ZIPC状態遷移表モデルチェッカ Garakabu2

孔　維強
九州大学　大学院システム情報科学研究院

自己紹介

孔　維強 (KONG Weiqiang)

- 中国・武漢大学 (学士2000年、修士2003年)
- 北陸先端科学技術大学院大学 (博士2006年)
  - 定理証明とモデル検査の総合検証方法論
- 北陸先端科学技術大学院大学、ポスドック、2006年～2008年
- 福岡県産業・科学技術振興財団、研究員、2008年～2011年
  - Garakabu2の開発に参与
- 九州大学、特任准教授、2012年1月～
  - 形式検証技術に関する研究及びその教育場への応用

日本語がまだまだ得意でなく、申し訳ございません。プレゼンの時、意味が十分伝えられない時に、ご指摘・ご質問、よろしくお願い申し上げます。

目次

- Garakabu2の全体概要
- 事例：
  簡易両替システム：モデリング、性質の作成、Garakabu2で検査
- Garakabu2に実装された検査技術の概要

目次

- Garakabu2の全体概要

Garakabu2とは？

- 文部科学省の「地域イノベーション戦略支援プログラム（グローバル型【第II期】）」として、キャッツ株式会社、九州大学（福田晃教授）、福岡県産業・科学技術振興財団で研究開発を行っていたモデルチェッカである。

- Garakabu2の検査対象：
  - ZIPC状態遷移表設計（キャッツ（株））　←　モデル（MCの要素①）
  - 線形時相論理公式　←　性質（MCの要素②）

- Garakabu2を用いて、ZIPC状態遷移表設計（モデル）が線形時相論理公式（性質）に対する正しさ、を確認・分析できる。

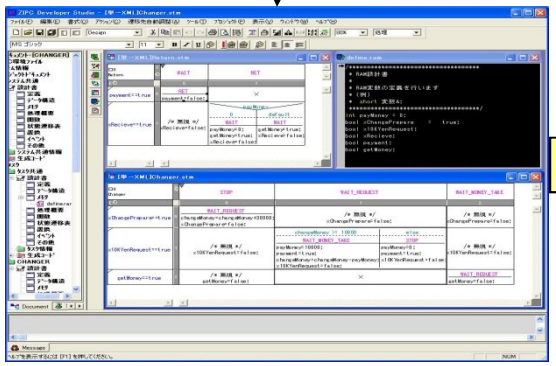
ソフトウェア開発工程においてどのように使えるのか？

⑤ 設計を修正、再検査. . .

