高信頼自動運転システム実現にむけて

ソフトウェア科学・形式手法・数理的基盤の視点からの技術俯瞰

蓮尾 一郎

国立情報学研究所 (NII) システム設計数理国際研究センター センター長・准教授

JST ERATO 蓮尾メタ数理システムデザインプロジェクト 研究総括

総合研究大学院大学 准教授

まず最初に：

ソフトウェア品質保証におけるスペクトル

形式手法

テスト test | モデル検査 | 自動定理証明 | 対話定理証明

低 ← コスト → 高

高 ← 自動化の度合い → 低

自動 | 専門家を月・年単位で拘束

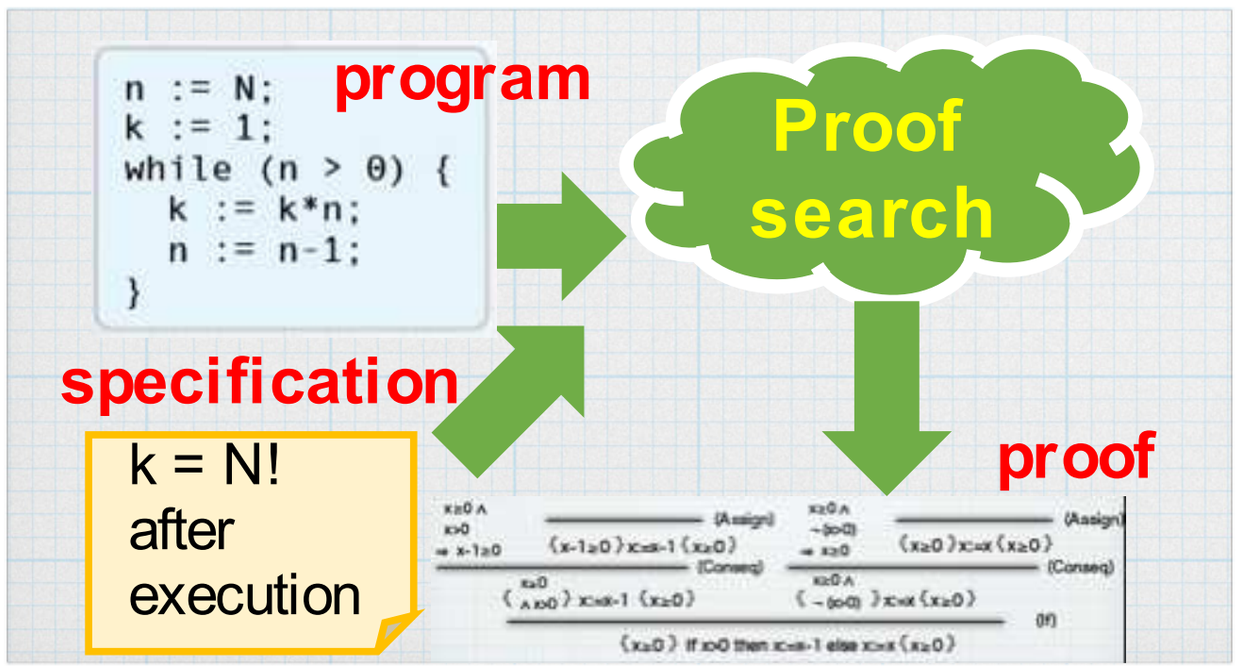
低 ← 保証の網羅性 → 高

低 ← 保証の数学的厳密性 → 高

経験論的保証 | 数学的保証 ➜ 「形式検証」

形式検証の例１：定理証明

* Example: program verification by Hoare logic
  * Proof search can be automatic or interactive
    (proof assistants like Coq, Agda, PVS, Isabelle, …)
  * A mathematical proof guarantees correctness
    * Unlike testing (empirical guarantee)
  * Can cover infinitely many inputs/environments (in principle)
    * Write “let ***i*** be the input” in the proof!



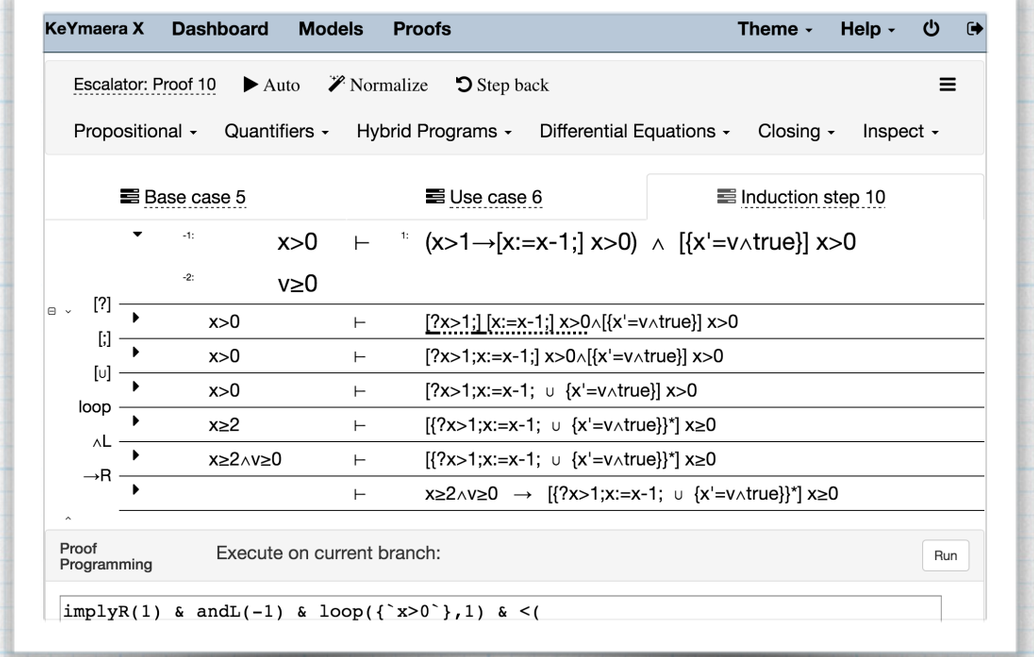

```
Theorem forall_exists : (forall P : Set->Prop,
          (forall x, ~(P x)) -> ~(exists x, P x)).
Proof.
  intros P.
  intros forall_x_not_Px.
  unfold not.
  intros exists_x_Px.
  destruct exists_x_Px as [ witness proof_of_Pwitness].
  pose (not_Pwitness := forall_x_not_Px witness).
  unfold not in not_Pwitness.
  pose (proof_of_False := not_Pwitness proof_of_Pwitness).
  case proof_of_False.
Qed.
```

A Coq proof of ∀x. ¬P(x) → ¬ ∃x. P(x)

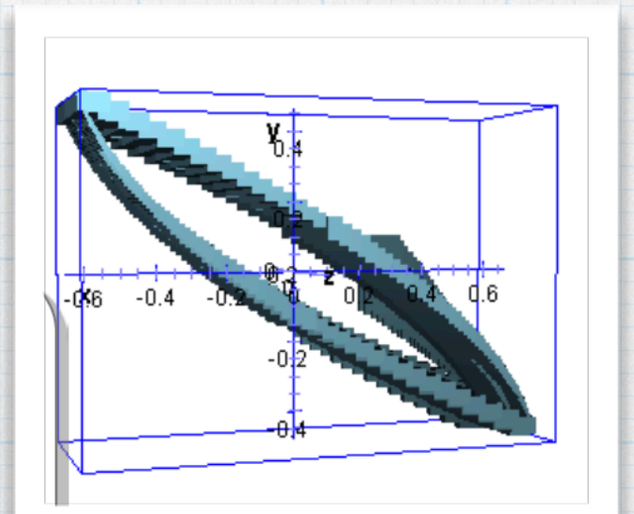
形式検証の例１：定理証明

* 物理情報システムに適用可能な例：
  differential dynamic logic [Platzer]
  * André Platzer: Logical Foundations of Cyber-Physical Systems. Springer 2018
  * 本質的に： Hoare 論理＋微分方程式
  * ツール： KeYmaera X
    * 自動定理証明系として（自動証明モード）
    * 証明支援系として（対話証明モード）

* 実用上のチャレンジ： 論理体系が代数的
  * 厳密で，単純なホワイトボックスモデルが必要
    ➜ 比較的単純なシステムの検証に応用が限られる
  * ブラックボックスコンポーネントや，外乱など，
    不確かさのモデリングが容易ではない



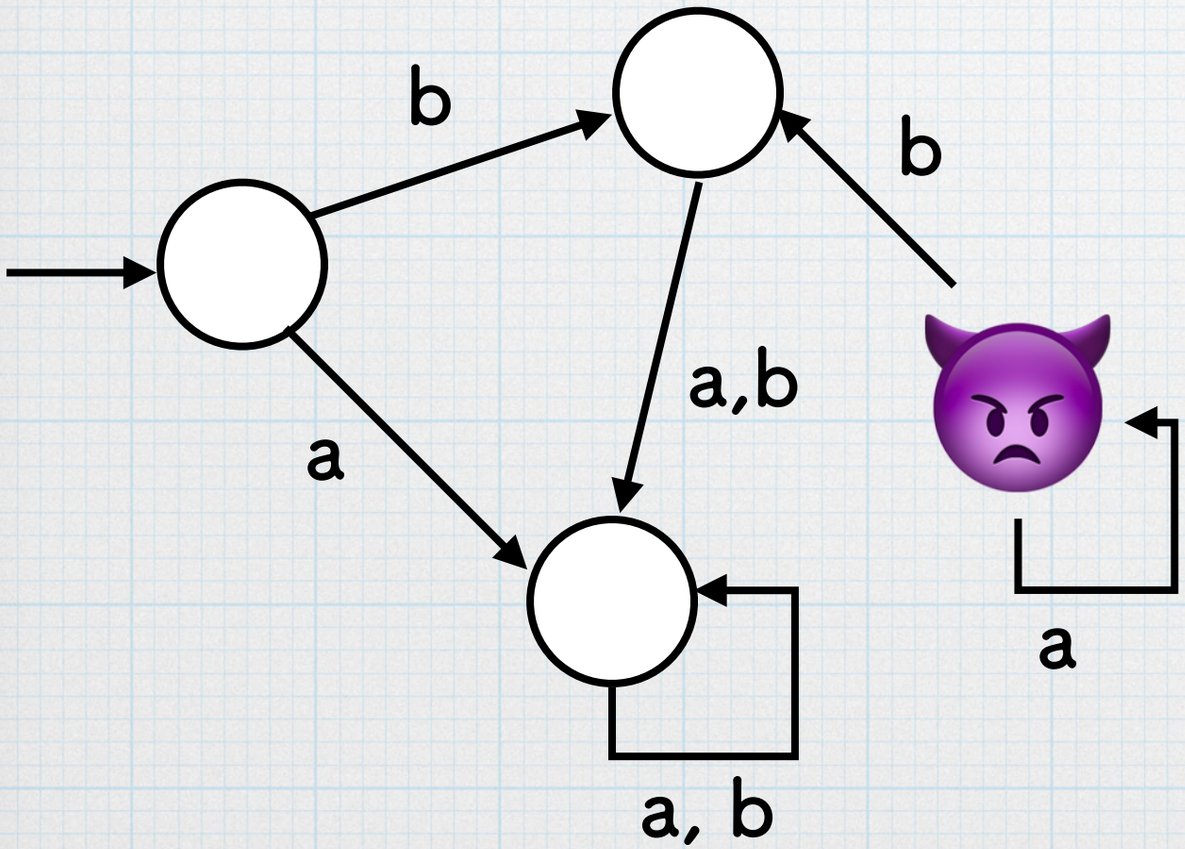
KeYmaera X スクリーンショット



参考： SpaceEx スクリーンショット
制度保証付き数値計算による可達性解析

形式検証の例２： モデル検査

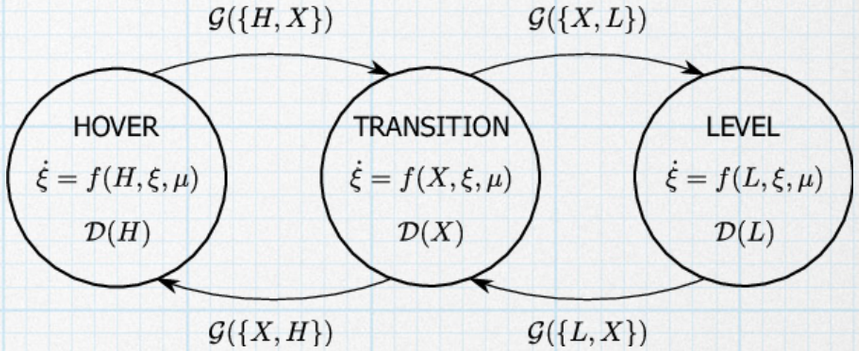
* オートマトンのアルゴリズムによる「自動証明」
  * 主にグラフの到達可能性判定に帰着
* オートマトンは有限 ➜ 数え上げによる自動証明が可能
* 例： 以下のオートマトンにて，😈 に至ることはない (*)



