (current time)
  - Run an observer through that window, with the state vector augmented with the fault parameter (joint state-parameter estimation)
- Alternate approach
  - Derive submodel expressing unknown parameter as function of known/measured variables

Hybrid Systems Diagnosis

1. How do you model hybrid systems?
2. How do you diagnose faults in hybrid systems?
3. What are some practical considerations while diagnosing hybrid systems?

Modeling

- A *hybrid system* combines both discrete event and continuous behavior
  - Discrete behavior captured by different system *modes* and transitions between modes modeled using events
  - Within each mode is a description of the continuous behavior
- In a *component-based* modeling paradigm, define modes at a component level
  - System-level modes defined by specification of modes of each component



Fault Modeling

- Faults can be modeled as parameter changes (continuous part)
  - Termed *parametric faults*
- Faults can be modeled as events/modes (event-based part)
  - Termed *discrete faults*
- Modeling a fault as parametric or discrete is a modeling decision
  - Often, it makes more sense to model as one vs the other
  - For example, a switch going stuck closed can be modeled as a resistance going in infinity
  - Or, a parameter changing to a new value could be modeled as a new mode where the parameter has the new value

Mode Changes

- Transitions between modes can be specified in different ways
- Mode transitions are typically classified as follows
  - *Controlled* mode changes are known/commanded mode changes
    - Example: turn valve on/off
  - *Autonomous* mode changes are unobserved mode changes that are dependent on the system state
    - Example: flow through pipe depends on pipe height and water level in tank
- Autonomous mode changes can be difficult to deal with, because then it is difficult to track the hybrid system state

Fault Signatures

- We can derive fault signatures for each system mode
  - If the mode doesn't change during diagnosis, equivalent to continuous systems diagnosis
  - If the mode does change, then we need to reason about what mode the system was in when the change occurred and if there was any observation delay (e.g., due to imperfect detectors)

Mode 0,0

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	00	00
K_2^-	00	+−	00
K_3^-	00	00	+−
R_1^+	−+	00	00
R_{12}^+	00	00	00
R_2^+	00	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{on}$	0−	0+	00
$Pipe_{23}^{on}$	00	0−	0+
$Pipe_{12}^{off}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode 1,0

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	0+	00
K_2^-	0+	+−	00
K_3^-	00	00	+−
R_1^+	−+	0+	00
R_{12}^+	0+	0−	00
R_2^+	0+	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{off}$	0+	0−	00
$Pipe_{23}^{on}$	0−	0−	0+
$Pipe_{12}^{on}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode 0,1

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	00	00
K_2^-	00	+−	00
K_3^-	00	00	+−
R_1^+	−+	00	00
R_{12}^+	00	00	00
R_2^+	00	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{on}$	0−	0+	00
$Pipe_{23}^{on}$	00	0−	0+
$Pipe_{12}^{off}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode 1,1

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	00	00
K_2^-	00	+−	00
K_3^-	00	00	+−
R_1^+	−+	0+	00
R_{12}^+	0+	0−	00
R_2^+	0+	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{off}$	0+	0−	00
$Pipe_{23}^{on}$	00	0−	0+
$Pipe_{12}^{on}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Hybrid Systems Diagnosis: One Approach

- Assume single faults
- Assume all commanded mode changes are observed
- Assume observation delay is bounded
- Algorithm updates current diagnosis set based on new observation and previous observation sequence
  - For all recent mode changes
    - Check if current observed signature can start a fault trace in the mode, for the sublanguage of not-yet-deviated residuals

Algorithm: *Fault Isolation*

Inputs: Current diagnosis, previous trace, new symbol, recent modes

Outputs: New diagnosis

- Set new diagnosis to empty set
- **For** each recent mode
- **For** each fault in the current diagnosis
- **If** new symbol is consistent with signatures in the new mode
- Add fault to diagnosis
- **End if**
- **End for**
- **End for**