① ZIPC状態遷移表で
設計（モデル）を作る

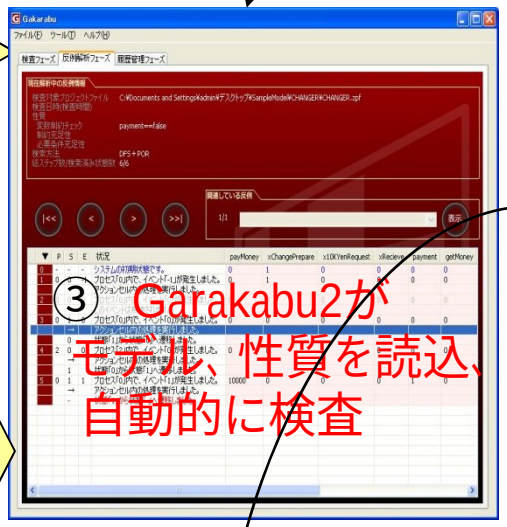
② 設計に対して確認・分析したい
性質を（検査項目として）入力

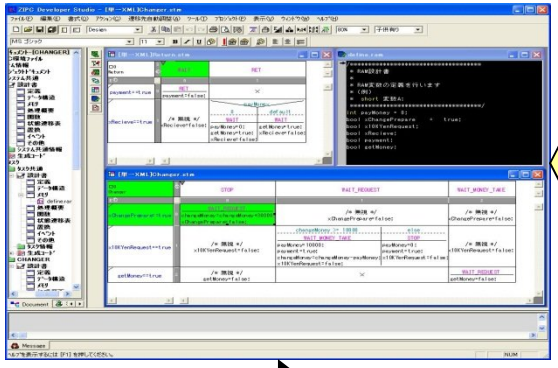
ZIPC



モデル取得

Garakabu2





異常実行
図表化の追跡

③ Garakabu2がモデル、性質を読込、自動的に検査

④ 設計の不具合を理解

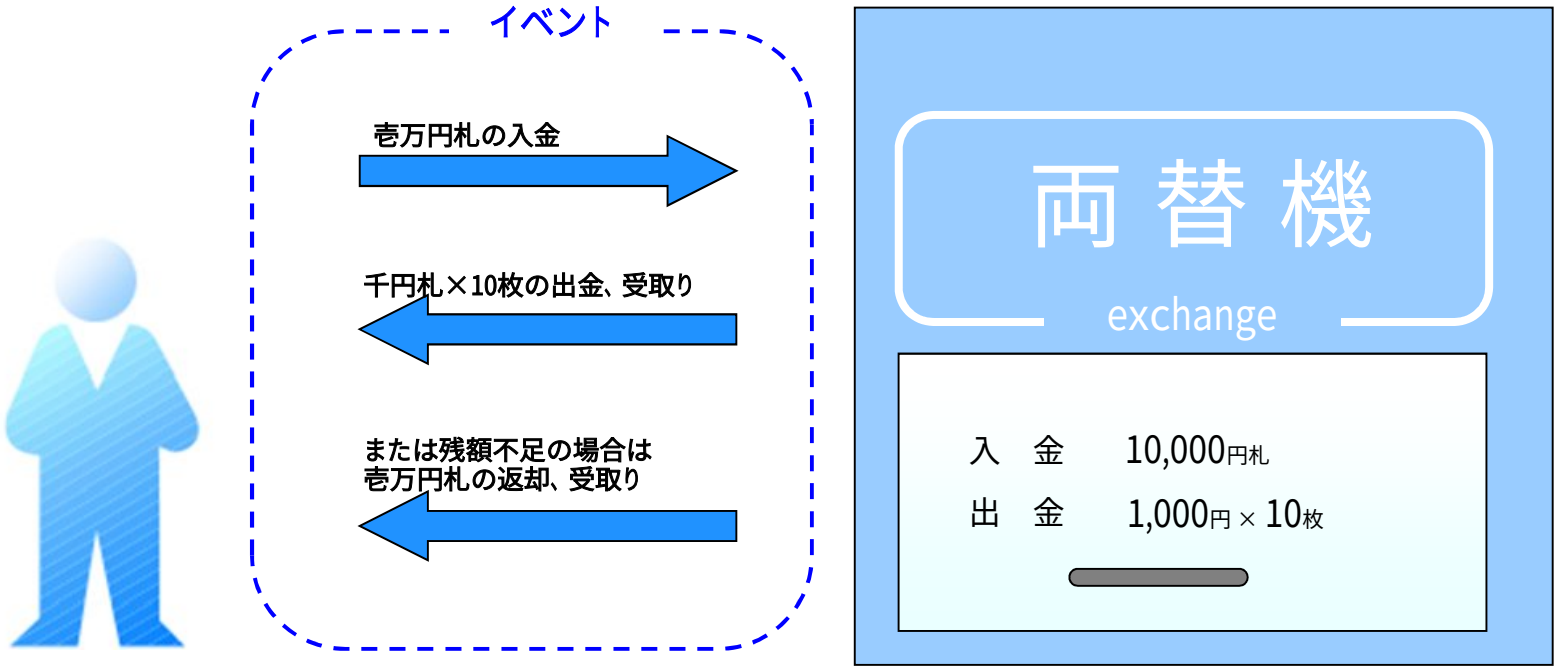
Garakabu2の特徴

- ZIPC状態遷移表モデルをそのまま検査できる
  - 使用しやすい： モデルチェッカの入力（形式）言語の習得は必要なし
- ZIPCと連動し、設計の不具合が図表化で表示できる
  - 理解しやすい： 不具合の発生の原因がより容易にわかる
- 検査結果がエビデンスとして残され、再現できる
  - トレーサビリティ： 以前の検査結果を確認できる
- 時相論理公式の作成がより簡単・直観
  - 産総研（関西CFV）との共同研究で開発されているSpecEditorを用いて、パターン化・図示化で公式作成できる

目次

- 事例：
  簡易両替システム：モデリング、性質の作成、Garakabu2で検査

簡易両替システム



基本仕様

- 壱万円札を入金すると、千円札×10枚に両替する。
- 残額不足の場合は両替出来ない (壱万円札を返却)
- 人が千円札×10枚を受け取るまで新しい壱万円札の入金は出来ない。
- 両替機は自動的に千円札×10枚を補充する。

モデル検査を試したいが…

- 質問1： どのようにZIPC状態遷移表を用いて、この簡易両替システムの振る舞いをモデリングするのか？
- 質問2： 両替システムのモデルに対して、確認･分析したいものは、どのように時相論理を用いて性質を作成するのか？
- 質問3： Garakabu2でどのような手順でモデル検査を行うのか？