* （不正解）
  仕様 (*) をすべての入力について確かめる
* （正解）
  到達可能な領域を計算し（グラフ探索，有限時間で飽和），😈が含まれないことを確かめる

形式検証の例２： モデル検査

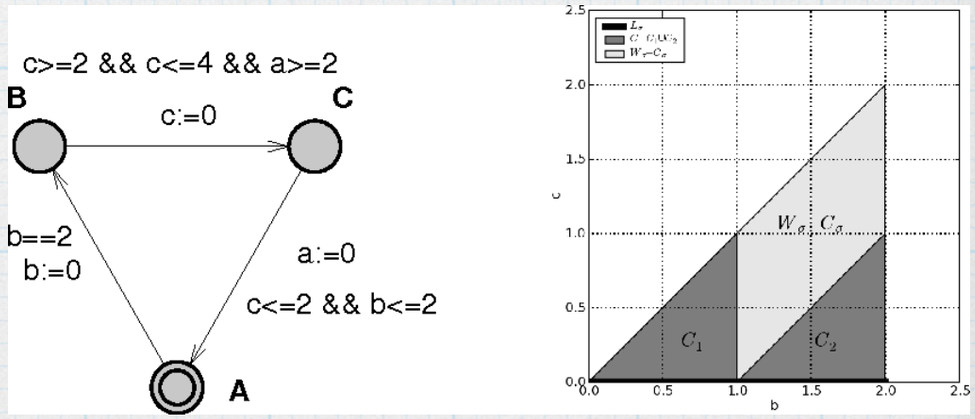
* 物理情報システムへの適用
  * 一義的には，連続量が現れた時点でアウト
    * 状態空間が無限になって，探索できない
    * 例： ハイブリッド・オートマトン
      （オートマトン＋微分方程式）では，到達可能性が決定不能



ハイブリッド・オートマトンの例
from [Casau+, CDC'11]

* 確率的オートマトンならばOK
  * Markov chain, Markov decision process, …
  * 到達可能性判定の代わりに，到達確率を計算する
    （線形計画法で解ける）

* 時間付きオートマトン timed automaton も OK
  * 連続量の領域が，決まったテンプレート
    （region, zone）で記述できる
  * region, zone は有限個しかない



時間付きオートマトンと zone graph
from [Daws+, FORMATS'06]

まず最初に：

ソフトウェア品質保証におけるスペクトル

形式手法

テスト test | モデル検査 | 自動定理証明 | 対話定理証明

低 ← コスト → 高

高 ← 自動化と合い → 低

自動

専門家を月・年単位で拘束

低 ← 保証の網羅性 → 高

低 ← 保証の数学的厳密性 → 高

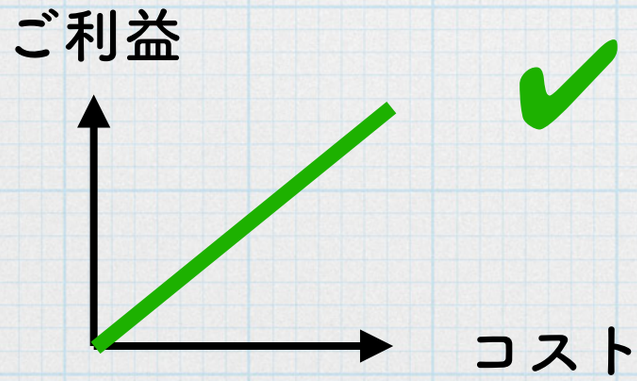
経験論的保証 | 数学的保証 ➜ 「形式検証」

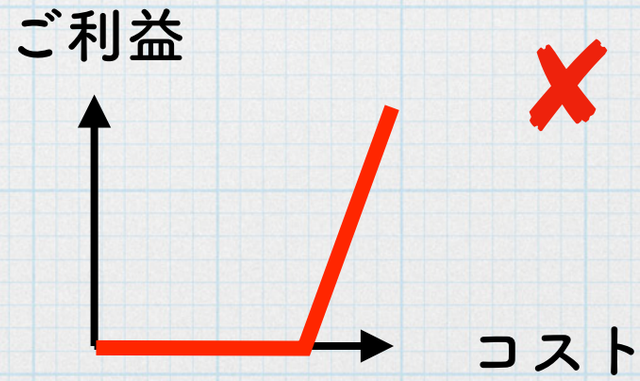
自動運転には？？

自動運転のための形式手法

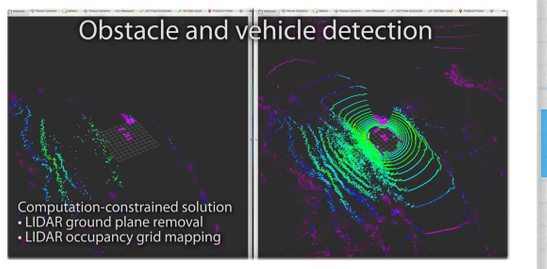
* 課題１：ブラックボックスコンポーネント
  * 中身がわからない ➜ モデルが立たない
    ➜ 形式検証は不可能
    （または「なんでもやらかし得る
    コンポーネント」とモデリング）

  * テストは可能（入手力の対応を観察）
  * 逆に，形式検証のためにはモデルが必須
    ➜ 形式検証は「スケールダウン不可」

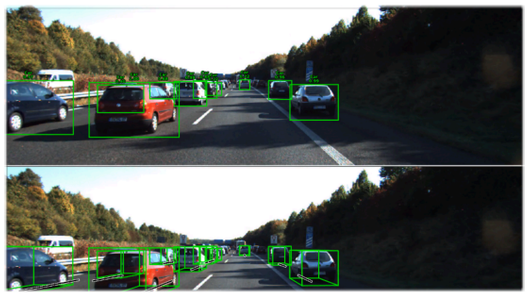




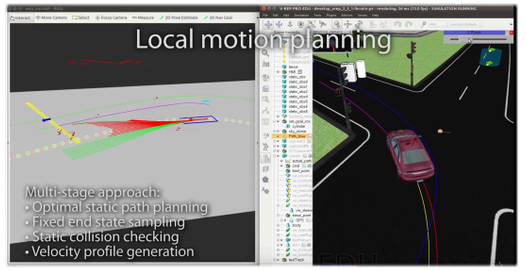
自動運転のための形式手法



知覚



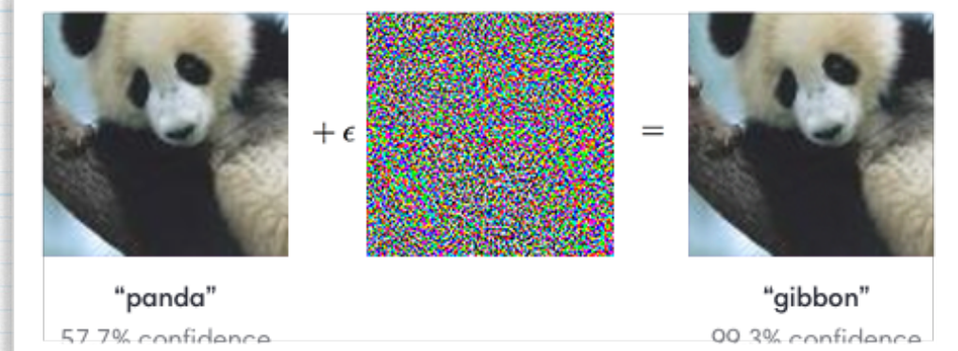
物体認識



判断，経路生成

操作，経路追跡

* 課題２： 機械学習コンポーネント
  * 自動運転システムには深層学習が必須.
    特に物体認識
  * しかし，深層学習は
    * 正しい結論を下すとは限らない.
      また，間違いの予知が困難
    * 判断の理由が説明しづらい
      (explainable AI)
  * ➜ 形式検証の数学的・論理的厳密性との兼ね合いは？？



https://openai.com/blog/adversarial-example-research/

(a) Husky classified as wolf (b) Explanation

Figure 11: Raw data and explanation of a bad model's prediction in the “Husky vs Wolf” task.

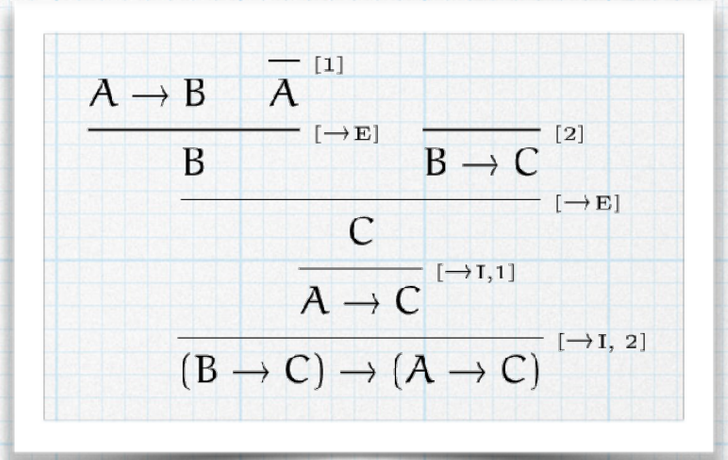
	Before	After
Trusted the bad model	10 out of 27	3 out of 27
Snow as a potential feature	12 out of 27	25 out of 27

Table 2: “Husky vs Wolf” experiment results.

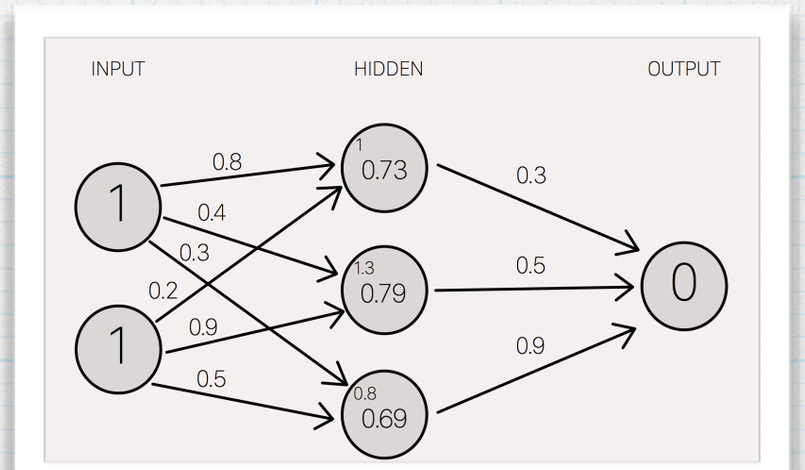
arXiv:1602.04938

ところで：AI 研究の歴史（超ダイジェスト）

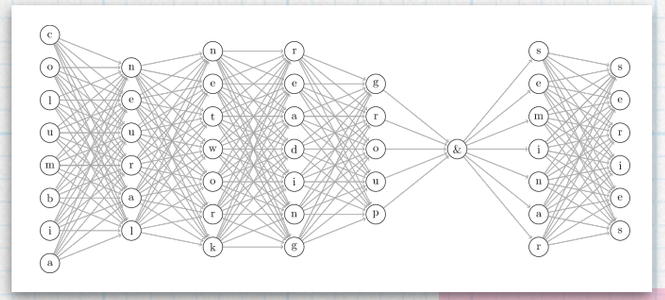
* 1980-1987「論理的人工知能，エキスパートシステム」
  * 与えられたルールによる論理的推論
    * 判断の過程が明らかにトレースできる
    * 「与えられたルール」の準備が大変（不可能）
  * Prolog, 第５世代コンピュータ
  * 関連する数学分野： 論理学，数学基礎論



* 2011-現在「統計的機械学習」
  * 大量のデータから自動で統計的に特徴量抽出
    * 驚くべきスケーラビリティ．
      画像処理，物体認識，自動翻訳，…
    * 判断の過程はブラックボックス
  * ディープニューラルネットワーク
  * 関連する数学分野： 統計学，最適化