Example: Diagnosis of Parametric Fault

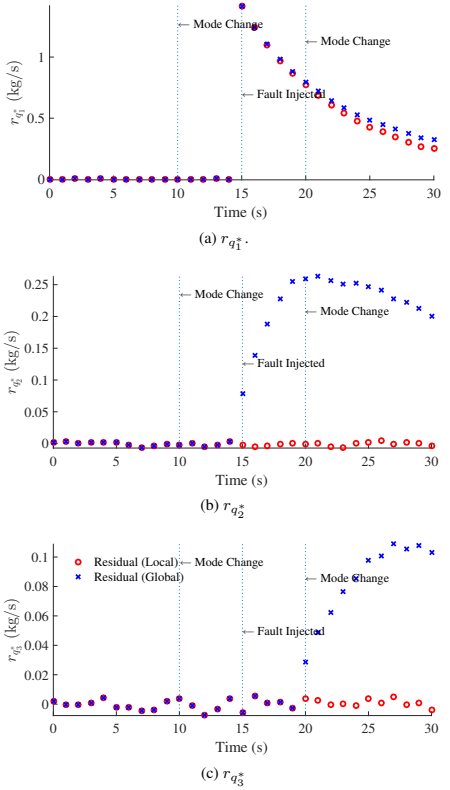
- Delay bounded by 5 s
- 0.0 s: Mode [0,0]
- 10.0 s: Mode [1,0]
- 15.0 s: K_1^- occurs
- Global
  - Consider past two modes
  - 15.0 s: r_{q1*} +- → $\{K_1^-\}$
  - 15.0 s: r_{q2*} 0+ → $\{K_1^-\}$

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	00	00
K_2^-	00	+−	00
K_3^-	00	00	+−
R_1^+	−+	00	00
R_{12}^+	00	00	00
R_2^+	00	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{on}$	0−	0+	00
$Pipe_{23}^{on}$	00	0−	0+
$Pipe_{12}^{off}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode [0,0]

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	0+	00
K_2^-	0+	+−	00
K_3^-	00	00	+−
R_1^+	−+	0+	00
R_{12}^+	0+	0−	00
R_2^+	0+	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{off}$	0+	0−	00
$Pipe_{23}^{on}$	0−	0−	0+
$Pipe_{12}^{on}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode [1,0]



Example: Diagnosis of Discrete Fault

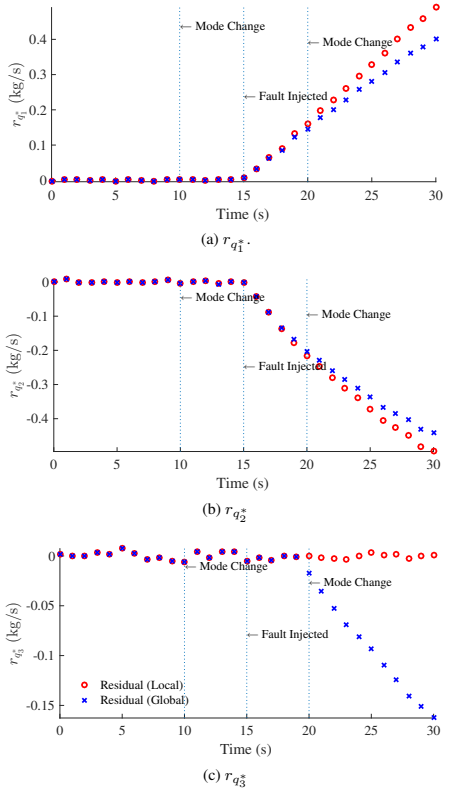
- Delay bounded by 5 s
- 0.0 s: Mode [0,0]
- 10.0 s: Mode [1,0]
- 15.0 s: $Pipe_{12}^{off}$ occurs
- Global
  - Consider past two modes
  - 16.0 s: r_{q1*} 0+ → $\{Pipe_{12}^{off}\}$
  - 16.0 s: r_{q2*} 0- → $\{Pipe_{12}^{off}\}$

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	00	00
K_2^-	00	+−	00
K_3^-	00	00	+−
R_1^+	−+	00	00
R_{12}^+	00	00	00
R_2^+	00	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{on}$	0−	0+	00
$Pipe_{23}^{on}$	00	0−	0+
$Pipe_{12}^{off}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode [0,0]

Fault	q_1^*	q_2^*	q_3^*
K_1^-	+−	0+	00
K_2^-	0+	+−	00
K_3^-	00	00	+−
R_1^+	−+	0+	00
R_{12}^+	0+	0−	00
R_2^+	0+	−+	00
R_{23}^+	00	00	00
R_3^+	00	00	−+
$Pipe_{12}^{off}$	0+	0−	00
$Pipe_{23}^{on}$	0−	0−	0+
$Pipe_{12}^{on}$	00	00	00
$Pipe_{23}^{off}$	00	00	00