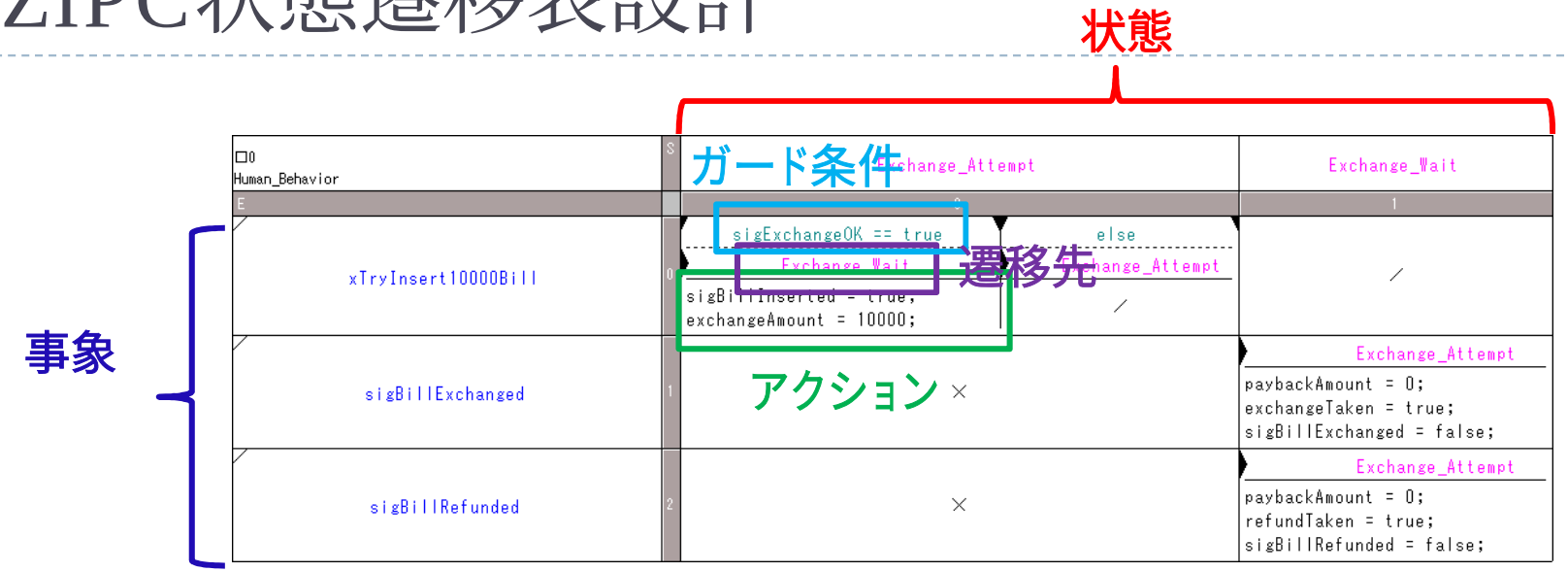
モデル検査を試したいが…

- 質問1： どのようにZIPC状態遷移表を用いて、この簡易両替システムの振る舞いをモデリングするのか？

質問2： 両替システムのモデルに対して、分析したいものは、どのように時相論理を用いて性質を作成するのか？

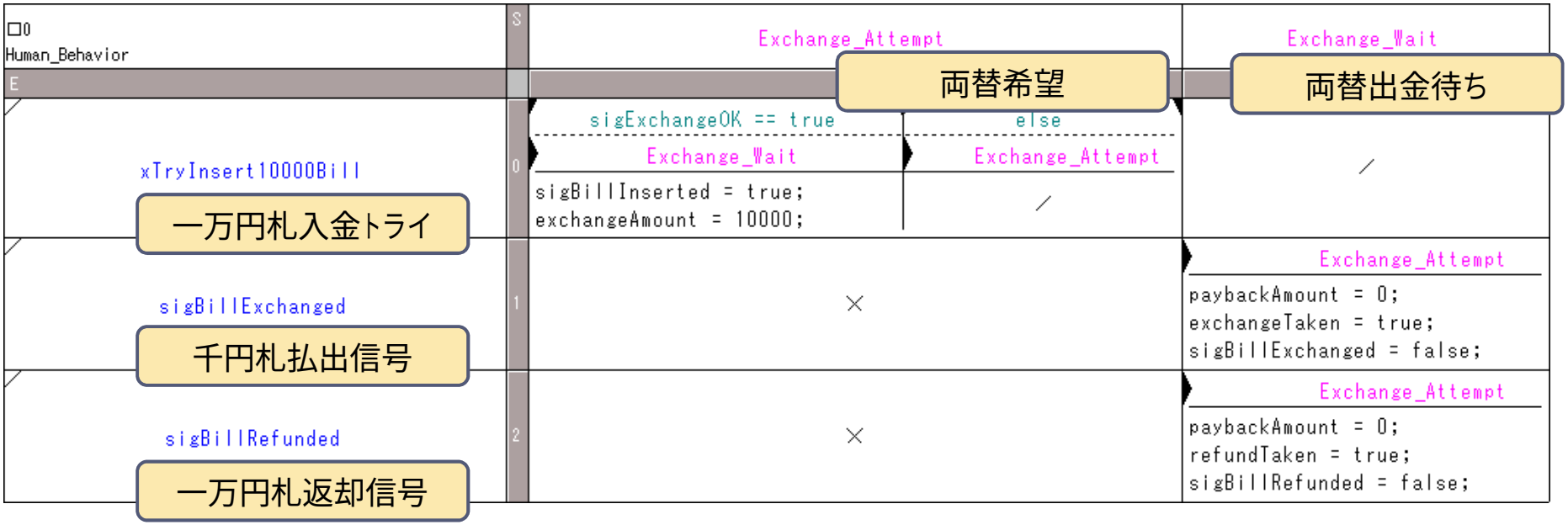
質問3： Garakabu2でどの手順でモデル検査を行うのか？

ZIPC状態遷移表設計



- ZIPC状態遷移表：
  - 状態を横軸、事象を縦軸に記入し、状態で事象が発火する時に、実行されるアクションとその後の遷移先状態は、状態と事象のクロスするセルに記入；
- ZIPC状態遷移表設計：
  - (複数の) 状態遷移表が含まれ、対象システムの全体の振る舞いを記述；
  - 状態遷移表の間、共通変数やメッセージなどの手段を用いて通信。

“人”の振る舞いを表す状態遷移表（参考案）

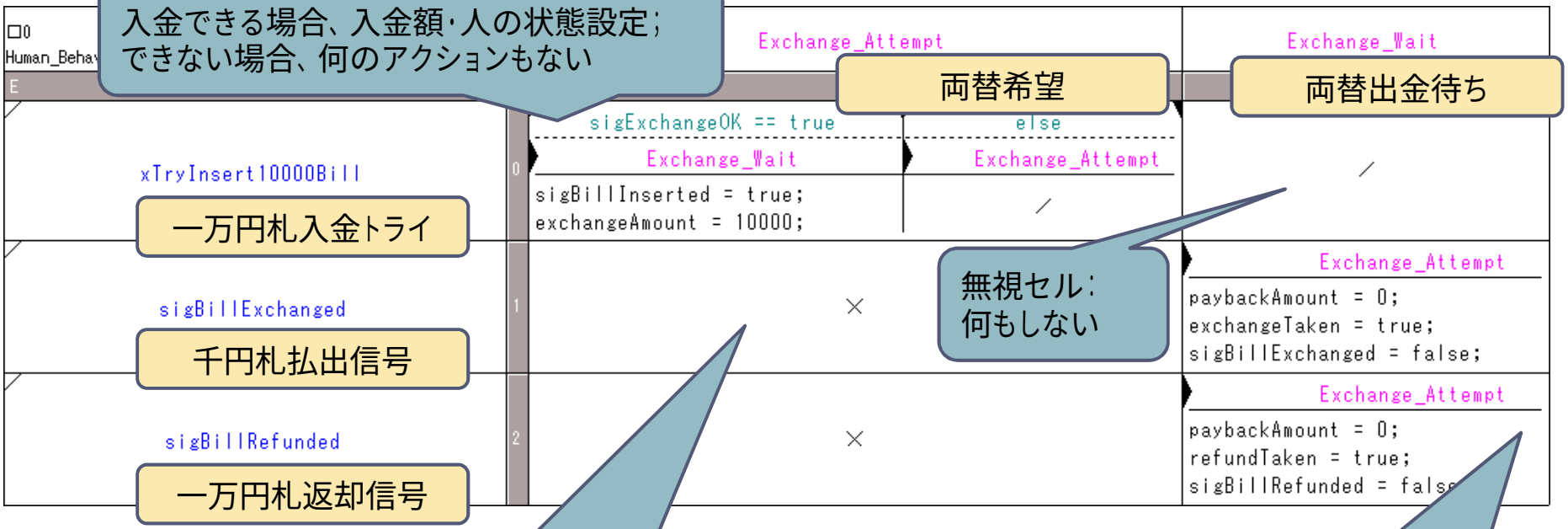


※異なった考え方によって異なった状態遷移表が作られるかもしれません

- 人は、【両替希望】と【両替出金待ち】の二つの状態を持っている
- 人は、【一万円札入金トラー】、【両替完了】、【入金返却】の事象に関わる
- 事象は、ブール型の表現式で表す（上記の場合、ブール変数）
  - 値が“false”→“true”の時、事象発火と、“true”→“false”の時、事象消耗と考える
- 先頭に“x”付きの事象名は、外部事象と呼び、いつでも発火可能の事象と考える

- ※共通変数を用いて通信する状態遷移表

“人”の振る舞いを表す状態遷移表（参考案）



※異なった考え方によって異なっ 移表が作られるかもしれません

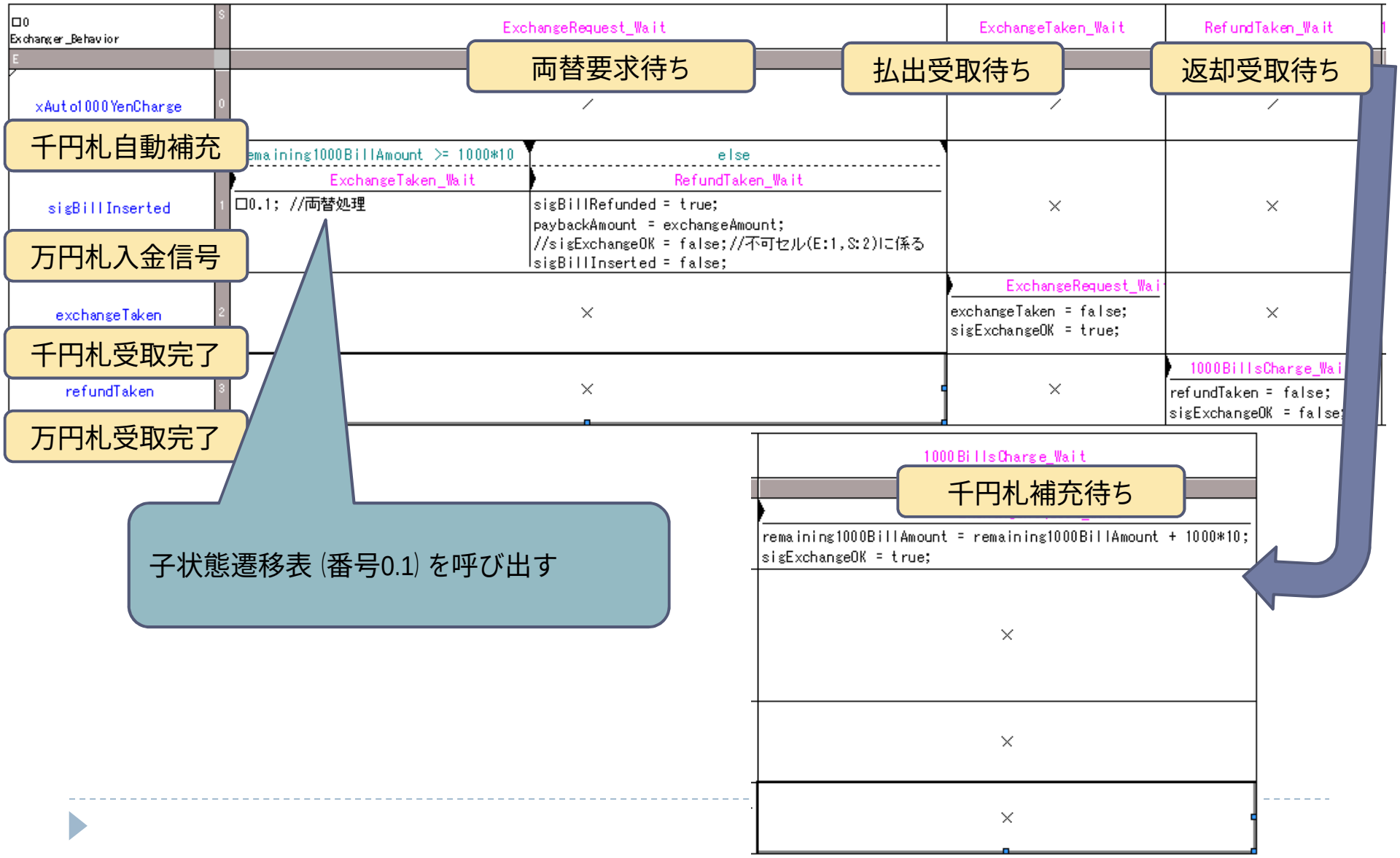
- の二つの状態を
- 人は、【一万円札入金トライ】、【両替完了】、【入金返却】の事象に関わる
- 事象は、ブール型の表現式で表す（上記の場合、ブール変数）
  - 値が“false”→“true”の時、事象発火と、“true”→“false”の時、事象消耗と考える
- 先頭に“x”付きの事象名は、外部事象と呼び、いつでも発火可能の事象と考える