統計的機械学習 vs 演繹的形式推論



$$\frac{A \quad A \supset B}{B}$$

統計的機械学習		演繹的形式推論
データのノイズを許容	入力の誤り	公理は絶対誤りは想定せず
保証されない	結論の正しさ	論理的に保証（cf. 数学的証明）
高い データから自動で特徴量発見	スケーラビリティ	低い 公理の準備は人力（cf. エキスパートシステム）
低い 判断の理由はパラメータ（重み）	説明可能性	高い 推論過程が証明として明示的

ERATO MMSD の目指すもの

これらの最適な組み合わせ

形式手法

テスト test | モデル検査 | 自動定理証明 | 対話定理証明

低 ← コスト → 高

＋ 統計的機械学習

高 ← 自動化の度合い → 低

自動 | 専門家を月・年単位で拘束

低 ← 保証の網羅性 → 高

低 ← 保証の数学的厳密性 → 高

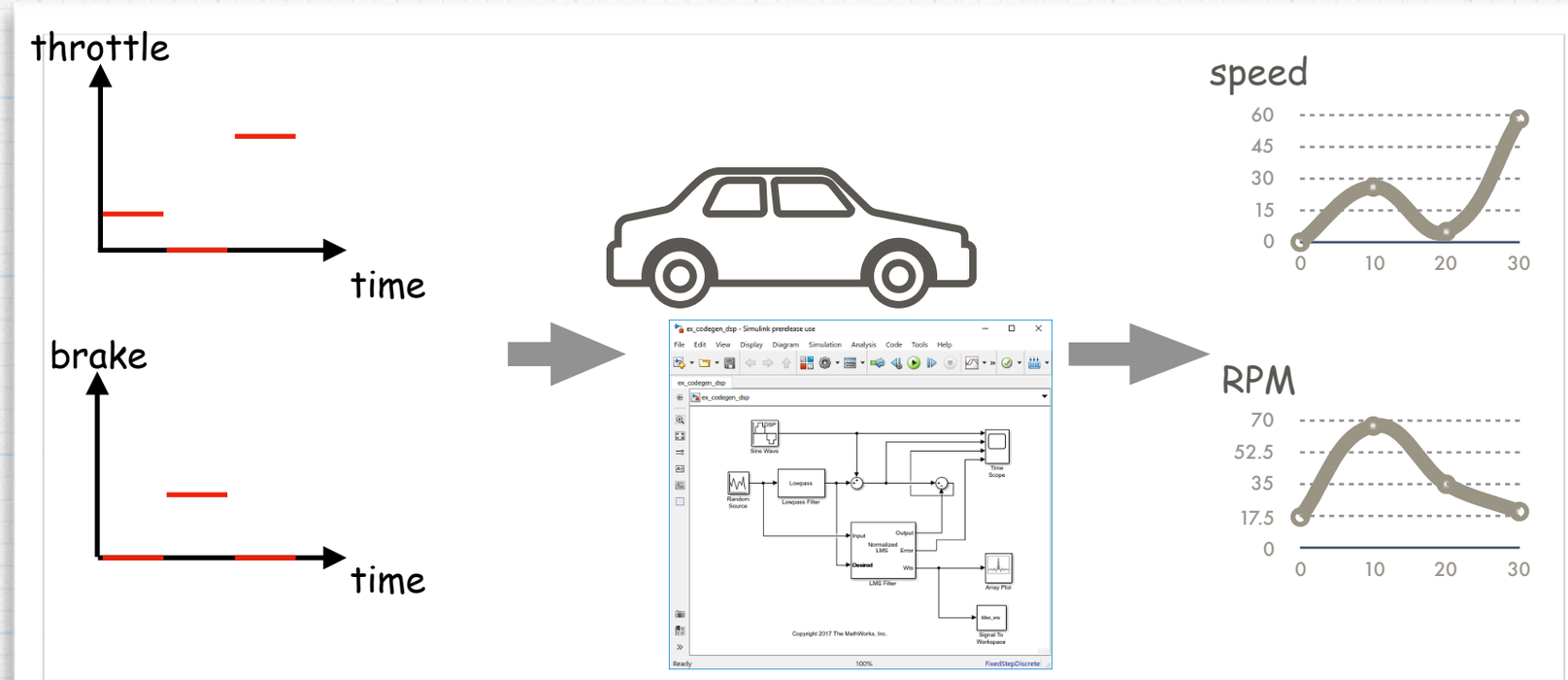
経験論的保証 | 数学的保証 ➜ 「形式検証」

アウトライン

- ソフトウェア品質保証におけるスペクトル
  - ERATO MMSD の目指すもの：
    テストと形式検証の最適な組み合わせ
- 具体的技術シーズの紹介
  - サーチベーステスト
  - 自動運転システムのテストシナリオ記述・生成
  - 形式的安全アーキテクチャ
  - 実行時監視アルゴリズム
  - ニューラルネットのオートマトン近似
  - 自動運転システムの判断・経路生成アルゴリズム
  - 形式仕様記述を支援する対話型ツール
  - 数理的理論基盤

強化学習によるサーチベーステスト：Simulink モデルの反例生成

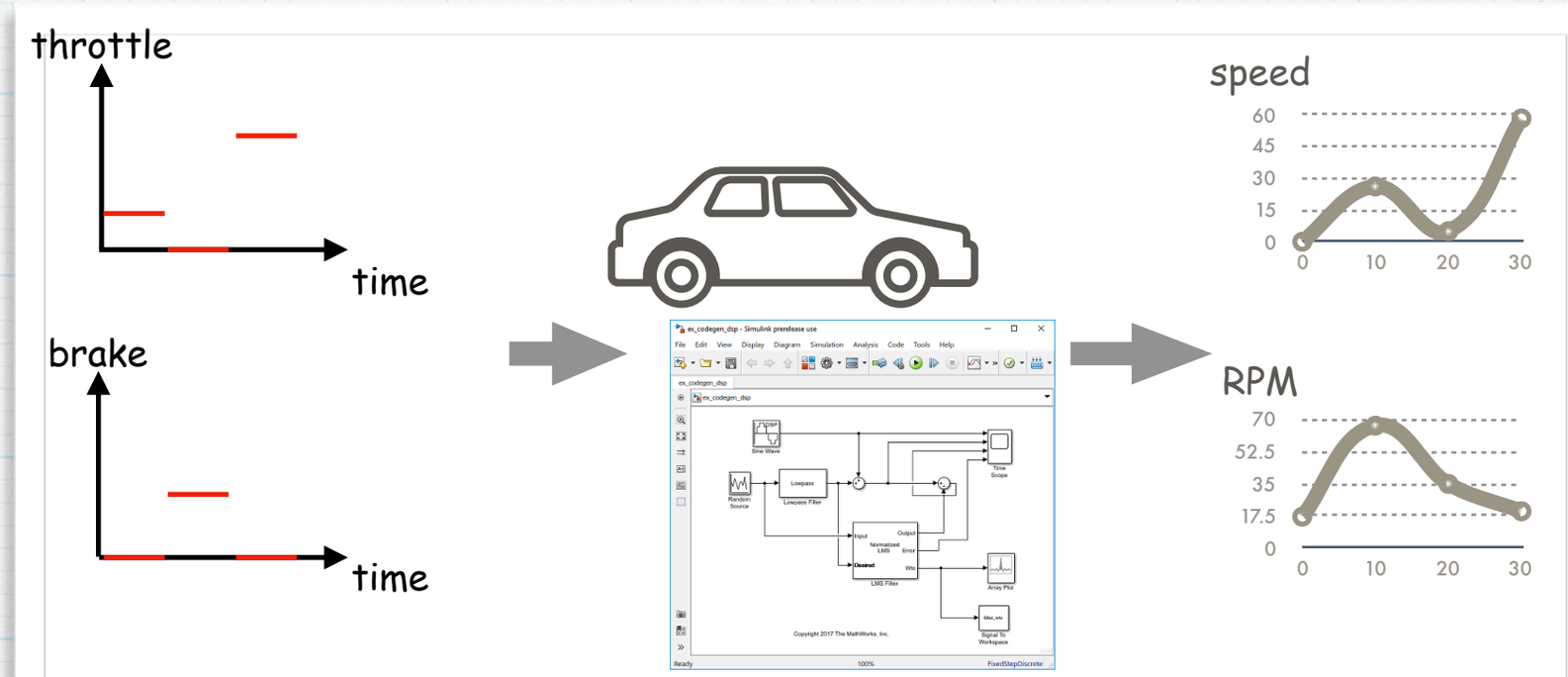
[Akazaki & Hasuo, CAV'15]
[Zhang, Ernst, Sedwards, Arcaini & Hasuo, EMSOFT'18]
[Zhang, Hasuo & Arcaini, CAV'19] …



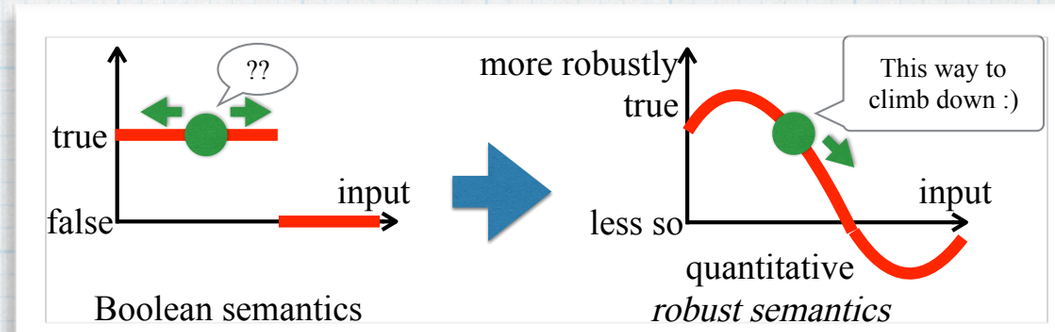
* ブラックボックステスト．
入出力の対応を見ながら，反例となる入力シグナルを探索
  * Given: Simulink モデル M, 時相論理仕様 ϕ
    * 仕様 ϕ の例：
      ギヤが4速になったらそれから3秒以内に時速 50 km/h 以上になる
  * 目標：
    出力シグナル M(i) が ϕ を
    みたさないような，入力シグナル i

強化学習によるサーチベーステスト：Simulink モデルの反例生成

[Akazaki & Hasuo, CAV'15]
[Zhang, Ernst, Sedwards, Arcaini & Hasuo, EMSOFT'18]
[Zhang, Hasuo & Arcaini, CAV'19] …



* 主流のアプローチ [Fainekos & Pappas, TCS'09]：
  * 強化学習，確率的最適化で「あぶないところ」を狙う
  * そのため，時相論理式 ϕ の真偽値を，ブール値（2値）から連続実数値に拡張 ➔ 勾配降下法



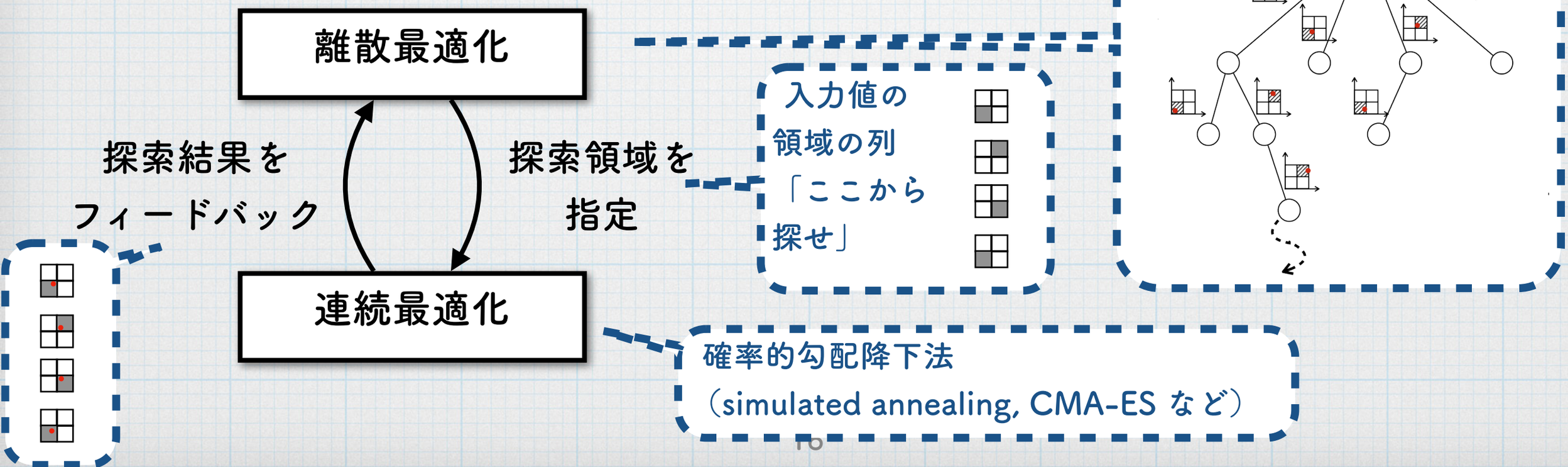
強化学習によるサーチベーステスト：Simulink モデルの反例生成

[Akazaki & Hasuo, CAV'15]
[Zhang, Ernst, Sedwards, Arcaini & Hasuo, EMSOFT'18]
[Zhang, Hasuo & Arcaini, CAV'19] …

* Survey: [Kapinski+, IEEE Control Syst. '16]

  J. Kapinski, J. V. Deshmukh, X. Jin, H. Ito, and K. Butts, "Simulation-based approaches for verification of embedded control systems: An overview of traditional and advanced modeling, testing, and verification techniques," IEEE Control Syst., vol. 36, no. 6, pp. 45–64, Dec. 2016.

