1M3-4

制約に基づいたパラメトリックハイブリッドシステムの精度保証シミュレーション

Validated Simulation of Parametric Hybrid Systems Based on Constraints

松本翔太 *1 Shota MATSUMOTO　別納健市 *1 Kenichi BETSUNO　増田健太 *2 Kenta MASUDA　上田和紀 *2 Kazunori UEDA

*1早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻
Graduate School of Fundamental Science and Engineering, Waseda University

*2早稲田大学理工学術院情報理工学科
Faculty of Science and Engineering, Waseda University

The purpose of this research is the development of a highly reliable simulator of hybrid systems, i.e., systems involving both discrete changes and continuous evolutions. In particular, we aim at rigorous simulation of parametric hybrid systems, which enables not only the analysis of possible behavior of models but also the design of parameters that realize desired properties. In this research, we have developed an integrated simulation method using symbolic formula manipulation and conservative overapproximation by interval arithmetic. This method allows us to simulate all possible trajectories of hybrid systems described by a constraint-based formalism. We focus on (i) reducing computational costs of complex symbolic formulas and (ii) computing zero-crossings of functions that cannot be handled analytically. This integrated method uses affine arithmetic, the interval Newton method and the mean value theorem. This method preserves the first-order dependencies of uncertain quantities throughout simulation. We have implemented the method with a web frontend that provides visualization of parametric trajectories.

1. はじめに

離散変化と連続変化の両方を含む動的システムはハイブリッドシステム [6] と呼ばれる．ハイブリッドシステムの一例としてサイバーフィジカルシステム（コンピュータプログラムとそれを取り巻く実世界の物理現象との相互作用を含む系）を考えることができ，そのシミュレーションや検証は現在世界的に注目されている分野である．ハイブリッドシステムは，乗り物や原子炉の制御など，安全性が重要視されるシステムを多く含む．それゆえ，ハイブリッドシステムのシミュレーションや検証を行うツールは計算結果に誤りを含むことが無いよう，数学的に正しさの保証された厳密な計算を行うことが望ましい．特に本研究では，不確定パラメタ（状態変数の初期値が幅を持っている場合など）を含むパラメトリックハイブリッドシステムのシミュレーションを精度保証された形で行うことを目標とする．

ハイブリッドシステムの精度保証シミュレーションには，大きく分けて 2 つの挑戦的課題が存在する．1 つめは微分方程式の解である連続軌道を計算することであり，2 つめは連続軌道と離散変化が発生する境界面との交点を計算することである．これらの計算ではしばしば複雑かつ非線形な方程式ないしは不等式系を解くことを要求される．本研究では 2 つめの課題に着目し，パラメトリックハイブリッドシステムの精度保証された軌道を求める方法を追求する．1 つめに関しては連続軌道を与える微分方程式が解析的に求解可能であるか，あるいはパラメタを含む線形微分方程式で精度保証近似可能であると仮定する．

ハイブリッドシステムの精度保証シミュレーションを行う他のツール [1][4][14][9] では，区間演算に代表される精度保証数値計算を用いることで真の軌道の到達可能範囲の過大近似 (overapproximation) を求めることができる．しかし，既存のツールの主眼は到達可能範囲の解析におかれており，パラメタを明示的に扱っていない．本研究では，パラメタに関する記号的情報を保存した軌道を計算する手法を構築し，パラメトリックハイブリッドシステムの感度解析に役立つツールを提供することを目指し，可視化環境を含む統合的なシステムを実装した．

連絡先: 松本翔太, 早稲田大学大学院基幹理工学研究科情報理工学専攻, 〒 169-8555 新宿区大久保 3-4-1 63 号館 5 階 02 号, 03-5286-3340, matsusho(at)ueda.info.waseda.ac.jp

2. ハイブリッド制約言語 HydLa

本研究で構築したシミュレーションアルゴリズムへの入力は HydLa [11] プログラムの形式で与えられる．HydLa は制約に基づきハイブリッドシステムをモデリングする宣言型言語である．制約による記述は宣言的でありながら，同期や条件分岐を含む制御構造を表現可能である．さらに，制約を用いることで不確定性を自然に扱うことも可能であり，パラメトリックハイブリッドシステムの記述に適している．本節では，例題を用いて HydLa の概要を紹介する．HydLa の構文や意味論に関する詳細は [11] を参照されたい．