- ※共通変数を用いて通信する状態遷移表

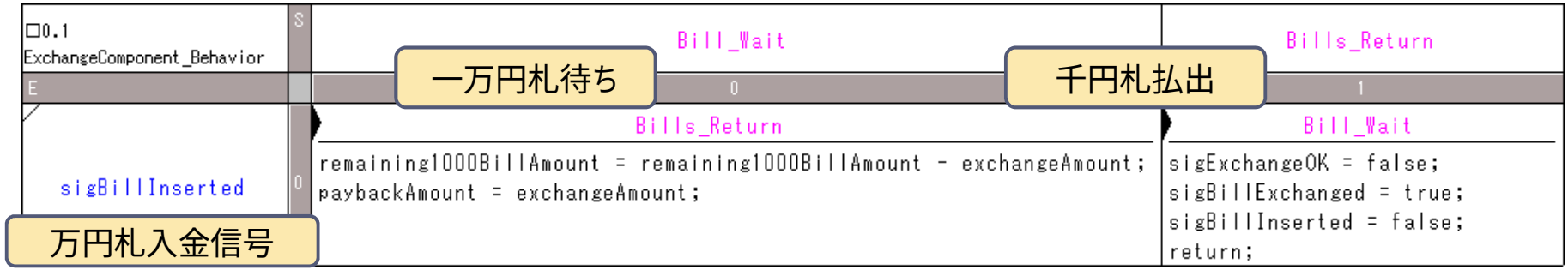
（事前に）定義された変数 & その初期値

- /**********
- * ZIPCのRAM設計書 * RAM変数の定義を行います
- **********/
- bool xAuto1000YenCharge=false; //外部変数 千円札自動チャージ
- bool xTryInsert10000Bill=false; //外部変数一万円札入金トライ
- bool sigBillInserted=false; //一万円札挿入信号
- bool sigBillExchanged=false; //千円札払出信号
- bool exchangeTaken=false; //千円札受取完了
- bool sigBillRefunded=false; //一万円札返却信号
- bool refundTaken=false; //一万円札受取完了
- bool sigExchangeOK=true; //紙幣挿入可信号
- int remaining1000BillAmount=10000; //両替機千円札残額
- int exchangeAmount=0; //両替紙幣（挿入）金額
- int paybackAmount=0; //払出金額

“両替機”の振る舞いを表す状態遷移表（参考案）



“両替機の両替部”を表す状態遷移表（参考案）



- この状態遷移表は、“両替機”状態遷移表の子状態遷移表として定義され、
- 親状態遷移表（両替機）から呼び出された時だけに、実行、
- “return”の式が実行された後に、親状態遷移表に戻る。

上述の三つの状態遷移表と変数定義はZIPC環境において定義

モデル検査を試したいが…

質問1： どのようにZIPC状態遷移表を用いて、この簡易両替システムの振る舞いをモデリングするのか？

- 質問2： 両替システムのモデルに対して、分析したいものは、どのように時相論理を用いて性質を作成するのか？

質問3： Garakabu2でどの手順でモデル検査を行うのか？

安全・活性性質 & 時相論理

- 形式手法における良く言及される性質
  - 安全性質 (Safety Property) ：何か悪いことが決して起こらない
  - 活性性質(Liveness Property)：何か良いことがいずれ起きる
  - デッドロック…等

- 線形時相論理 (Linear Temporal Logic)
  - [G](φ)：Globally ― φは今後常に真である。【安全性】
  - [F](φ)：Finally ― φは将来のいずれかの時点で真となる。【活性】
  - [N](φ)：Next ― φは次の状態で真である。
  - (ψ)[U](φ)：Until ― φは現在または将来の時点で真であり、かつψはその時点まで真である。
  - (ψ)[R](φ)：Release ― …
  - 上記の表記の組み合わせ、例えば、[G][F](φ)、[F][G](φ)

確認したいものを時相論理を用いて表す

- “不可セルに到着できない”の検査（ありえない状態にならないこと）
  - ZIPC状態遷移表における“×”のセル → 不可セル
    設計者の想定であるため、実際にそうであるかどうか、検査必要
  - 時相論理で表せるが、Garakabu2で不可セル検査に対して入力不要

- 他の仕様に関すること、例えば、
  - 入金額と出金額が一致すること

```
[G]((sigBillExchanged || sigBillRefunded)
                => (paybackAmount == exchangeAmount))
```

変数の意味の回顧：

```
bool sigBillExchanged;    //千円札払出；
bool sigBillRefunded;     //万円札返却；
int    exchangeAmount; //両替紙幣（挿入）金額；
int    paybackAmount     //払出金額；
```