* ERATO MMSD の貢献 [Zhang+, EMSOFT'18] を例に． [Zhang+, CAV'19] も同種のアイデア
  * 確率的勾配降下法における離散構造の利用，
    これによる反例発見能力の向上
  * 階層的最適化アーキテクチャ



ERAT
ERATO
国立情報学研究所

強化学習によるサーチベー
Simulink モデルの反例生成

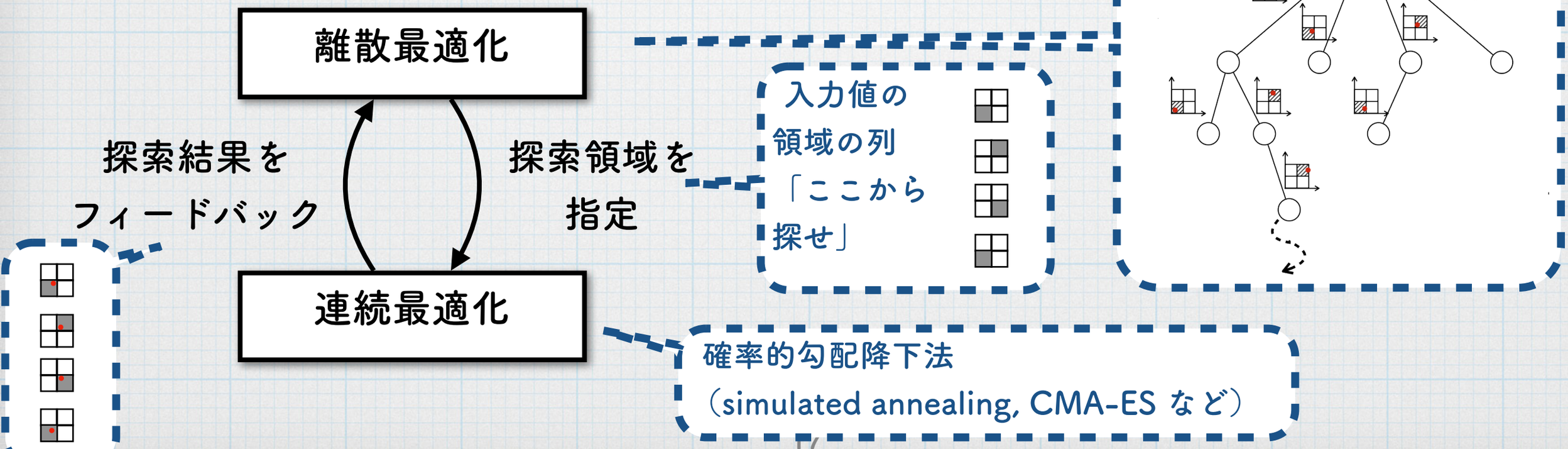
ポスター

* Kapinski+, IEEE Control
  eshmukh, X. Jin, H. Ito, and K. Butts, "Simu
  traditional and advanced modeling, testing, and verification
* ERATO MMSD の貢献 [Zhang+, EM
  * 確率的勾配降下法における離
    これによる反例発見能力の向
  * 階層的最適化アーキテクチ

- 品質保証の問題を，
  論理式で記述された形式仕様を用いて
  最適化問題に帰着
- 最適化の際にも，もともとの離散構造を
  利用

→

論理と統計の協働，
形式検証とテストの協働



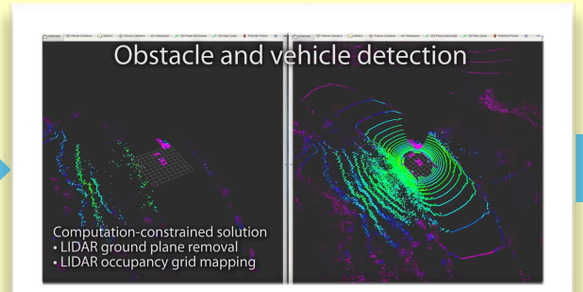
自動運転システムのテストシナリオ記述・生成

環境入力

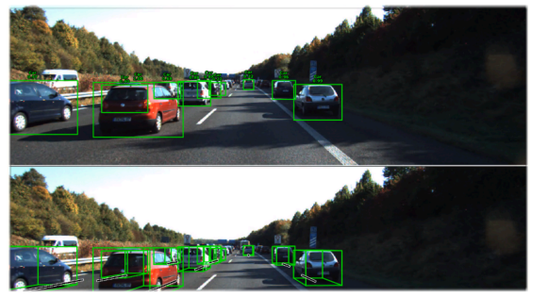
（路形，歩行者の位置，他車位置，…）

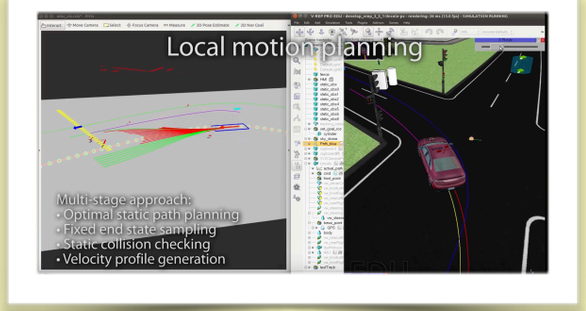
自車位置，状態

（知覚）



物体認識





操作，経路追跡

判断，経路生成

シミュレーション環境

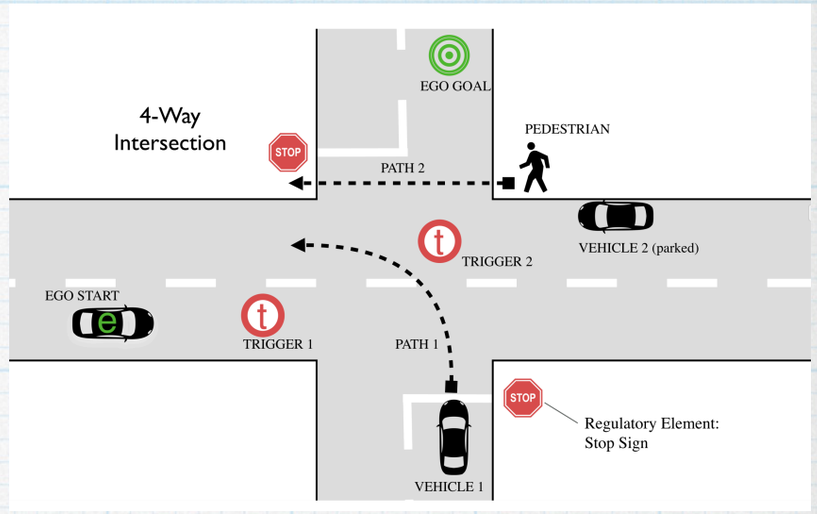
（多くはゲームエンジン使用）

* 意図した状況を実現する環境入力を多数生成したい
  （歩行者とニアミス，など）
* シミュレーションは決して速くない（～実時間）
  ➜ あてずっぽではダメ

自動運転システムのテストシナリオ記述・生成

* アイデア： 高位シナリオから，低位シナリオへの翻訳
  * 高位シナリオ，機能シナリオ，意図
    「歩行者とニアミスする状況を見たい」
  * 低位シナリオ，具体シナリオ
    「歩行者を座標 (x0, y0) に配置，歩行速度は (vx, vy)」
* 必要な技術
  * 高位・低位のシナリオの記述言語
    * autonomoose は低位シナリオ GeoScenario を独自に開発 [Queiroz+, IV'19]
  * 高位から低位への翻訳・コンパイリング・サンプリングアルゴリズム

自動運転システムのテストシナリオ記述・生成



[Queiroz+, IV'19]

* アイデア： 高位シナリオから，低位シナリオへの翻訳
  * 高位シナリオ，機能シナリオ，意図
    「歩行者とニアミスする状況を見たい」
  * 低位シナリオ，具体シナリオ
    「歩行者を座標 (x0, y0) に配置，歩行速度は (vx, vy)」
* アプローチ
  * データ駆動型（ほしいのはレアイベントなので，単純な検索ではダメ）
  * ベイズ推論 ➜ 確率的プログラム．ERATO MMSD で進行中

自動運転システムのテストシナリオ記述・生成
—— 確率的プログラムによるアプローチ

* 確率的プログラム =
  プログラム
  ＋ 確率的分岐，サンプリング
  (ifp, x ~ Unif (…))
  ＋ 観測 (observe)

```
1   double x,y := 0;
2   ifp (0.05) then
3    x ~ Unif(0,1);
4   else
5    x ~ Unif(-1,0);
6   ifp (0.05) then
7    y ~ Unif(0,1);
8   else
9    y ~ Unif(-1,0);
10  observe(x + y >= 1.9);
11  return(x)
```

* 求めたいのは，return value (x, 11行目)
  の（ベイズの意味での）事後分布

* ナイーブな計算方法： rejection sampling
  * 観測が rare event の場合効率悪化

* SMC, MCMC などのサンプリング手法が用いられる.
  確率的プログラム処理系： Anglican, Stan, Venture など
  サーベイ：[Gordon, Henzinger Nori & Rajamani, FOSE'14]

自動運転システムのテストシナリオ記述・生成
—— 確率的プログラムによるアプローチ

もとのプログラム	観測条件の back propagation	効率化されたプログラム
1 x ∼ Unif(0,20);	// no knowledge	x ∼ Unif(7,10);
2 obs(x < 10);	7 < x < 10, x ∼ Unif(0,20)	// no observation
3 y ∼ Beta(1,1);	7 < x < 10	y ∼ Beta(1,1);
4	8 < x + y < 10, 0 ≤ y < 1	obs(8 < x + y < 10);
5 x := x + y;	8 < x + y < 10	x := x + y;
6 obs(x < 10);	8 < x < 10	// no observation
7 y ∼ Beta(1,1);	8 < x < 10	y ∼ Beta(1,1);
8	9 < x + y < 10, 0 ≤ y < 1	obs(9 < x + y < 10);
9 x := x + y;	9 < x + y < 10	x := x + y;
10 obs(x < 10);	9 < x < 10	// no observation
11 y ∼ Beta(1,1);	9 < x	y ∼ Beta(1,1);
12	10 ≤ x + y, 0 ≤ y < 1	obs(10 ≤ x+y);
13 x := x + y;	10 ≤ x + y	x := x + y;
14 obs(10 ≤ x);	10 ≤ x	// no observation

* 確率的プログラミング言語処理系 R2 [Nori+, AAAI'14]
  * プログラム論理によって観測条件を持ち上げる(back propagation)
  * 結果，sample rejection の起こりにくい，より効率的なプログラムへ
* ERATO MMSD の取り組み： この手法の改良，テストシナリオ生成への応用