Mode [1,0]



Distributed Diagnosis

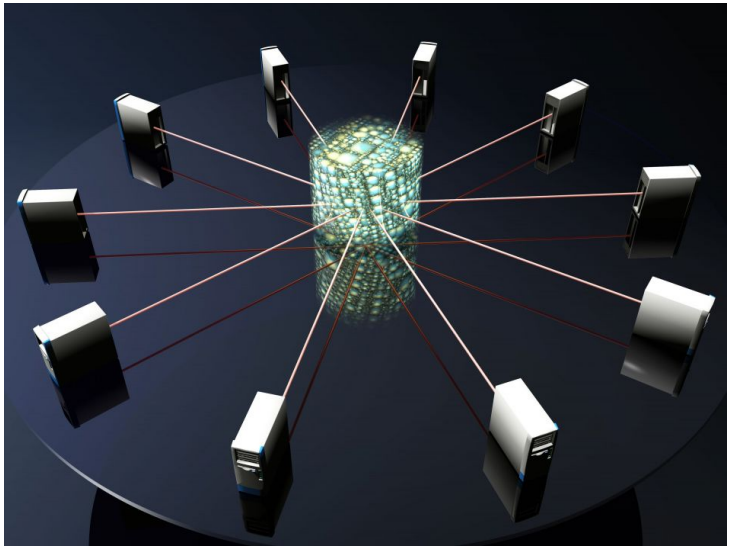
1. Why distributed diagnosis?
2. Global vs local diagnosability
3. A distributed diagnosis approach

Why Distributed?

- Centralized diagnosis schemes have some issues
  - Expensive in memory and computational requirements
  - Poorly scalable
  - Contain single points of failure
- Distributed diagnosis schemes address these issues
- One can distribute all aspects of diagnosis
  - Distributed detection
  - Distribute isolation
  - Distributed identification

What Does 'Distributed' Mean?

- Model-based diagnosis approaches broadly classified into
  - Centralized
    - Construct single diagnoser from global system model
  - Decentralized
    - Use a global system model but distribute diagnosis computations among multiple local diagnosers
    - Local diagnosis decisions are based on subset of observations and these decisions are communicated to other diagnosers or to a central coordinator
      - Use global model to generate globally consistent results
  - Distributed
    - Uses subsystem models and assume global model is unknown
    - Local diagnosers for each subsystem communicate their diagnosis results to each other to arrive at the global solution



Global vs Local Diagnosability

- Local diagnosability
  - Faults in a subsystem can be diagnosable from other faults in the same subsystem
- Global diagnosability
  - Faults in a subsystem can be diagnosable from all other faults in the system

Fault	p_{Tank1}	q_{Pipe1}	p_{Tank2}	q_{Pipe2}
C_1^-	+-	+-	0+	0+
R_{12}^+	0+	0+	0-	0-
C_2^-	0+	0+	+-	+-
R_{23}^+	0+	0+	0+	0+

Fault	p_{Tank1}	q_{Pipe1}	p_{Tank2}	q_{Pipe2}
C_1^-	+-	+-	0+	0+
R_{12}^+	0+	0+	0-	0-
C_2^-	0+	0+	+-	+-
R_{23}^+	0+	0+	0+	0+

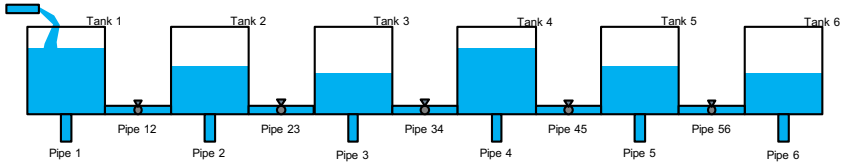
Distributed Diagnosis: Our Approach

- We want to distribute the diagnosis task into subtasks that can be executed on separate processors
- The (composed) global system model is analyzed offline to design distributed local diagnosers which generate globally correct diagnosis results
  - Without any coordination
  - With minimal or no exchange of information amongst themselves
  - Only observations are exchanged, if at all. No results are exchanged
- **Design Problem1:** Given the partition structure of the system, construct a local diagnoser for each known subsystem such that they require minimal information exchange amongst themselves
- **Design Problem 2:** Assuming no knowledge of subsystem partition, simultaneously construct subsystem partition + local diagnoser for each subsystem such that no information exchange between diagnosers is required for generating correct diagnosis results