論理オプレータの意味の復習：

[G]は、“常に”と意味
|| は、“もしくは”と意味
=> は、“if 左 then 右”と意味

モデル検査を試したいが…

質問1： どのようにZIPC状態遷移表を用いて、この簡易両替システムの振る舞いをモデリングするのか？

質問2： 両替システムのモデルに対して、分析したいものは、どのように時相論理を用いて性質を作成するのか？

- 質問3： Garakabu2でどのような手順でモデル検査を行うのか？

Step1: ZIPC状態遷移表設計をGarakabu2に入力

- 設計の記述ミス、もしくは、Garkabu2が対応していない記述方法がありましたら、その詳細が報告される。

Step2: 検査したい状態遷移表を選択

- 通常、ZIPC設計に含まれたすべての状態遷移表を検査するが、特定の状態遷移表の一部のみを検査する時に、部分選択・検査もOK
- 選択できるパターン（制限）はすでにGarakabu2に実装されている。言い換えると、制限違反のパターンは選択できない。
- 例えば、子状態遷移表がその親状態遷移表しか呼び出せられないため、子状態遷移表のみ（親を選択せず）と、他タスクの状態遷移表は選択できない。

Step3: 変数の初期値・最小値・最大値を設定

変数型	初期値の範囲	
	最小値	最大値
bool	false	true
byte	-128	127
char	-128	127
short	-32768	32767
int	-2147483648	2147483647

- ZIPCのRAMファイルに定義された初期値は自動的に読み込まれる
- モデル検査中に設定した閾値範囲外の値になった場合、報告される
- 最小値、最大値の設定は、必須ではない

Step4: 検査したい性質の入力

- ✓ 不可セル
- ✓ 時相論理性質

- 閾値チェック
  - 最小値･最大値の範囲外の値になったこと

- 制約充足性
  - 状態遷移表AがXの状態の時、状態遷移表BがYの状態であること

- 必要条件充足性
  - 状態遷移表Aの状態がXからX’に遷移する時、状態遷移表BがYの状態であること

- デッドロック　(入力不要)

性質の具体例

- 閾値チェック
  - 最小値･最大値の範囲外の値になったこと

  exchangeAmountの最大値を10000に設定（*5000に設定したら反例あり*）

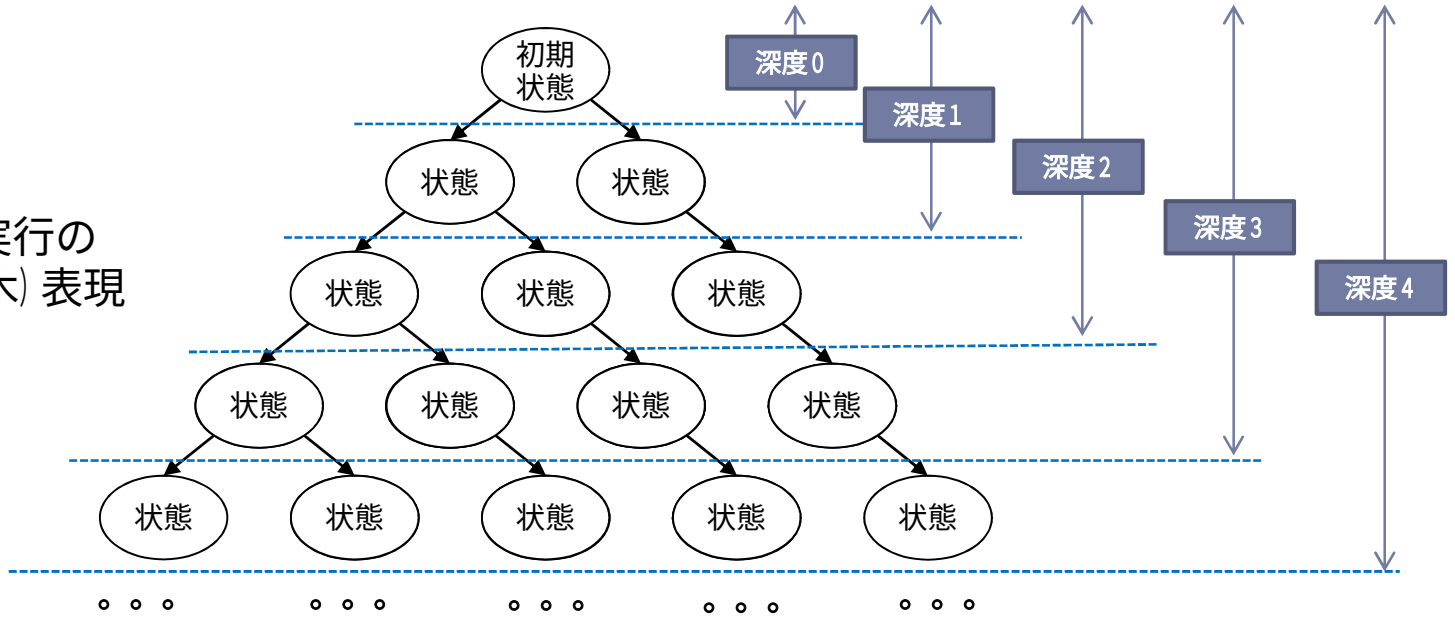
- 制約充足性
  - 状態遷移表AがXの状態の時、状態遷移表BがYの状態であること

  両替機が両替処理をしている間、人は両替完了を待っている

“深度”とは

- Garkabu2では、有限モデル検査を行います。

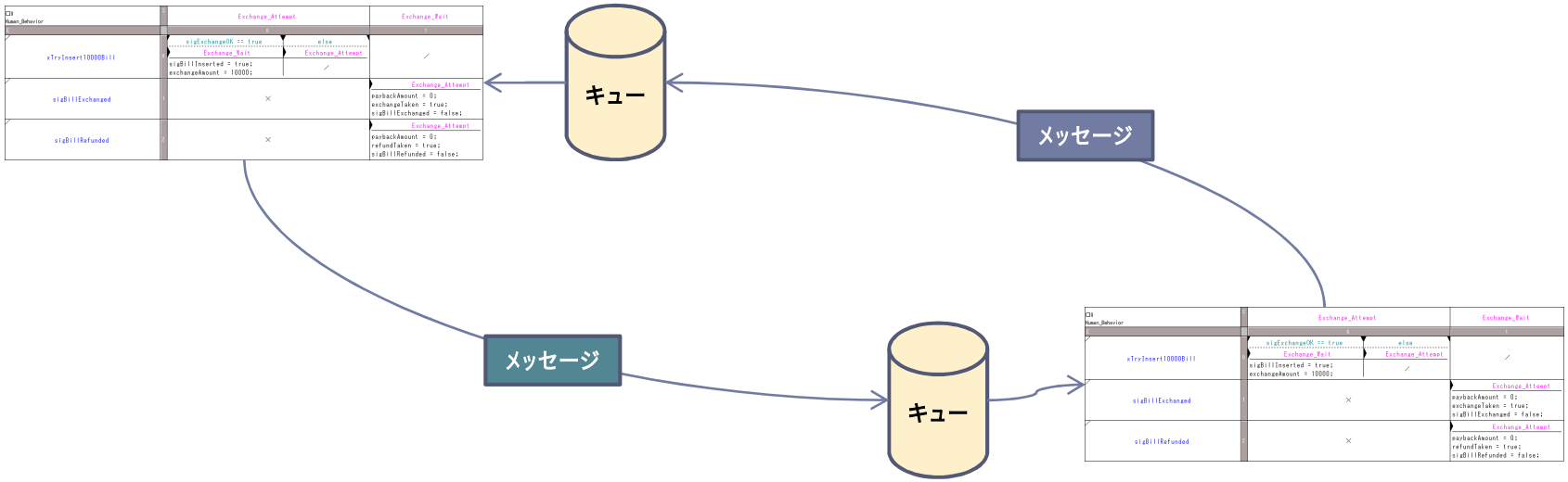
システム実行の
内部（実行木）表現



- 注意点： 変数閾値検査がパスした後に、変数閾値条件は他の性質の前提条件として使われたため、変数閾値の深度を超える設定ができない。

“キューサイズ”とは

- 複数の状態遷移表の間、メッセージで通信の場合、キューが必要となる
- タスクごとに(一つの)キューを持っている
- キューサイズは、キューに入られるメッセージの数である



