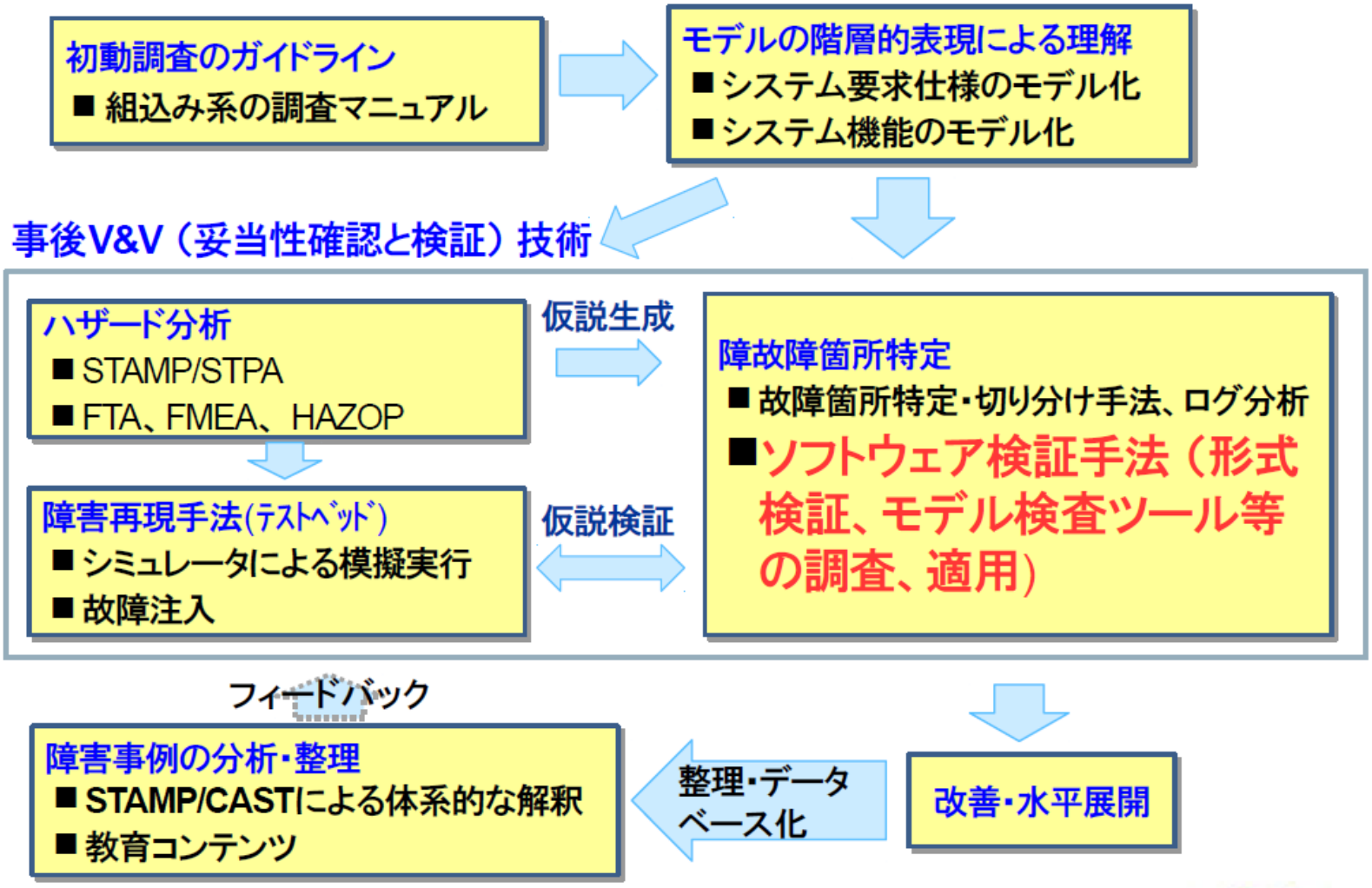
システム障害原因診断
～　形式手法の活用例　～

北道　淳司
会津大学　教授
kitamiti ＠ u-aizu.ac.jp

発表内容

1. システム障害原因診断WGの概要
2. 形式手法の概要と障害原因診断への適用
3. 対象システム例
4. LTSAを用いたモデリング
5. NuSMVを用いたモデリング
6. まとめ