Minimizing Shared Measurements – Algorithm 1

- This algorithm is useful for designing diagnosis schemes in practical distributed systems with known partition structure
- Given the prior knowledge of a subsystem Si with faults, Fi, and measurements, Mi
  - We identify the faults that are not globally diagnosable given Mi
  - Search for a minimal number of additional measurements that will make these faults globally diagnosable, giving preference to measurements from nearer subsystems

Fault	p_1	q_1	p_2	q_2	p_3	q_3	p_4	q_4	p_5	q_5	p_6	q_6
C_1^-	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+
R_{12}^+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-	0-	0-
C_2^-	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+
R_{23}^+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-
C_3^-	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+
R_{34}^+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-
C_4^-	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+
R_{45}^+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-
C_5^-	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+
R_{56}^+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-
C_6^-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-



({C_1, R_{12}},{p1, q1, **p2**})

({C_2, R_{23}},{p2, q2, **p3**})

({C_3, R_{34}},{p3, q3, **p4**})

({C_4, R_{45}},{p4, q4, **p5**})

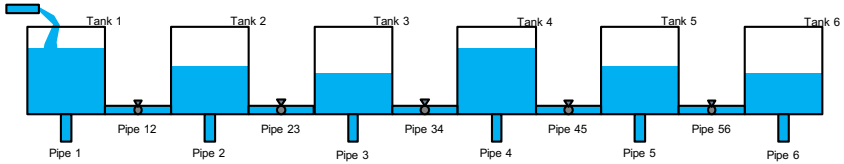
({C_5, R_{56}},{p5, q5, **p6**})

({C_6},{p6, q6})

Minimizing Shared Measurements – Algorithm 1

- This algorithm is useful for designing diagnosis schemes in practical distributed systems with known partition structure
- Given the prior knowledge of a subsystem Si with faults, Fi, and measurements, Mi
  - We identify the faults that are not globally diagnosable given Mi
  - Search for a minimal number of additional measurements that will make these faults globally diagnosable, giving preference to measurements from nearer subsystems

Fault	p_1	q_1	p_2	q_2	p_3	q_3	p_4	q_4	p_5	q_5	p_6	q_6
C_1^-	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+
R_{12}^+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-	0-	0-
C_2^-	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+
R_{23}^+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-
C_3^-	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+
R_{34}^+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-
C_4^-	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+
R_{45}^+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-
C_5^-	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+
R_{56}^+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-
C_6^-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-



({C_1, R_{12}},{p1, q1, **p2**})

({C_2, R_{23}},{p2, q2, **p3**})

({C_3, R_{34}},{p3, q3, **p4**})

({C_4, R_{45}},{p4, q4, **p5**})

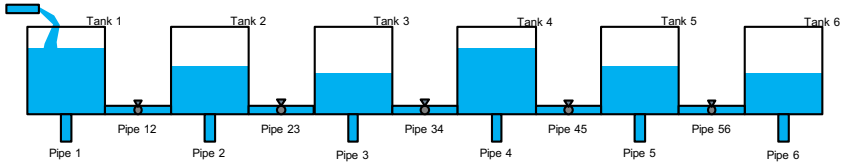
({C_5, R_{56}},{p5, q5, **p6**})

({C_6},{p6, q6})

Minimizing Shared Measurements – Algorithm 1

- This algorithm is useful for designing diagnosis schemes in practical distributed systems with known partition structure
- Given the prior knowledge of a subsystem Si with faults, Fi, and measurements, Mi
  - We identify the faults that are not globally diagnosable given Mi
  - Search for a minimal number of additional measurements that will make these faults globally diagnosable, giving preference to measurements from nearer subsystems

Fault	p_1	q_1	p_2	q_2	p_3	q_3	p_4	q_4	p_5	q_5	p_6	q_6
C_1^-	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+
R_{12}^+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-	0-	0-
C_2^-	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+
R_{23}^+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-
C_3^-	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+
R_{34}^+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-
C_4^-	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+
R_{45}^+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-
C_5^-	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+
R_{56}^+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-
C_6^-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-