- *※共通変数(フラグ型)で通信の場合、キューが必要なし。*

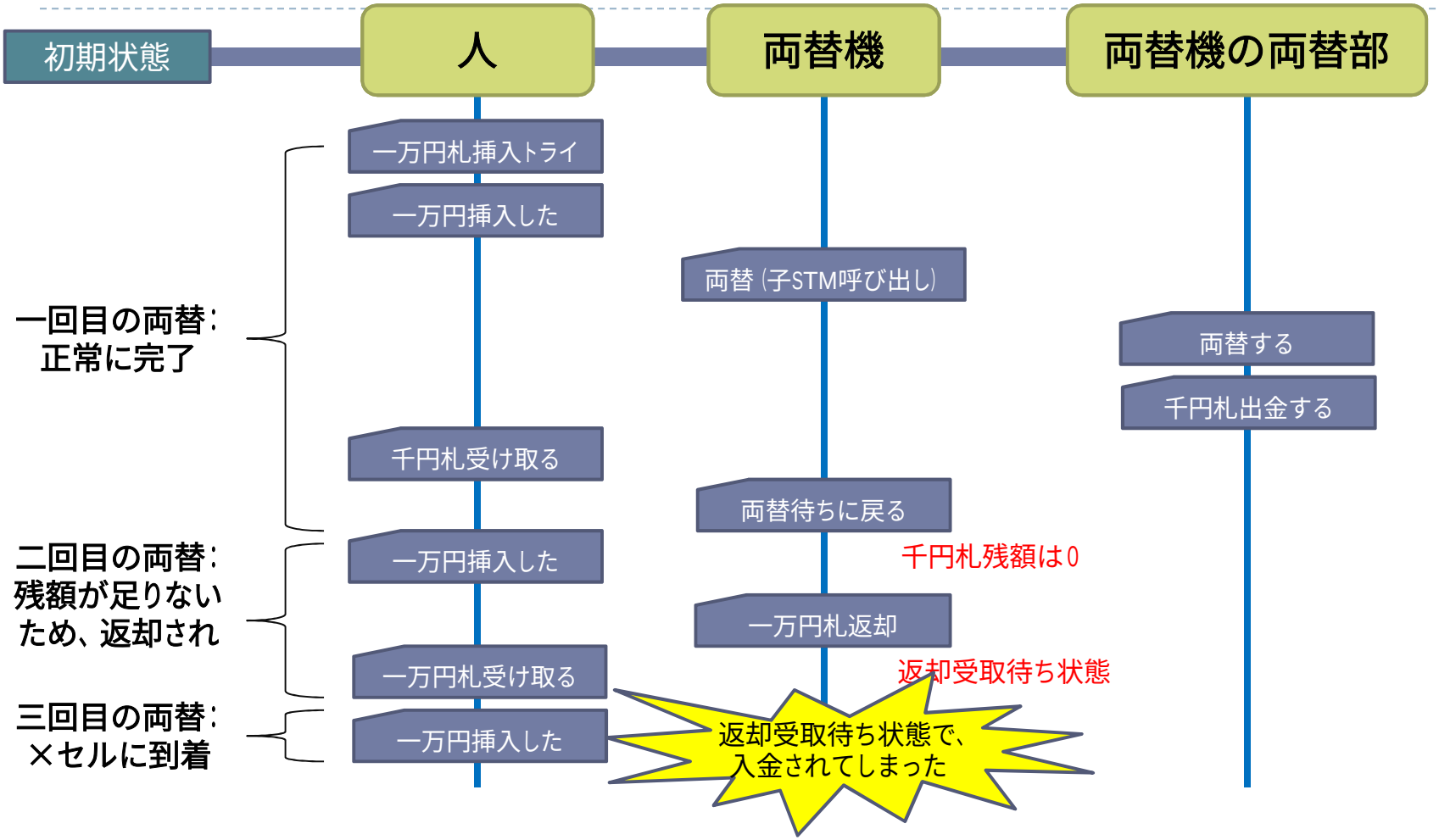
Step5: 検査結果の読み方

- 検査完了後に、
  - 黒字の性質は、反例なし（成り立ち）
  - 赤字の性質は、反例あり（成り立たない・不具合あり）

- 反例あり（赤字）の性質をクリックすると、
  初期状態から反例になるまでの状態の遷移が一覧で表示されます。
  （反例経路と呼ばれ）

- この一覧の行をクリックすると、
  ZIPC状態遷移表の対象セルが動的に表示されるので、反例発生するまでの流れを確認することができます。

反例経路の読み方



- 修正してみたら？もう一度検査してみましょう！！！

履歴管理とは

- 検査した結果を、一覧で確認することができます

①履歴管理フェーズタブを選択し、

②リスト内の項目をダブルクリックすると反例解析フェーズ画面に切り替わり、

その時のモデル検査結果を確認することができます。その場合、画面上部のタイトルは「過去のモデル検査情報」と表示され、検査開始日時は黄色で表示されます。

- 赤字の反例をクリックすると、反例経路が表示されます。

時相論理公式の作成は難しい…

- 下記の性質の公式はどうやって作成できるのか？

日本語で：

両替機が千円札×10枚を払い出してから、人が受取を完了するまで、
新しい一万円札の入金ができないこと

論理公式で：

```
[G](((sigBillExchanged == true) && !(exchangeTaken == true) &&
      [F](exchangeTaken == true)) =>
                  ((sigExchangeOK == false) [U] (exchangeTaken == true)))
```

- SpecEditorを用いて、より簡単に作成できる

※SpecEditorは、産総研 (関西CFV) が開発 (中)