故障原因診断WGの提言



形式手法の概要

- 論理的・数学的な枠組みを用いたシステムの記述法・設計法・検証法
  - ベースとなる論理、検証アルゴリズムなど 多くの種類がある
  - ET2014 においても、ツールなど形式手法に関連したブースがある

- ソフトウエア、ハードウエア、ネットワークなど様々なタイプのシステムに対して応用されている
  - 形式手法に関する書籍も揃ってきている
  - ・形式手法入門：中島　震　先生　、　オーム社
  - ・4日で学ぶモデル検査初級編：産総研システム検証研究センター、NTS
  - など多数

設計における形式手法

通常形式手法が用いられるのは、

- システム設計における初期段階
  - システム設計でも、抽象度の高いレベル
  - システムの要求仕様を策定する段階
- クリティカルな部分/信頼性を必要とされる部分に適用
  - 対象システムが、人（設計者）も検証ツールも扱える大きさ
    - 対象システムの規模に従い、形式検証ツールは、膨大な検証時間、必要とされるメモリ量を必要とする→実質、使えない
  - 開発開始前に、システムの仕様のリファレンスとして用いられることも
  - 実行可能な形式記述もある

モデル検査手法(モデルチェッキング)

形式検証手法の1つ

- システムを形式的な モデル として表す
  - 有限状態機械(オートマトン) など
- システムが満たすべき 性質 を論理式で表す
  - 時間の概念を含む性質に対しては 時相論理(Temporal Logic) など

- モデル検査手法: モデル が、性質 を満たすか、満たしながら動作するかを検索する
  - Yes なら、モデルは問題なく動作する
  - No なら、多くのツールは反例(性質を満たさない場合の実行系列)を出力
    反例を、デバッグに利用できる

障害原因診断

- 障害を発生したシステム
- 発生した障害
- に対して、なぜそのシステムにおいて障害が発生したかを診断する
- できれば、システムのどこにバグがあるか判断し、その障害が発生しないようにシステムを修正する

- 形式手法 を活用したい
- もともとシステムは形式手法で開発されていたかもしれないけれど
  - 具体レベルのシステム全体を対象としていないかもしれない。下流工程or開発においてバグが混入したかもしれないor障害発生箇所がクリティカルとみなされていない（形式手法の範囲外だった）かもしれない
- 派生開発、システムの経年劣化などで、システム動作の前提が、当初とは違ってきているかもしれない
  - モジュール交換時に、モジュールの仕様が変わっているものに置き換えられているかも知れない…