ERATO
ERATO
国立情報学研究所

自動運転システムのテ
—— 確率的プログラム

- テストシナリオ生成問題を，
  確率的プログラミング言語の問題に帰着
- ベイズ推論とプログラム論理

➜

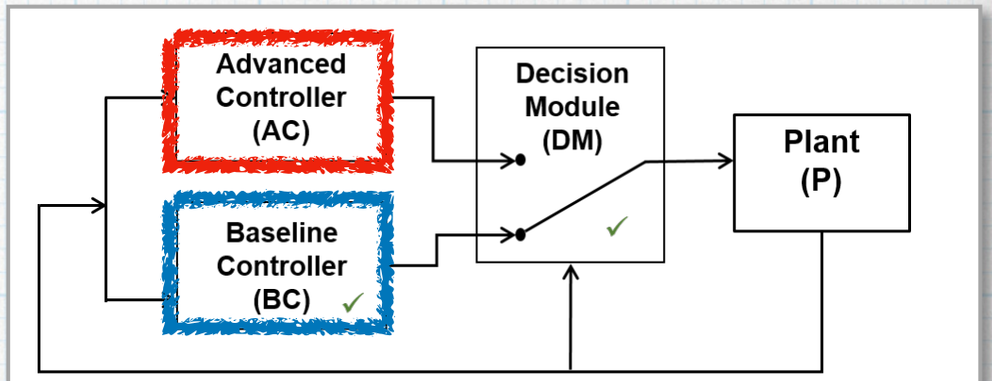
論理と統計の協働，
形式検証とテストの協働

もとのプログラム	観測条	
1 x ∼ Unif(0,20);	// no kn	
2 obs(x < 10);	7 < x <	
3 y ∼ Beta(1,1);	7 < x <	
4	8 < x <	
5 x := x + y;	8 < x <	
6 obs(x < 10);	8 < x <	
7 y ∼ Beta(1,1);	8 < x <	
8	9 < x <	
9 x := x + y;	9 < x	
10 obs(x < 10);	9 < x < 10	// no observation
11 y ∼ Beta(1,1);	9 < x	y ∼ Beta(1,1);
12	10 ≤ x + y, 0 ≤ y < 1	obs(10 ≤ x+y);
13 x := x + y;	10 ≤ x + y	x := x + y;
14 obs(10 ≤ x);	10 ≤ x	// no observation

* 確率的プログラミング言語処理系 R2 [Nori+, AAAI'14]
  * プログラム論理によって観測条件を持ち上げる(back propagation)
  * 結果，sample rejection の起こりにくい，より効率的なプログラムへ
* ERATO MMSD の取り組み： この手法の改良，テストシナリオ生成への応用

形式安全アーキテクチャ

* 安全アーキテクチャの例：
  simplex architecture （右図）
  * 通常系（AC）は
    複雑，性能重視，ブラックボックス
  * 保証系（BC）は
    単純，安全性重視，（できるだけ）ホワイトボックス



Phan et al., ACSD'17

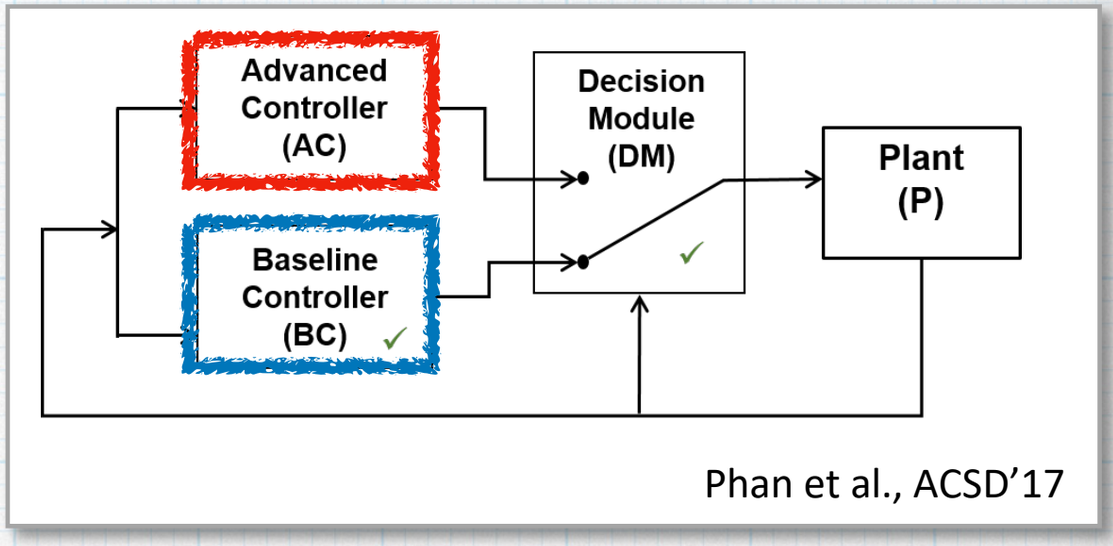
* ポイント： AC がブラックボックスであっても，システム全体を形式検証できる．
  BC の安全性 及び DM （スイッチング戦略）の正しさが示せれば十分

* 仮に形式検証が不可能でも，システム設計・テストの系統的指針を与える

* 留意点
  * 一般に： 冗長性，独立性，多様性，…
  * 自動運転に特化して： 制御部の構造（物体認識 ➜ 経路生成 ➜ 経路追跡），
    フォールバック（路肩に止める）

形式安全アーキテクチャ

* ERATO MMSD の取り組み：
  * 安全アーキテクチャの Event-B による形式化

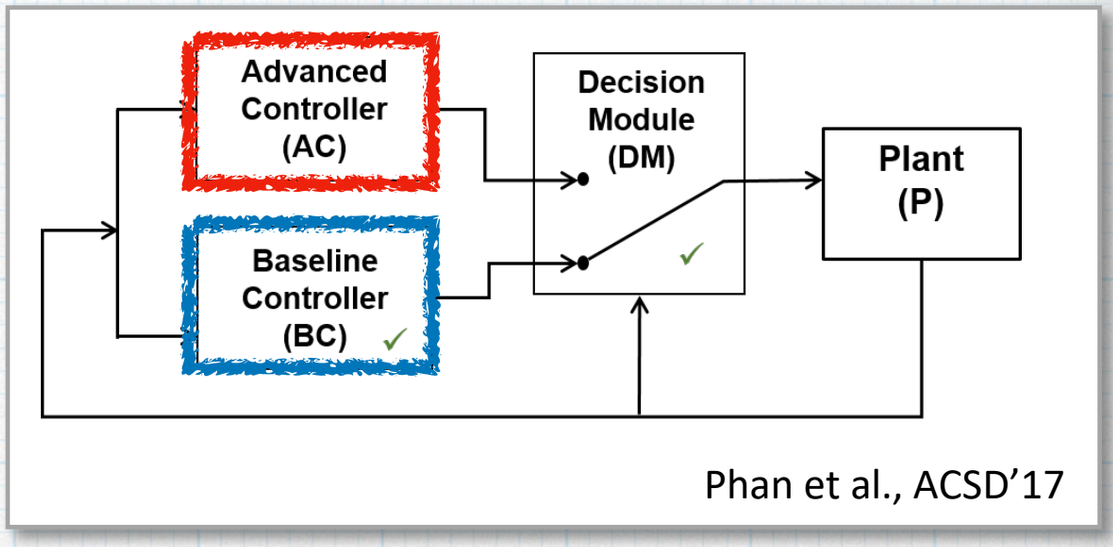


Phan et al., ACSD'17

* Event-B: 形式モデリングツール
  [Abrial, "The Event-B Book", 2010 CUP] [Kobayashi+, ICFEM'18]など
  状態遷移系を基礎とする．次をサポート
  * 安全性証明
    * SMT ソルバによる自動証明など
    * コンポーネント間の契約に基づく，assume-guarantee 型推論 ➜ ブラックボックスを許容
    * 安全アーキテクチャの安全性・信頼性保証に使える
  * モデルの段階的詳細化をサポート
    * 個別のアーキテクチャの体系的導出に使える

形式安全アーキテクチャ

* ERATO MMSD の取り組み：
  * 安全アーキテクチャの Event-B による形式化

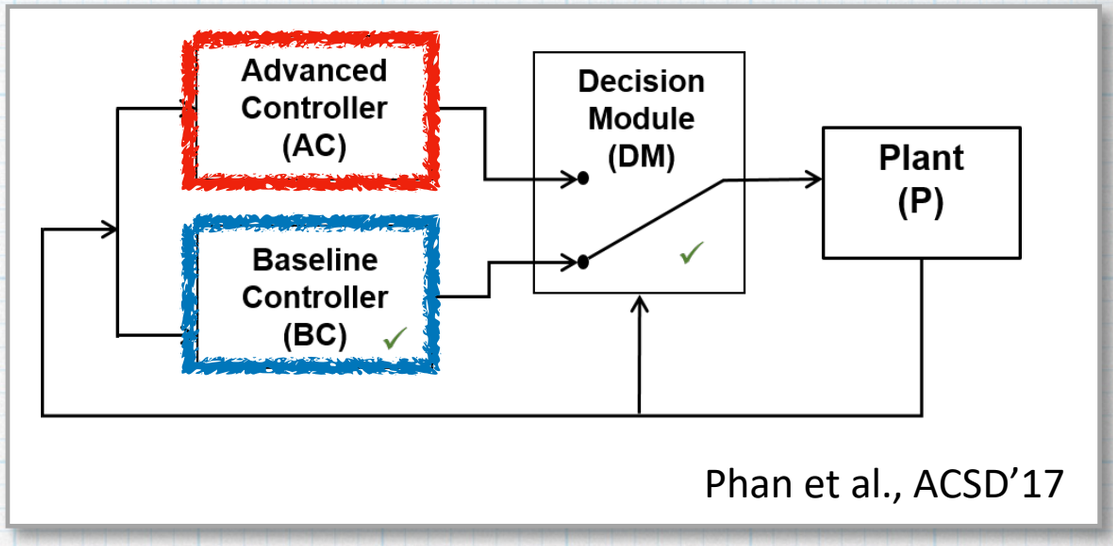


Phan et al., ACSD'17

* 現在の具体的取組み
  * 自動運転システムのための，一般的安全アーキテクチャを構築，その信頼性を確認
    （autonomoose と協働，自動運転独自のニーズを反映）
  * 一般的安全アーキテクチャから，企業さまの個別ニーズに応える個別安全アーキテクチャを導出

形式安全アーキテクチャ

* Event-B による取組みの優位性：
  モデルの詳細度を（ある程度）自由に選べる
  * 粗いモデル，弱い契約
    ➜ ミニマルな信頼性保証
  * 細かいモデル，強い契約
    ➜ 強い信頼性保証



Phan et al., ACSD'17

* 技術的課題：
  証明でなくテストも包含したい
  ➜ Event-B の定理証明系を，統計的尤度を許容するよう拡張

ERAT
ERATO
国立情報学研究所

形式安全アー

* Event-B による取組みの優
  モデルの詳細度を（ある程度
  に選べる
  * 粗いモデル，弱い契約
    ➜ ミニマルな信頼性保証
  * 細かいモデル，強い契約
    ➜ 強い信頼性保証

* 技術的課題：
  証明でなくテストも包含したい
  ➜ Event-B の定理証明系を，統計的
  尤度を許容するよう拡張

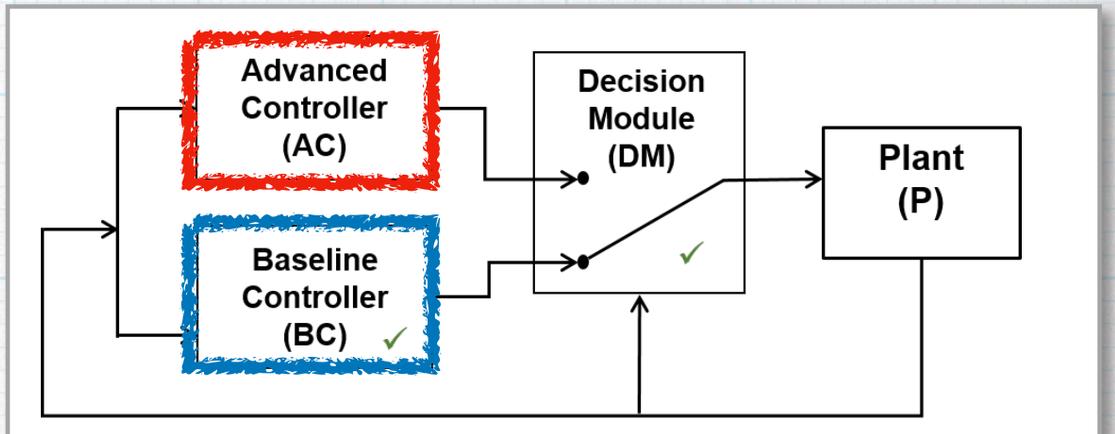
- Event-B の段階的詳細化機能，
  さらに（来るべき）「ファジー化」を
  通じて，システム形式検証を
  ダウンスケール可能に
- 「戦場を注意深く選ぶ」