目次

- Garakabu2に実装された検査技術の概要

SMTベースにした有限モデル検査の概要

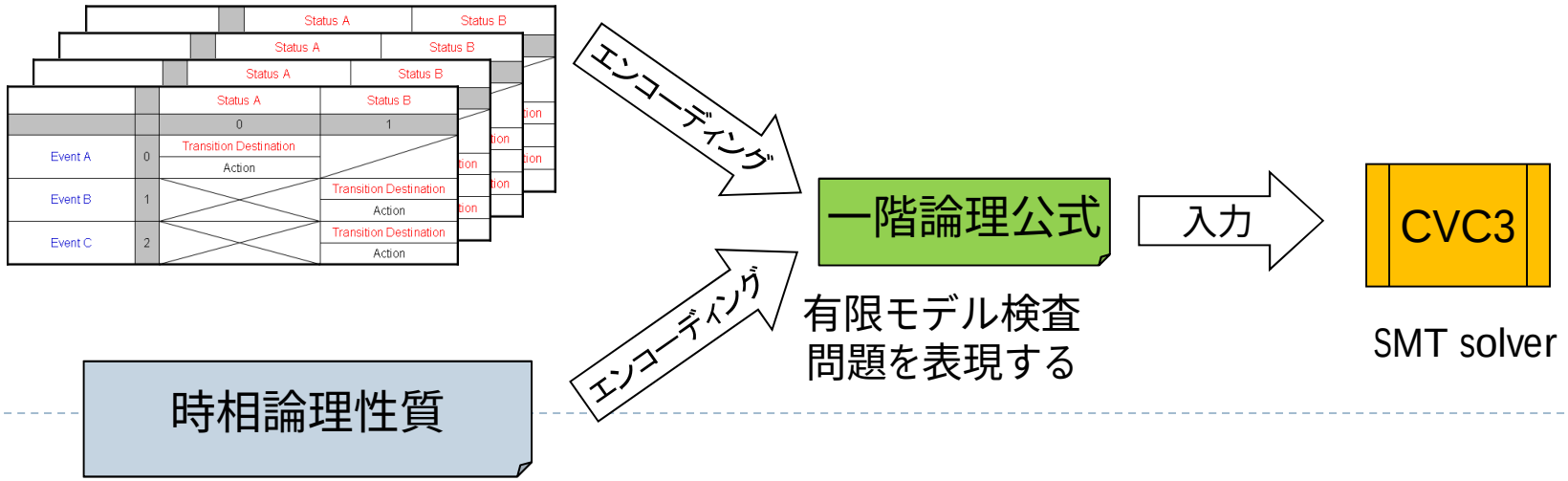
- SMT (Satisfiability Modulo Theories) 技術とは