障害原因診断における形式手法

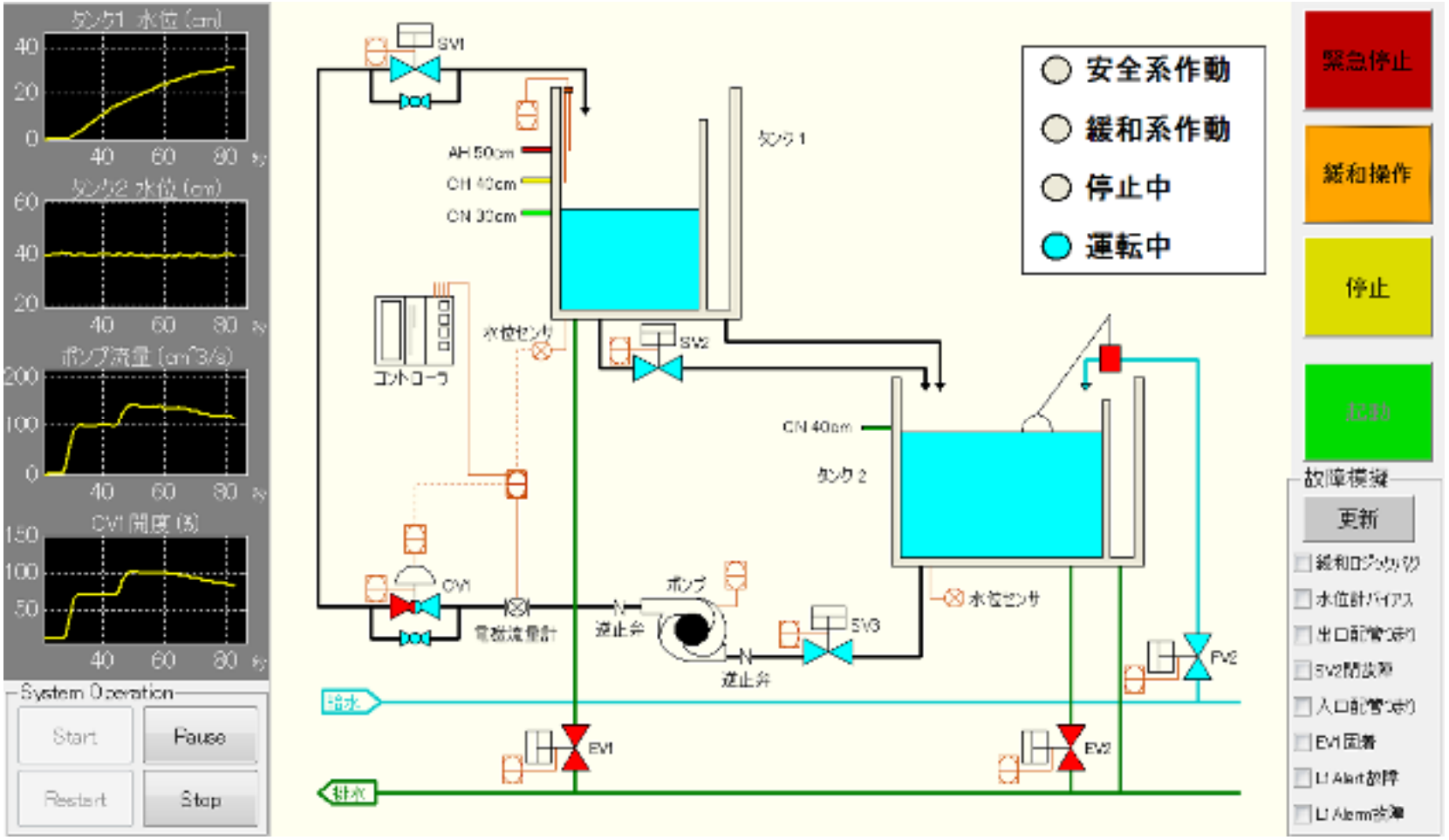
- システムを (抽象)モデル で表す
- 障害は、システムが満たしてほしい 性質 に対する違反とみなせる
  - モデル と 性質 を議論する モデル検査(チェッキング)手法 を利用する
- 通常の形式手法の利用と異なるのは
  - 参照するシステムのレベルが違う
    - 実装レベルが対象
    - 範囲をうまく限定する、動作＆データを抽象化してモデルを作成する必要がある
    - 障害の原因を保持して抽象化する必要がある

故障原因診断の例

- 化学プラント水槽
  - 2つの水槽（メインのタンク1、サブのタンク2）
  - タンク1が溢水しないように、運転員による手動とセンサ検出による緩和排水を制御する

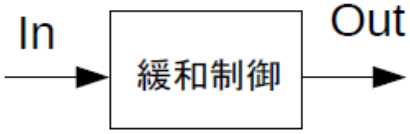
- 起こってしまった故障
  - センサがオンの状態（もうすぐ溢れるかもしれない状態）で、緩和期間をすぎているのに排水できない
  - 運転員の手動操作も受け付けない