➜

論理と統計の協働，
形式検証とテストの協働

実行時監視

* 設計時 V&V に加えて，実行時監視も重要
* 例１： 安全アーキテクチャの DM（スイッチング戦略）が，現在のシステムの状況に依存
* 例２： 安全アーキテクチャの安全性証明が何らかの契約（＝仮定）に依存している
  * 契約違反 ➜ もはや安全性保証なし ➜ フォールバック（路肩に止める）
* 設計時の形式検証に比べ，容易
  * ホワイトボックスモデルは必要なし



Phan et al., ACSD’17

実行時監視：さまざまな定式化

* Given: 離散時間ログ w = abaaacb…bbc
  仕様 ϕ 「b の後，6 step 待っても c が現れず」
  Answer: w の部分列で ϕ を満たすもの全体

* Given: 連続時間ログ w = (a, 0.12) (b, 1.28) …
  仕様 ϕ 「b の後，6 秒待っても c が現れず」
  Answer: w の部分列で ϕ を満たすもの全体

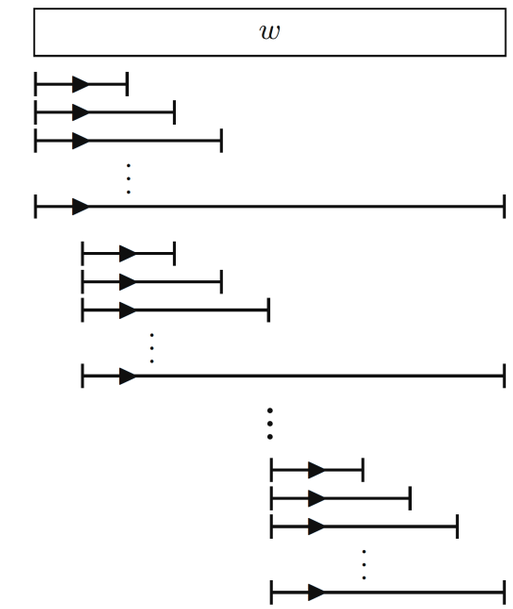
[Ulus, CAV'17] [Waga+, FORMATS'17] など

解は無限個
（開始時刻は t=1? t = 1.01? t = 1.001? …）
➜ 時間付きオートマトンの zone 構成で，効率的に表現・計算可能

* Given: 連続時間ログ w = (a, 0.12) (b, 1.28) …
  パラメータ付き仕様 $\phi(p)$ 「b の後，p 秒待っても c が現れず」「a は概ね p 秒周期で現れる」
  Answer: (p, {w の部分列で $\phi(p)$ を満たすもの}) のペア全体

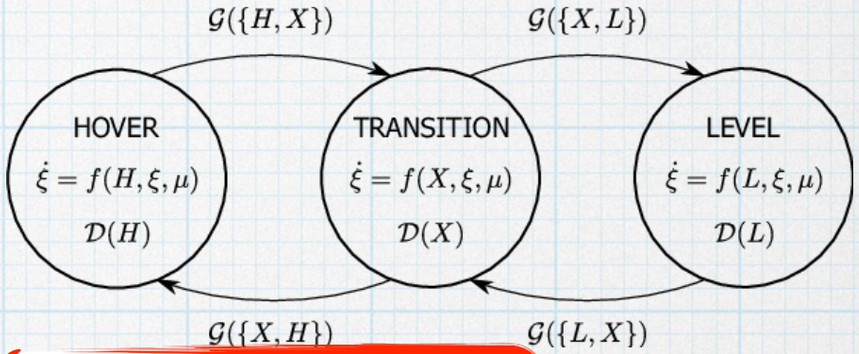
[Waga+, ICECCS'18] [Waga+, CAV'19]など

時間的因果関係があり，そう簡単ではない



形式検証の例２： モデル検査

* 物理情報システムへの適用
  * 一義的には，連続量が現れた時点でアウト
    * 状態空間が無限になって，探索できない
    * 例： ハイブリッド・オートマトン
      （オートマトン＋微分方程式）では，到達可能性が

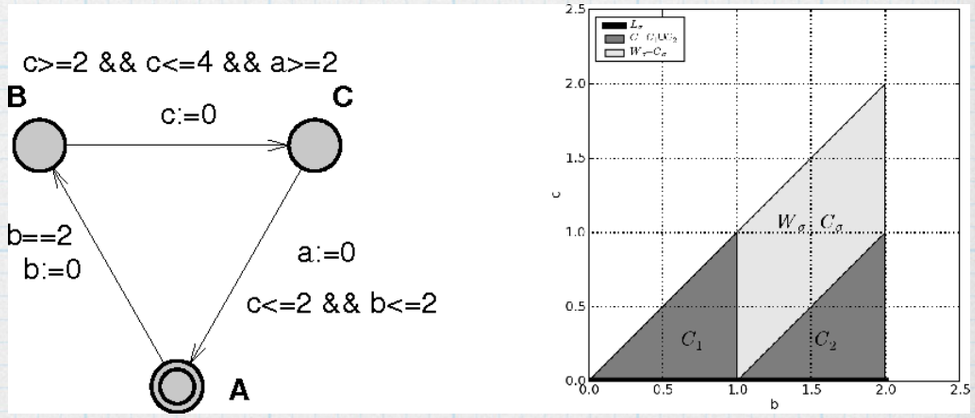


オートマトンの例
[Hu+, CDC’11]

・再掲

* 確率的オートマトンならばOK
  * Markov chain, Markov decision process, …
  * 到達可能性判定の代わりに，到達確率を計算する
    （線形計画法で解ける）

* 時間付きオートマトン timed automaton も OK
  * 連続量の領域が，決まったテンプレート
    （region, zone）で記述できる
  * region, zone は有限個しかない



時間付きオートマトンと zone graph
from [Daws+, FORMATS’06]

実行時監視： ERATO MMSD の取り組み

ポスター

* Given: 離散時間ログ w = abaaacb…bbc
  仕様 ϕ 「b の後，6 step 待っても c が現れず」
  Answer: w の部分列で ϕ を満たすもの全体

* Given: 連続時間ログ w = (a, 0.12) (b, 1.28) …
  仕様 ϕ 「b の後，6 秒待っても c が現れず」
  Answer: w の部分列で ϕ を満たすもの全体
  [Ulus, CAV'17] [Waga+, FORMATS'17] など

時間付きオートマトンの理論による高速アルゴリズム．ラップトップで毎秒，数M の長さのログを処理
[Waga+, FORMATS'17]
https://github.com/maswag/monaa
ルネサス RH850 実装もあり

パラメータ・時間付きオートマトンの理論による高速アルゴリズム．ラップトップで毎秒，数十 K の長さのログを処理
[Waga+, NFM'19]
https://github.com/maswag/symon

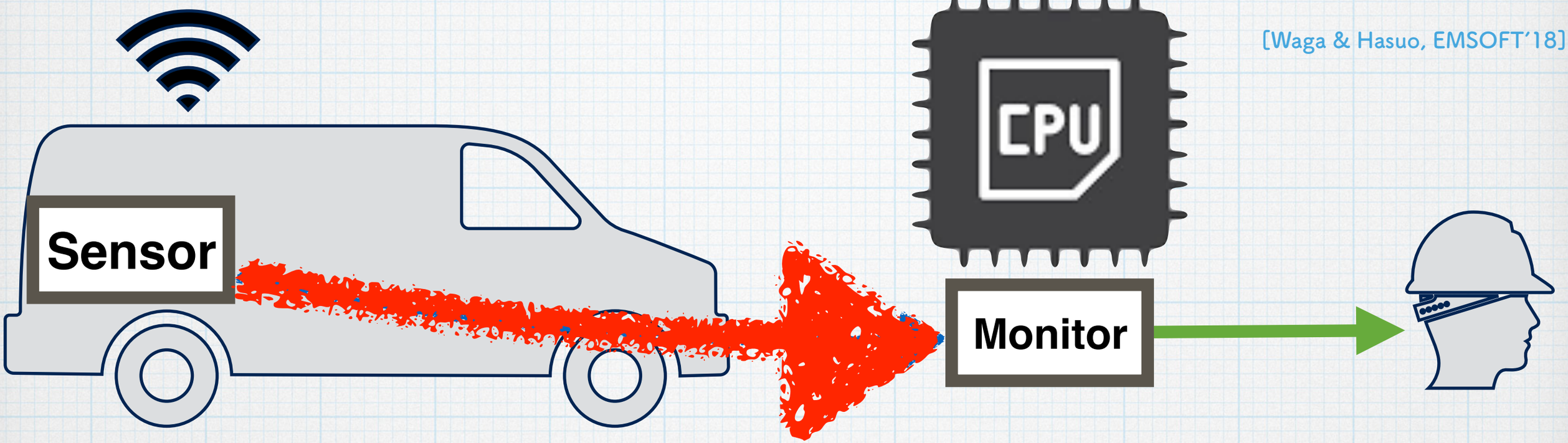
* Given: 連続時間ログ w = (a, 0.12) (b, 1.28) …
  パラメータ付き仕様 $\phi(p)$ 「b の後，p 秒待っても c が現れず」「a は概ね p 秒周期で現れる」
  Answer: (p, {w の部分列で $\phi(p)$ を満たすもの}) のペア全体
  [Waga+, ICECCS'18] [Waga+, NFM'19] [Waga+, CAV'19] など

ポスター

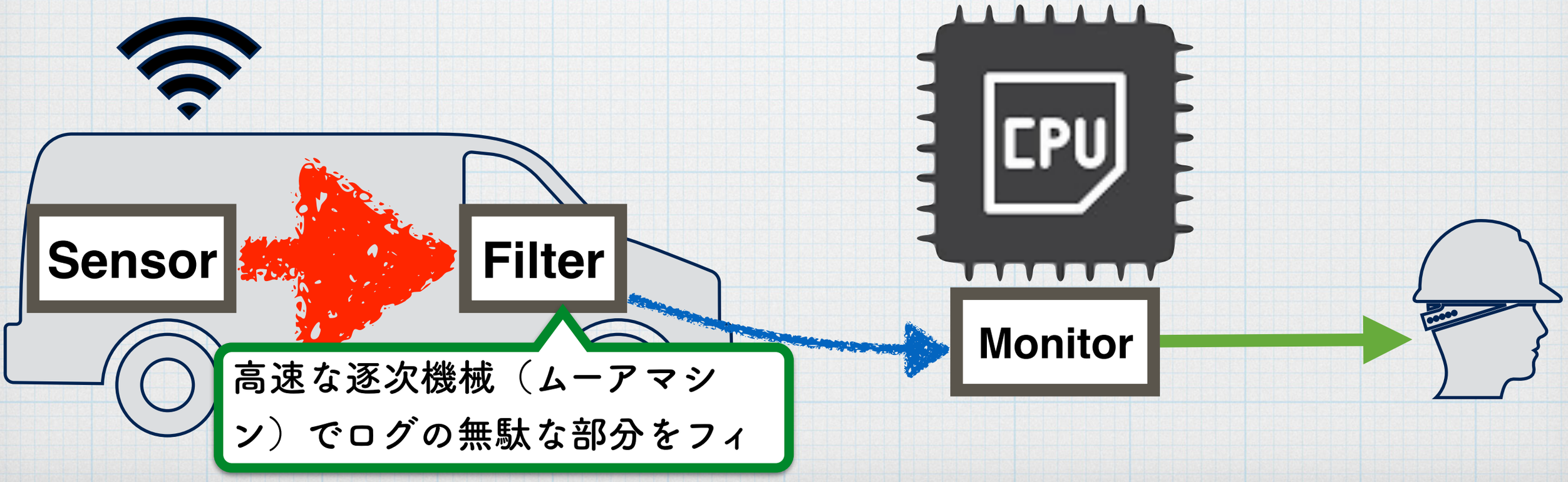
フィルタリングによる前処理

[Waga & Hasuo, EMSOFT'18]

Naive Remote Monitor



Contribution: Filter for Remote Monitor



理システムデザインプロジェクト
ics for Systems Design Project
nal Institute of Informatics & Japan Science and Technology Agency

の取り組み (1/

ERATO MMSD

- 形式検証のオートマトン的理論の目先を変えて，実用的問題に応用

➜

形式検証とテストの協働

[Ulus, CAV'17] [Waga+, FORMATS'17] など

時間付きオートマトンの理論による高速アルゴリズム．ラップトップで毎秒，数M の長さのログを処理
[Waga+, FORMATS'17]
https://github.com/maswag/monaa
ルネサス RH850 実装もあり

パラメータ・時間付きオートマトンの理論による高速アルゴリズム．ラップトップで毎秒，数十 K の長さのログを処理
[Waga+, NFM'19]
https://github.com/maswag/symon