公式の値をtrueになるため、公式に含まれたすべての変数の値（即ち、変数の代入）を見つける技術である。

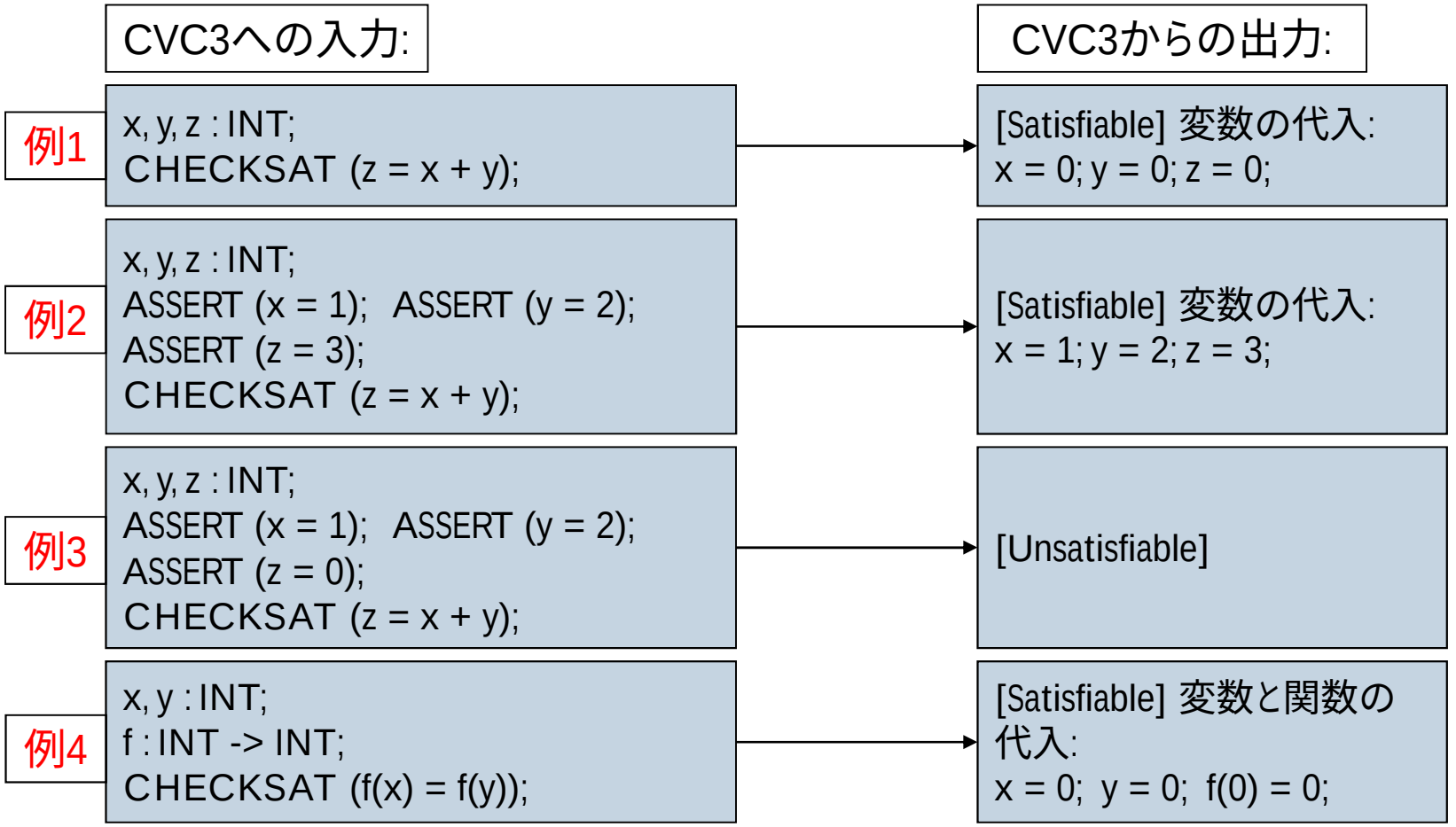
- Garakabu2に使用されているSMT solverは

CVC3とは

ニューヨーク大学はメインとして開発したオーポンソースSMT solverである。



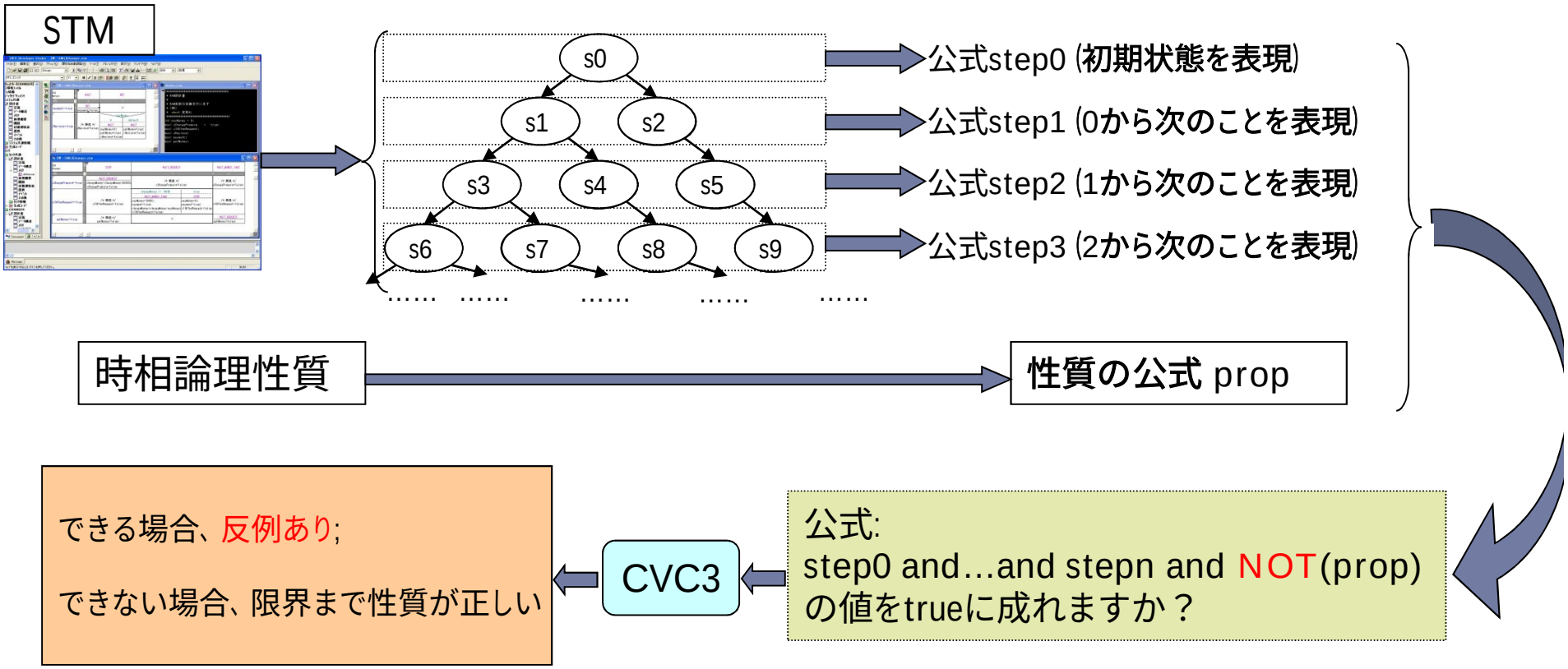
CVC3の使用例



例に使われたCVC3のコマンド:

1. ASSERT: 論理コンテキストに公式を追加;
2. CHECKSAT: 作成した論理コンテキストに対して、公式の値がtrueに成れるかどうか計算。trueになれる場合は、変数の代入を出力; ではない場合は、Unsatisfiableを出力。

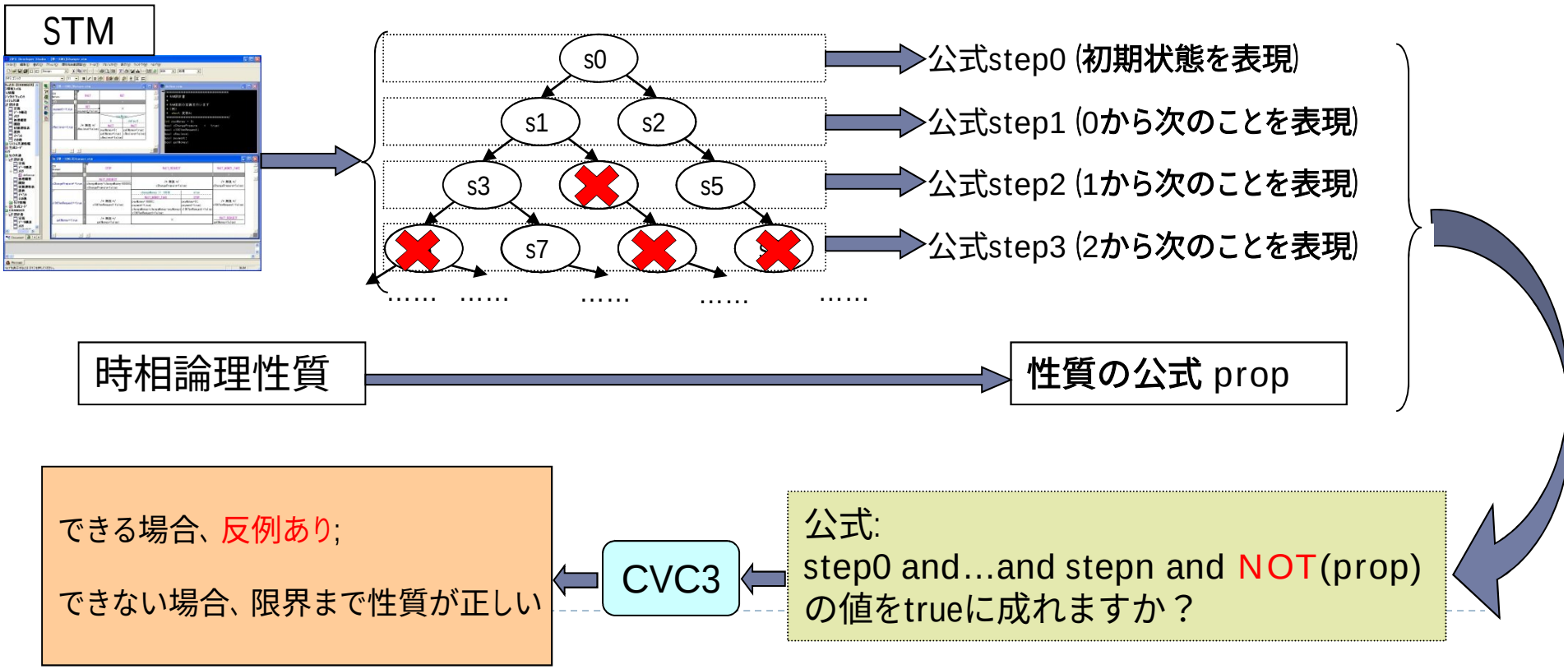
ZIPC状態遷移表の
有限モデル検査（BMC）のアイディア



ZIPC状態遷移表の
有限モデル検査（BMC）のアイディア

ハイブリッドモデル検査 ― Explicit検査とBMC検査の融合

① BMC検査の前に、Explicit検査で、有限状態空間を（stateless）探索；
※デッドロック等を検出；Explicit状態空間縮小技術を活用
② BMC検査で、各階層での実行できるトランジションのみを、エンコード；



技術の詳細は

- 国際会議 APSEC2011とAPSEC2012に出版のGarakabu2に関する論文にご参考いただければ幸いです。【2011：Garakabu2の基本の検査方法；2012：Garakabu2の加速検査方法】

“簡単に使えるモデルチェッカ”
を目指しています！

ご清聴ありがとうございました！