({C_1, R_{12}},{p1, q1, **p2**})

({C_2, R_{23}},{p2, q2, **p3**})

({C_3, R_{34}},{p3, q3, **p4**})

({C_4, R_{45}},{p4, q4, **p5**})

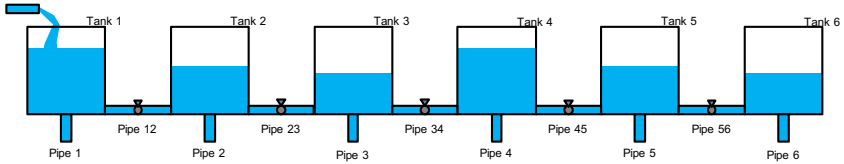
({C_5, R_{56}},{p5, q5, **p6**})

({C_6},{p6, q6})

Minimizing Shared Measurements – Algorithm 2

- This algorithm is more open-ended and designs distributed diagnosers based on diagnosability and efficiency criteria
- Assumes no knowledge of subsystem partition
- Simultaneously constructs subsystem partition + local diagnoser for each subsystem
- We partition sets F and M into subsets, F1, F2, …, Fn and M1, M2, …, Mn such that each Fi is globally diagnosable using Mi
  - Therefore no information is exchanged between diagnosers

Fault	p_1	q_1	p_2	q_2	p_3	q_3	p_4	q_4	p_5	q_5	p_6	q_6
C_1^-	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+
R_{12}^+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-	0-	0-
C_2^-	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+	0+	0+
R_{23}^+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-	0-	0-
C_3^-	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+	0+	0+
R_{34}^+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-	0-	0-
C_4^-	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+	0+	0+
R_{45}^+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-	0-	0-
C_5^-	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-	0+	0+
R_{56}^+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	0-	0-
C_6^-	0+	0+	0+	0+	0+	0+	0+	0+	0+	0+	+-	+-



({C_1},{p1})

({C_2, R_{12}},{p2})

({C_3, R_{23}},{p3, q2})

({C_4, R_{34}},{p4, q3})

({C_5, R_{45}},{p5, q4})

({C_6},{q6})

({R_{56}},{p6,q5})

Distributed Diagnosers

- The different diagnosers are independently run on the different processors
  - A distributed Extended Kalman Filter implementation provides each distributed diagnoser with a distributed observer
- Since the faults for each diagnoser forms a partition of the global fault set, a global diagnosis result is obtained when:
  - All measurements for a local diagnoser have deviated and the fault hypothesis set is reduced to a singleton fault set, or,
  - A local diagnoser's hypothesis set is reduced to a singleton but all of its measurements have not deviated, and all other diagnosers produce a null hypothesis, i.e., their candidate sets are empty

Conclusions

1. Challenges
2. Open problems

Challenges

- Modeling!
  - At what level of abstraction should one model?
  - How to combine results from different levels of abstractions?
- Online simulation
  - Problems with dynamic systems: initial conditions
- What is the source of complexity?
  - Complex systems or large systems (# components)
  - Exponential number of multiple fault candidates

Adapted from Anibal Bregon's talk on "Consistency-Based Diagnosis" at PHME14, Nantes, France.

Open Problems

- Integration of model-based diagnosis
  - With other diagnosis techniques
  - With other tasks, such as prognostics, re-configuration, repair, monitoring, supervision, …
  - Model-based diagnosis in the product life-cycle
- Re-usable model libraries

Adapted from Anibal Bregon's talk on "Consistency-Based Diagnosis" at PHME14, Nantes, France.

Selected Bibliography

- Armengol, J., Bregon, A., Escobet, T., Gelso, E., Krysander, M., Nyberg, M., Olive, X., Pulido, B., and Travé-Massuyès, L. Minimal Structurally Overdetermined sets for residual generation: A comparison of alternative approaches. In Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, SAFEPROCESS09, pages 1480–1485, Barcelona, Spain, 2009.
- Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. Diagnosis and FaultTolerant Control. Springer, 2006.
- Biswas, G., Cordier, M., Lunze, J., Travé-Massuyè, L., and Staroswiecki, M. Diagnosis of complex systems: bridging the methodologies of the FDI and DX communities. IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics, 34(5):2159–2162, 2004.
- Cordier, M., Dague, P., Lévy, F., Montmain, J., Staroswiecki, M., and Travé- Massuyès, L. Conflicts versus Analytical Redundancy Relations: a comparative analysis of the Model-based Diagnosis approach from the Artificial Intelligence and Automatic Control perspectives. IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics, 34(5):2163–2177, 2004.
- Guckenbiehl, T. and Schaefer-Richter, G. SIDIA: Extending prediction based diagnosis to dynamic models. In Proceedings of the 1st International Workshop on Principles of Diagnosis, DX90, pages 74–82, Stanford, California, USA, 1990.