* Given: 連続時間ログ w = (a, 0.12) (b, 1.28) …
  パラメータ付き仕様 $\phi(p)$ 「b の後，p 秒待っても c が現れず」「a は概ね p 秒周期で現れる」
  Answer: $(p, \{w$ の部分列で $\phi(p)$ を満たすもの$\})$ のペア全体

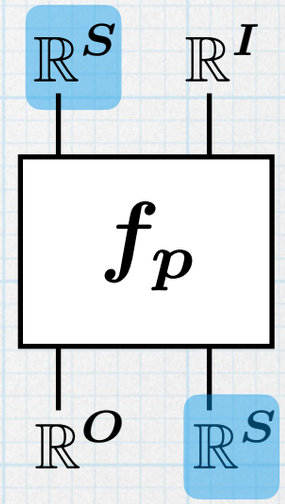
[Waga+, ICECCS'18] [Waga+, NFM'19] [Waga+, CAV'19]など

有限オートマトンによるRNNの近似

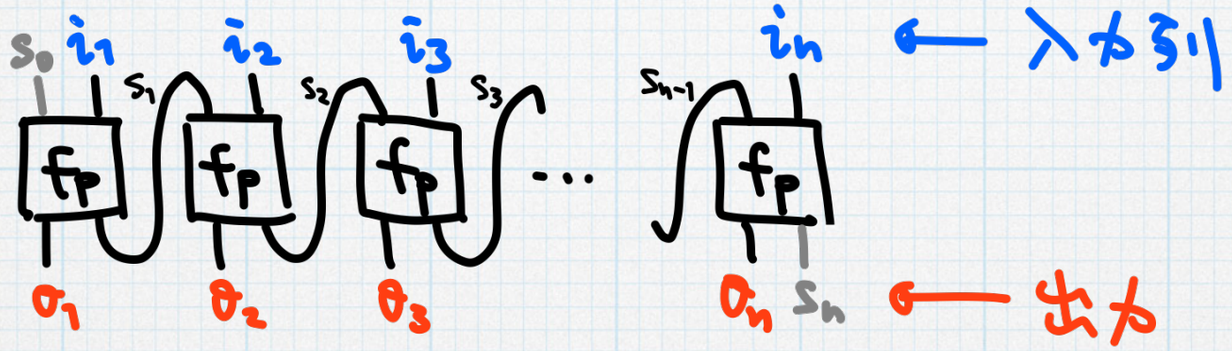
[Weiss, Goldberg & Yahav, ICML'18] [Okudono, Waga, Sekiyama & Hasuo, LearnAut'19. arXiv:1904.02931]

* Recurrent Neural Network (RNN)

$$\mathbb{R}^S \times \mathbb{R}^I \xrightarrow{f_p} \mathbb{R}^O \times \mathbb{R}^S$$



* 動作モード



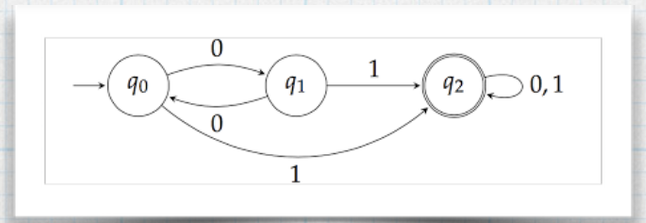
* 有限オートマトンを近似的に抽出

モデル検査など，形式手法を用いて安全性検証可能，「理解」可能

$$\mathbb{R}^S \times \mathbb{R}^I \xrightarrow{f_p} \mathbb{R}^O \times \mathbb{R}^S$$

$$Q \times I' \xrightarrow{g} O' \times Q$$

$(Q, I', O'$ are finite set$)$



- Angluin のオートマトン学習アルゴリズム (RNN は "gray box")
- ユークリッド空間のクラスタリング（およびその拡張）による状態空間抽象化

* 応用： 極端な入力の例示，RNNの実行時モニタリング

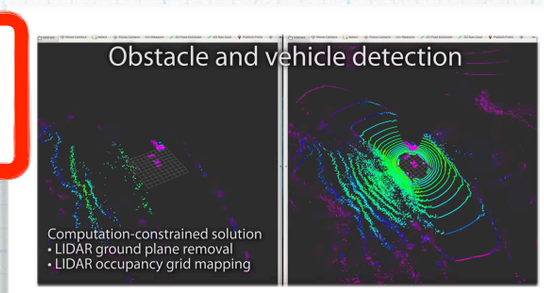
自動運転システムの判断，経路生成

* 制御理論チームを中心に，ソフトウェア工学・形式手法と協働しながら推進

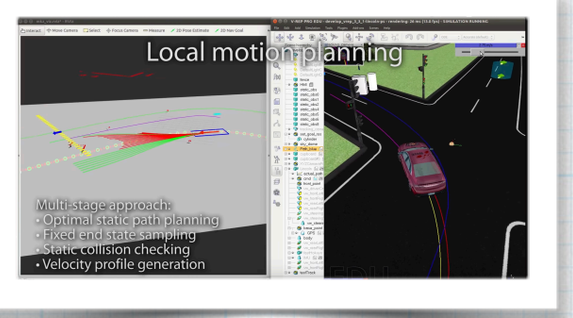
ポスター

* 取り組み1： 追跡可能性を考慮した経路生成 [Huang+, arXiv:1905.03444]
* 取り組み2：「安全性を保ちつつ，保守的になりすぎない」ためには
  * 他車が「なんでもやらかしうる」という仮定は現実的でない
  * ➜ 互いの意図を推測しながら意思決定，経路生成．
  * ゲーム理論の応用．Nash equilibrium の動的計画法による計算で，安全性向上 [Pruekprasert+, arXiv:1904.10158, arXiv:1904.06224]．

ポスター



操作，経路追跡



判断，経路生成

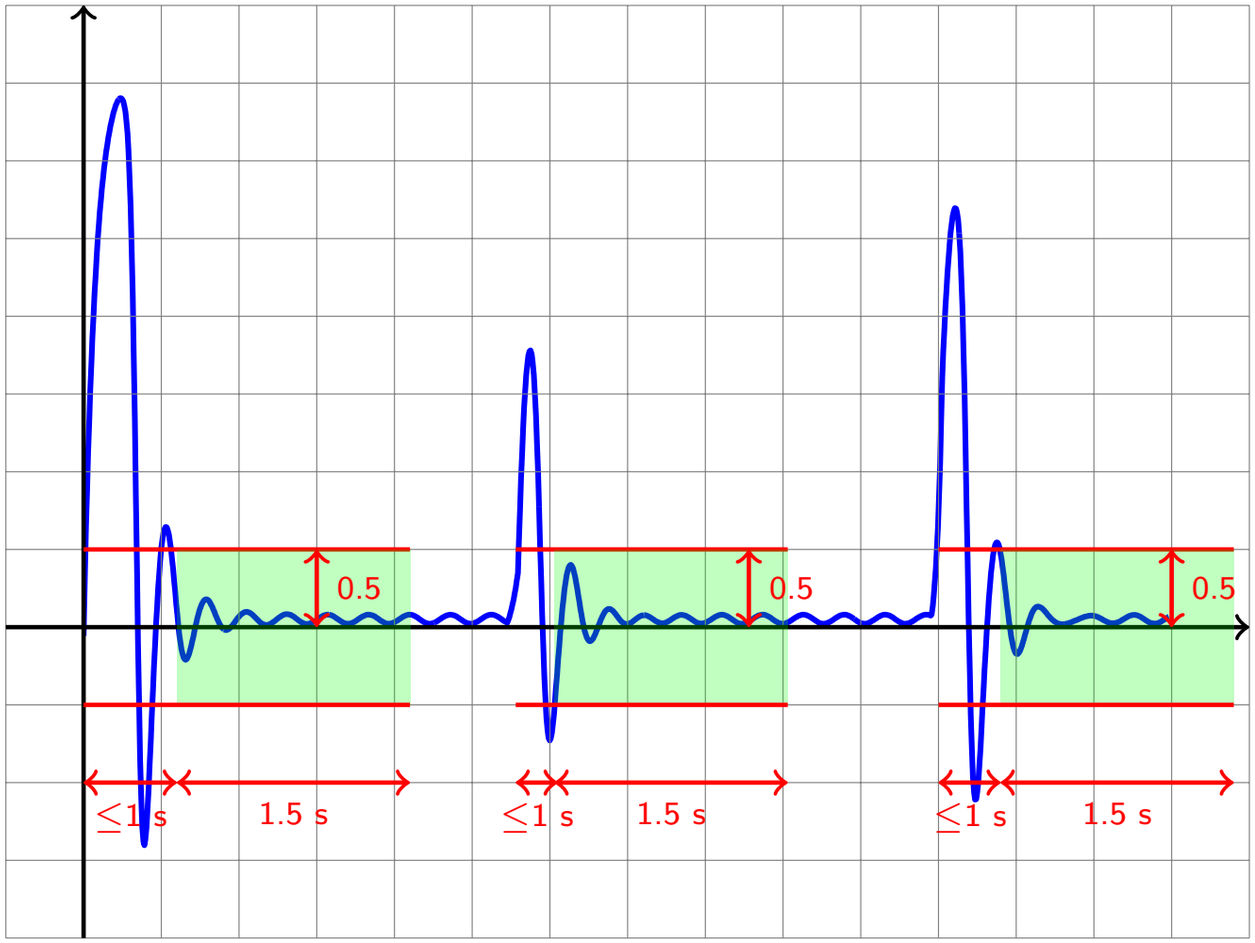
形式仕様記述を支援する対話型ツール

Alexandre Donze

STL Examples

Always $|x|>0.5 \Rightarrow$ *after 1 s,* $|x|$ *settles under 0.5 for 1.5 s*

$$\varphi := \mathsf{G}(x[t] > .5 \rightarrow \ \mathsf{F}_{[0,.6]} \ \ (\ \mathsf{G}_{[0,1.5]} \ x[t] < 0.5))$$



- いきなり正しいものを書くのは困難
- しかし，正解はエンジニアさんの頭の中！
- ➜
  形式仕様記述を支援する対話型ツール
  - 正例・反例を例示
    cf. [Prabhakar+, RTSS'18]
  - ユーザーのフィードバックを受けて，改訂をsuggest
- 研究推進中

アウトライン

- ソフトウェア品質保証におけるスペクトル
  - ERATO MMSD の目指すもの：
    テストと形式検証の最適な組み合わせ
- 具体的技術シーズの紹介
  - サーチベーステスト
  - 自動運転システムのテストシナリオ記述・生成
  - 形式的安全アーキテクチャ
  - 実行時監視アルゴリズム
  - ニューラルネットのオートマトン近似
  - 自動運転システムの判断・経路生成アルゴリズム
  - 形式仕様記述を支援する対話型ツール
  - 数理的理論基盤