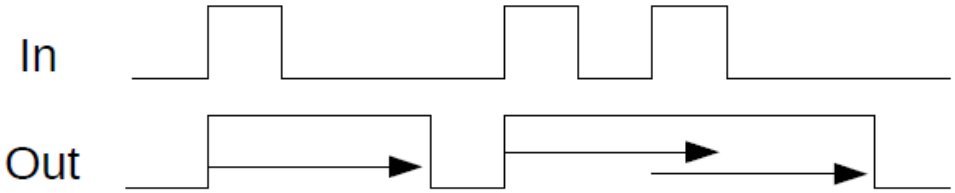
診断対象システムの概要



システムにおける緩和制御の詳細

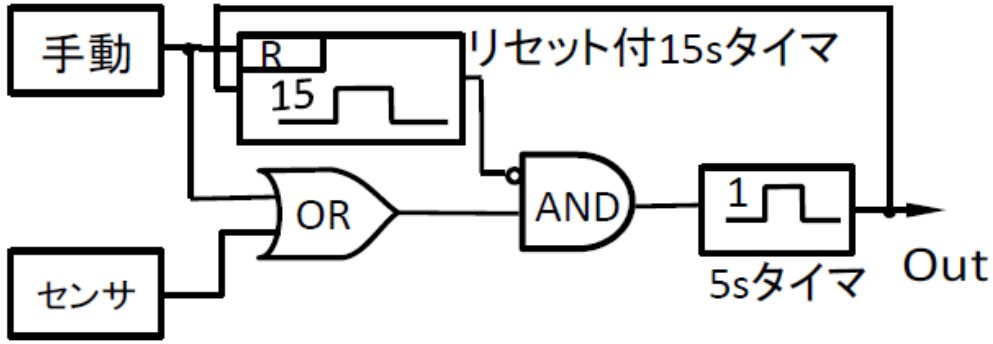
- 概要： In(手動orセンサいずれでも)が立ち上がると、指定された間(5秒間)緩和操作を行う(Outを立ち上げておく)





- 詳細： ディレイタイマを用いて手動入力を優先した実装

(手動による緩和が始まると15秒入力を無視、ただし手動入力が入るとリセット、その後入力受け付け)



というシステムで故障原因をしらべる

形式手法の例　LTSA

Labelled Transition System Analyser

開発元のホームページ：　http://www.doc.ic.ac.uk/ltsa/

単純なモデル検査ツール

NuSMV　や　Uppaal　などの手法より単純、それほど研究対象にもなってないが、形式手法の入門には適している

特徴

モデル：　複数のステートマシン

性質：　線形時相論理（Linear Temporal Logic）

ステートマシンの合成、デッドロック（それ以降実行可能な遷移がない状態）などのチェック、シミュレーションなどができる

LTSAを用いたリセット付きタイマのモデリングの例

- タイマの入出力の値の組み合わせを 状態 、値の変化を 遷移 に対応させる

Timer2_out = Timer2_i0_o0,

Timer2_i0_o0 = (and_out_up -> Timer2_i1_o0)

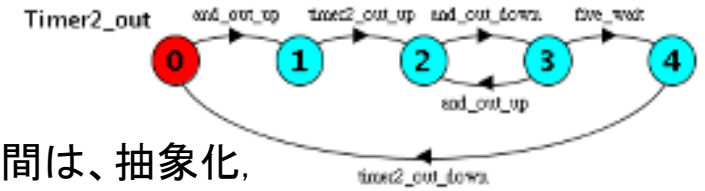
Timer2_i0_o1 = (and_out_up -> Timer2_i1_o1

| five_wait -> timer2_out_down -> Timer2_i0_o0), （パラメータ化）される

Timer2_i1_o0 = (timer2_out_up -> Timer2_i1_o1),

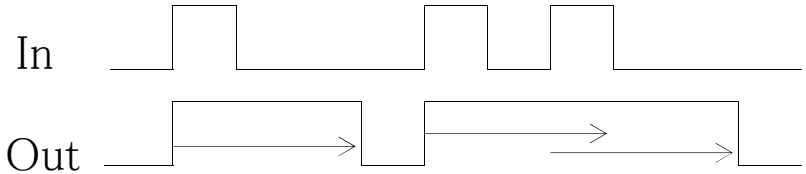
Timer2_i1_o1 =(and_out_down -> Timer2_i0_o1).