Selected Bibliography

- Krysander, M., Aslund, J., and Nyberg, M. An Efficient Algorithm for Finding Minimal Over-constrained Sub-systems for Model-based Diagnosis. IEEE Trans. on Systems, Man, and Cybernetics. Part A: Systems and Humans, 38(1):197–206, 2008.
- Loiez, E. and Taillibert, P. Polynomial temporal band sequences for analog diagnosis. In Proceedings of the 15th International Joint Conference on Artificial Intelligence, IJCAI97, pages 474–479, Nagoya, Japan, 1997.
- Mosterman, P. and Biswas, G. Diagnosis of continuous valued systems in transient operating regions. IEEE Transactions on Systems, Man, and Cybernetics, 29(6):554– 565, 1999.
- Staroswiecki, M. Fault Diagnosis and Fault Tolerant Control, chapter Structural Analysis for Fault Detection and Isolation and for Fault Tolerant Control. Encyclopedia of Life Support Systems. Eolss Publishers, Oxford, UK, 2002.
- Struss, P. Model-based diagnosis for industrial applications. In ColloquiumApplications of Model-based Reasoning, Savoy Place, London, United Kingdom. Institute of Electrical Engineers (IEE), 1997.
- Travé-Massuyès, L. and Milne, R. Gas-Turbine Condition Monitoring Using Qualitative Model-Based Diagnosis. IEEE Expert, 12(3):22–31, 1997.
- I. Roychoudhury, G. Biswas, and X. Koutsoukos, "Designing Distributed Diagnosers for Complex Continuous Systems," IEEE Transactions on Automation Science and Engineering, vol. 6, no. 2, pp. 277-290, April 2009.
- I. Roychoudhury, M. Daigle, A. Bregon, and B. Pulido, "A Structural Model Decomposition Framework for Systems Health Management," in Proceedings of the 2013 IEEE Aerospace Conference, March 2013.
- A. Bregon, M. Daigle, and I. Roychoudhury, "An Integrated Framework for Model-Based Distributed Diagnosis and Prognosis," in Proceedings of the Annual Conference of the Prognostics and Health Management Society 2012, Minneapolis, MN, September 2012.
- I. Roychoudhury, G. Biswas, and X. Koutsoukos, "Distributed Diagnosis in Uncertain Environments Using Dynamic Bayesian Networks," in Proceedings of the 18th Mediterranean Conference on Control and Automation (MED'10), Marrakech, Morocco, June 2010.