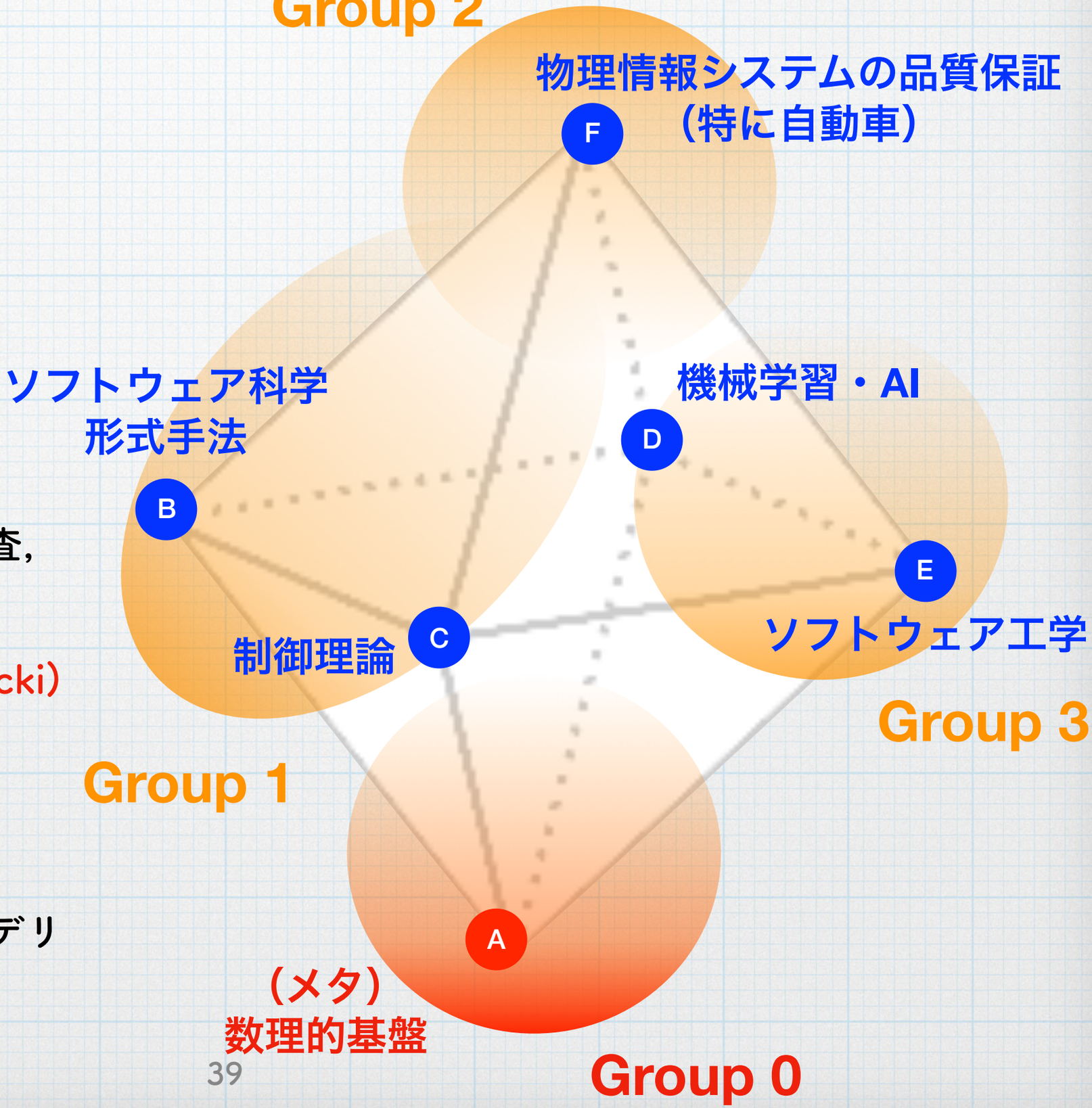
研究体制

* G0： メタ理論的統合グループ
（勝股）
論理学，代数学，圏論，
プログラミング言語理論，…

* G1： ヘテロジニアス
形式手法グループ（蓮尾・岸田）
ソフトウェア科学，
制御理論，形式手法，モデル検査，
自動定理証明，…

* G2： 産業応用グループ(Czarnecki)
自動運転システム

* G3： インテリジェンス協働
形式手法グループ（石川）
ソフトウェア工学，テスト，モデリング，要件工学，…



論理学,
代数学, 圏論,
…

ERATO MSD

Abstract Technique

$T[\_]$

$$A ::= \texttt{true} \mid \texttt{false} \mid A_1 \wedge A_2 \mid \neg A \mid a_1 < a_2 \mid \forall x \in {}^*\mathbb{N}.\, A \mid \forall x \in {}^*\mathbb{R}.\, A$$

$$\begin{array}{ccc} FX & \xrightarrow{F\mathbf{beh}_c} & FZ \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\ X & \xrightarrow[\mathbf{beh}_c]{} & Z \end{array} \qquad \begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ {\scriptstyle c}\uparrow & \sqsupseteq & \uparrow{\scriptstyle d} \\ X & \xrightarrow[f]{} & Y \end{array}$$

system behavior　　simulation

Identify "mathematical essense"

Choose parameter e_1

Choose parameter e_2

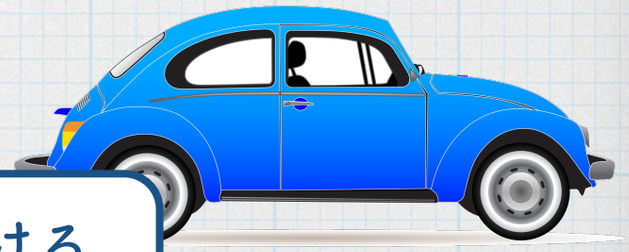
Existing Technique

$T_1 = T[e_1]$

Novel Technique

$T[e_2]$

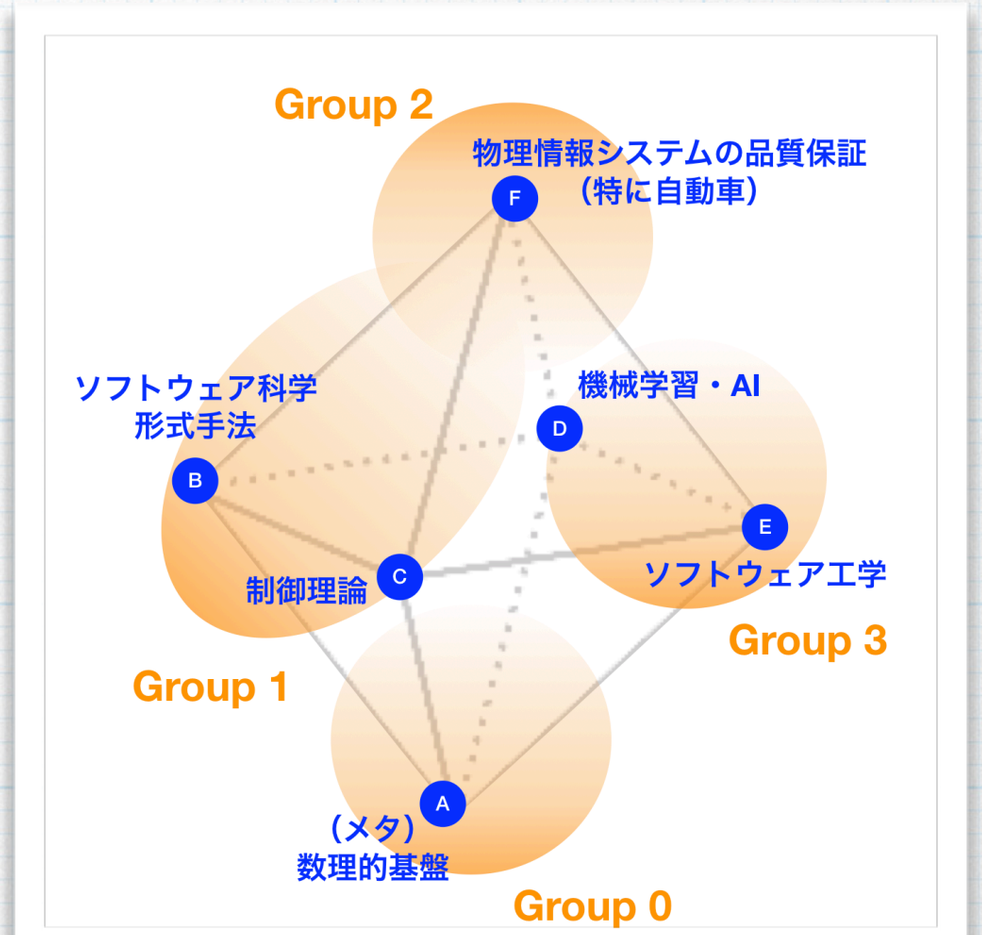
オートマトンにおける
グラフ可達性



確率的オートマトンにおける
到達確率

数理的基盤

* 理論的レベルでの貢献
  * 前のスライド
* プロジェクトにおける人的貢献
  * 異分野協働の通訳に
  * （未知の領域に踏み出す際は，理論家と一緒に）

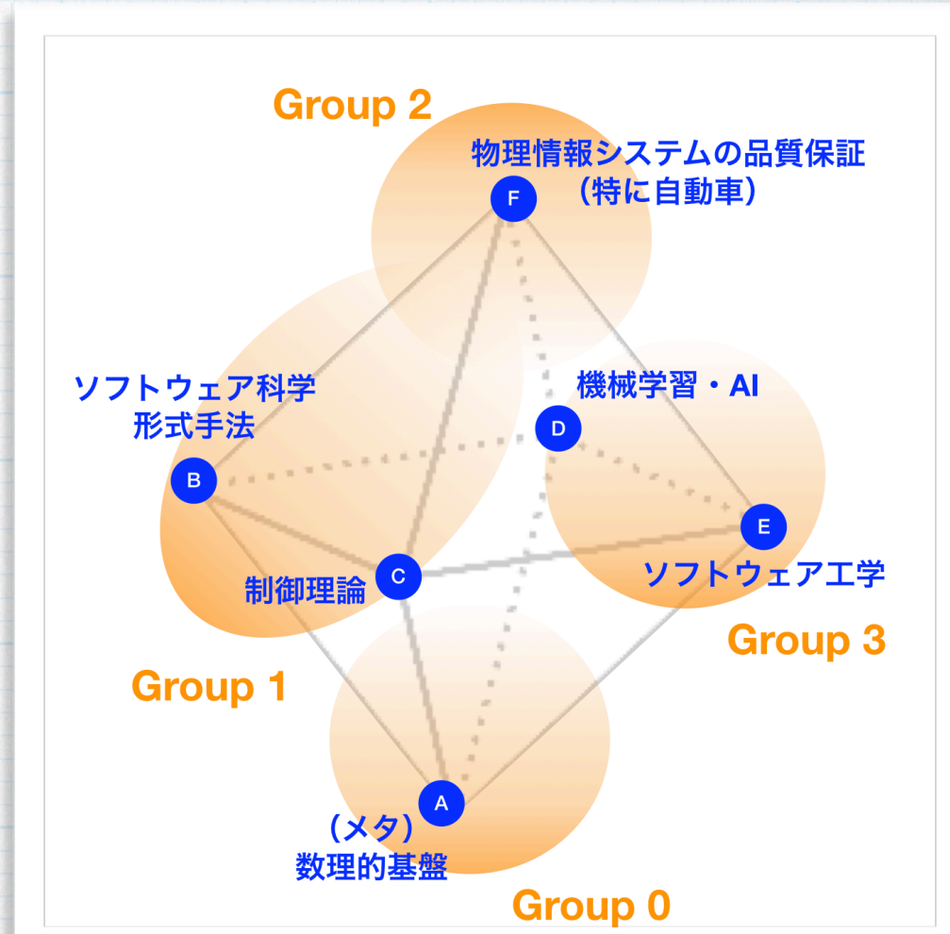


アウトライン

- ソフトウェア品質保証におけるスペクトル
  - ERATO MMSD の目指すもの：
    テストと形式検証の最適な組み合わせ
- 具体的技術シーズの紹介
  - サーチベーステスト
  - 自動運転システムのテストシナリオ記述・生成
  - 形式的安全アーキテクチャ
  - 実行時監視アルゴリズム
  - ニューラルネットのオートマトン近似
  - 自動運転システムの判断・経路生成アルゴリズム
  - 形式仕様記述を支援する対話型ツール
  - 数理的理論基盤

ERATO MMSD の「今日」

* 新規な理論・アルゴリズム
  * 学術界で確かな国際的 visibility
  * ツールプロトタイプも多数
    （一部は公開済み，前述）
    * 実行時監視，サーチベーステスト，…



* 多分野を統合する学術研究が，
  具体的応用に牽引され，
  また，確固とした数理的基盤に支えられるような
  研究体制ができた．

ERATO MMSD の「明日」

* （プロジェクト終了時，2022年3月までの未来像）

* 以下のツール群を具体的目標に設定．自動運転システムの研究開発に確かな貢献を行う．
  （各項目については後で説明します）

* 1. 形式的安全アーキテクチャ
* 2. 自動運転システムのテストシナリオ生成ツール
* 3. （安全ながら保守的になりすぎない）
  自動運転システムの判断・経路生成アルゴリズム
* 4. 形式仕様記述を支援する対話型ツール
* 5. 上記を結合する自動運転ソフトウェアプラットフォーム．
  認識・判断・操作・信頼性保証

ERATO MMSD の「明日」

* （プロジェクト終了時，2022年3月までの未来像）

* これらツール群の技術移転
  （持続的な保守・改良のため商業化）

* 基礎的方法論の戦略的特許出願
  ➜ 業界全体に広く使っていただくことが目的

* 各企業さまとの協働を継続
  ➜ ニーズとシーズのマッチング，我々の一般論を個別応用へ適合

ERATO MMSD の「明後日」

* スケールダウンできる形式手法を実現
  コスト-利益のバランスを選べるツール・手法を産業界に多数提供

* その実現のため，以下のペアそれぞれにおいて，両輪の発展を行う（詳細は後ほど）
  * 論理的手法 vs 統計的手法
  * 形式検証 vs テスト
  * 学術的基礎研究 vs 産業応用