:



タイマの遅延時間は、抽象化，

| はチョイス（選択）で、図では分岐を表す。



- 入力が変化すると出力が変化
- する。同時には変化しない

LTSAにおいて時間の概念を表す手段

システムの性質を、遷移が実行される順として表す

fluent ： ある遷移が実行されてから、ある遷移が実行されるまでの間 のように時間を定義する構文

例 fluent Period = < event1 , event2 >

Assert ： 時間に関する性質を指定する構文

時間に関する演算子: [] いつも <> いつか

論理演算子: &&論理積 || 論理和 ！否定

例 assert property = event3 -> <> Period

遷移event3 が実行された後、いつか Preiod が真になる（event1が実行され、その後event2が実行される ということを表す論理式（命題）

LTSAを用いた性質の記述

発生した障害は

「センサがオンで、緩和期間をすぎているのに排水できない」

これから、成り立ってほしい 性質 を考える

↓

通常の状態で、センサがオンとなった時、いずれ「緩和モード」になってほしい

↓

出力Outが下がった後で、センサ入力Sensor_Inが上がったら、いずれ出力Outが立ち上がってほしい

↓

assert ALART_PROPERTY =

(timer2_out_down ->

<> (sensor_in_up -> <>timer2_out_up))

（本来なら）LTSAを用いた検証

モデル と 性質 を作成したら、LTSAを用いて

モデル　上において　性質 が成り立つか検証する

成立する：モデルが障害原因を含まない場合もある

モデルを見直し、再検証

成立しない：LTSAは反例を出力する

・初期状態から障害が発生する遷移系列を出力するので、動作を追って原因を特定する

・必要なら障害を発生しないモデルを作成して検証. 障害のない製品開発に利用する

形式手法　NuSMV

Nu(New) Symbolic Model Verifier

- FBK-irst, CMU, U. of Genova, U. of Trento の大学研究者が参加しているプロジェクト

  イタリアの非営利団体Fondazione Bruno Kesslerのホームページ　http://nusmv.fbk.eu/

・オープンソース、LGPL v2.1 licenseで利用可能

・「４日で学ぶモデル検査」などに解説、例題がある

NuSMVを用いたモデリング

遷移モデル（離散時間、離散データ）

変数：論理型、上下限値のある整数型

初期状態initと　遷移nextで動作を表す

状態遷移図（有限状態機械）は変数stateを用いて

init(state) := 0

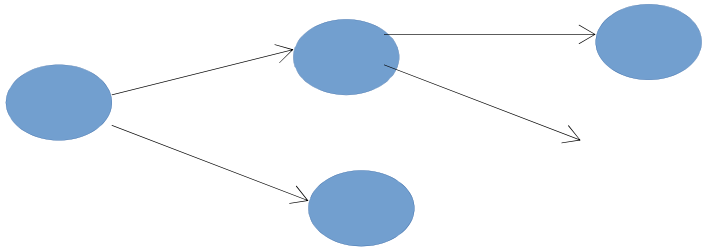
next(state) := case

state = 0 & i < 7 : 2;...

状態遷移 next が、時間、イベント、動作などを表し、時間に関する抽象度が変えられる

NuSMVを用いた性質の記述

- CTL（Computation Tree Logic)
- 初期状態からの実行系列を木と捉え
  - 全ての分岐で
  - ある分岐で
  - いつも、いつか、次の状態で
  - などを表す演算子で 性質 を表す



- LTL(Linear Temporal Logic)
- LTSAで用いたものと同じ（書き方は異なる）
- 常に、いつか、Until などを表す演算子で 性質 を表す
- NuSMVは CTL と LTL の両方使える（性質として用いることができる）

まとめ

- システム障害原因診断に対する形式手法の可能性
  - モデル、性質を作成し、障害（性質違反）を発見
  - 多数の　形式手法　から適したものを選択する
  - 　　時間（動作、時間の経過）、 データタイプをどう捉えるか、抽象化するか
  - フリーの形式ツールも多いので、是非お試しを。。

- 明日11月21日　14:00-15:30

  IPAセミナー [第７部]

  モデルベースアプローチに基づく障害原因診断手法

　にて、WGの活動内容、本プレゼンで用いたシステム、LTSA以外の形式手法（Uppaal: 今回の化学プラントの例ではLTSAより適しているかもしれません。）、障害原因とその対策方法の紹介があります