Selected Bibliography

- Travé-Massuyés, L., Escobet, T., and Olive, X. Diagnosability analysis based on component supported analytical redundancy relations. IEEE Trans. Syst. Man Cy. A., 36(6), 2006.
- Washio, T., Sakuma, M., and Kitamura, M. A new approach to quantitative and credible diagnosis for multiple faults of components and sensors. Artificial Intelligence, 91(1):103–130, 1997.
- Williams, B. and Millar, B. Automated decomposition of Model-based Learning Problems. In Proceedings of the 7th International Workshop on Principles of Diagnosis, DX96, pages 258–266, Val Morin, Quebec, Canada, 1996.
- Williams, B. and Millar, B. Decompositional Model-based Learning and its analogy to diagnosis. In Proceedings of the 15th National Conference on Artificial Intelligence, AAAI98, Madison, Wisconsin, USA, 1998.
- M. Daigle, A. Bregon, X. Koutsoukos, G. Biswas, and B. Pulido, "A Qualitative Event-based Approach to Multiple Fault Diagnosis in Continuous Systems using Structural Model Decomposition," Engineering Applications of Artificial Intelligence, vol. 53, pp. 190-206, August 2016.
- M. Daigle, I. Roychoudhury, and A. Bregon, "Qualitative Event-based Diagnosis Applied to a Spacecraft Electrical Power Distribution System," Control Engineering Practice, vol. 38, pp. 75-91, May 2015.
- M. Daigle, A. Bregon, and I. Roychoudhury, "Distributed Prognostics Based on Structural Model Decomposition," IEEE Transactions on Reliability, vol. 63, no. 2, pp. 495-510, June 2014.
- A. Bregon, M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, and B. Pulido, "An Event-based Distributed Diagnosis Framework using Structural Model Decomposition," Artificial Intelligence, vol. 210, pp. 1-35, May 2014.
- M. Daigle, X. Koutsoukos, and G. Biswas, "An Event-based Approach to Integrated Parametric and Discrete Fault Diagnosis in Hybrid Systems," Transactions of the Institute of Measurement and Control, Special Issue on Hybrid and Switched Systems, vol. 32, no. 5, pp. 487-510, October 2010.
- M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, A. Patterson-Hine, and S. Poll, "A Comprehensive Diagnosis Methodology for Complex Hybrid Systems: A Case Study on Spacecraft Power Distribution Systems," IEEE Transactions of Systems, Man, and Cybernetics, Part A, Special Section on Model-based Diagnosis: Facing Challenges in Real-world Applications, vol. 4, no. 5, pp. 917-931, September 2010.
- M. Daigle, X. Koutsoukos, and G. Biswas, "A Qualitative Event-based Approach to Continuous Systems Diagnosis," IEEE Transactions on Control Systems Technology, vol. 17, no. 4, pp. 780-793, July 2009.

Additional Information Sources

- Conferences
  - PHM: http://www.phmsociety.org/
  - DX: http://www.dx-2016.org/
  - IJCAI: http://ijcai.org/
  - Safeprocess (part of IFAC organization): http://safeprocess15.sciencesconf.org/
  - IFAC world conference: http://www.ifac2014.org/
- Journals
  - Artificial Intelligence Journal
  - International Journal of the PHM Society (IJPHM)
  - Journal of AI Research
  - IEEE Transactions On Systems, Man and Cybernetics
  - AI Communications
  - Control Engineering Practice
  - Engineering Application on Artificial Intelligence
  - …

Adapted from Anibal Bregon's talk on "Consistency-Based Diagnosis" at PHME14, Nantes, France.

Acknowledgements

- Diagnostics and Prognostics Group, NASA Ames Research Center
  - Kai Goebel, Scott Poll, Edward Balaban, Chetan Kulkarni, Shankar Sankararaman, Adam Sweet, Lilly Spirkovska, Chris Teubert, George Gorospe, Ann Patterson-Hine
- Former NASA
  - Abhinav Saxena, Jose Celaya, Sriram Narasimhan
- Vanderbilt University
  - Gautam Biswas, Xenofon Koutsoukos
- University of Valladolid, Spain
  - Anibal Bregon, Belarmino Pulido, Carlos J Alonso-Gonzalez
- Some of the slides have been adapted from Anibal Bregon's talk on "Consistency-Based Diagnosis" at PHME14, Nantes, France.


thank you
danke
謝謝
ngiyabonga
شكراً جزيلاً
рахмат
Баярлалаа
спасибо
teşekkür ederim
gracias
tapadh leat
dank je
merci
obrigado
bedankt
dziękuję
sukriya
kop khun krap
go raibh maith agat
mochchakkeram
arigatô
dakujem
grazie
terima kasih
감사합니다
ありがとう
ευχαριστώ
мерси
хвала
faafetai lava
kiitos
dankie
dhanyavad
hvala
mauruuru
köszönöm
vinaka
спасибі
blagodaram
mersi
barka
welalin
tack
spas
kia ora
misaotra
matondo
paldies
grazzi
mahalo
asante
manana
obrigada
tenki
murakoze
djiere dieuf
tau
дякую
mamnun
chokrane
trugarez
takk
shukriya
mercé
dhanyavadagalu
diolch
taiku
sulpáy
tanemirt
rahmet
xièxie
rahmat
kam sah hamnida
najis tuke
sagolun
didi madloba
mèsi
děkuji
sobodi
enkosi
bayarlalaa
grazie
nanni
nandri
chnorakaloutioun
grafias ago
gràcies
akun dankon
aciú