図 1 は例題モデル（[2] より引用）の概要を示しており，図 2 は本モデルに対応する HydLa プログラムを示している．本モデルは 2 つの水槽を含んでおり，1 つ目の水槽の水がパイプを通して 2 つ目の水槽に流れ込んでいる．`x1` と `x2` は各水槽の水位を示しており，`v1` と `v2` は各水槽のバルブの状態を示している．v$i = 1$ (0) はそれぞれバルブが開いている（閉じている）ことを示している．HydLa における変数はすべて（暗黙的に）時刻に関する関数変数である．本プログラムは，時刻 0 において成り立つ性質を記述したいくつかの制約から成り立っている．制約 `INIT` はシステムの初期状態を記述している．本モデルでは，1 つ目の水槽の初期水位は $1.9 \leq$ `x1` ≤ 1.9001 を満たす不定値となっている．制約 `X1` と制約 `X2` は各水位の連続的挙動に関する制約である．時相演算子 `[]` はその制約が有効になった時刻以降常に `[]` のついた制約が成立することを意味

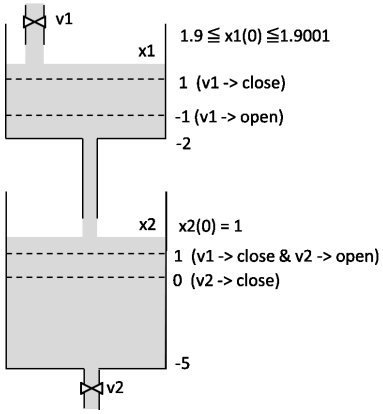

図 1: 2 つの水槽のモデル

```
INIT <=>
   1.9 <= x1 <= 1.9001 /\ x2 = 1
   /\ v1 = 0 /\ v2 = 1.
X1 <=>
   [](((v1 = 0 => x1' = -x1 - 2)
      /\ (v1 = 1 => x1' = -x1 + 3)).
X2 <=>
   [](((v2 = 0 => x2' = x1)
      /\ (v2 = 1 => x2' = x1 - x2 - 5)).
V1_CONST <=> [](v1' = 0).
V2_CONST <=> [](v2' = 0).
V1_OFF2ON <=>
   [](v1- = 0 /\ x1- = -1 => v1 = 1).
V1_ON2OFF <=>
   [](v1- = 1 /\ v2- = 1 /\ x1- = 1 => v1 = 0).
V1V2_OFF2ON <=>
   [](v2- = 0 /\ x2- = 1 => v2 = 1 /\ v1 = 0).
V2_ON2OFF <=>
   [](v2- = 1 /\ x2- = 0 => v2 = 0).

INIT, X1, X2,
(V1_CONST, V2_CONST)
   << (V1_OFF2ON, V1_ON2OFF,
       V1V2_OFF2ON, V2_ON2OFF).
```

図 2: 2 つの水槽の制御を記述した HydLa プログラム

する．x1' は時間微分 $dx1/dt$ を表す．含意記号 => の右辺は，その左辺（ガードと呼ぶ）が満たされる時に有効になる．制約 V1_CONST と制約 V2_CONST は v1 と v2 が変化しないことを記述しており，これがこのモデルの通常時のふるまいとなっている．制約 V1_OFF2ON，V1_ON2OFF，V1V2_OFF2ON，V2_ON2OFF はバルブの開閉に関する制約であり，x1- のように変数のあとにマイナス記号がついたものは左極限値 $\lim_{t_l \uparrow t} x1(t_l)$ を表す．最後の 4 行はこれらの制約間の優先順位を << を用いて宣言しており，水位が閾値をまたぐ時以外はバルブの状態は変化しないことを表現している．なお，より優先度の高い制約を持たない制約（本プログラムでは Vi_CONST 以外の制約）は常に有効な制約である．HydLa プログラムの宣言的意味はプログラムに記述された仕様を満たす軌道の集合であり，各時刻における仕様は制約間の優先順位に違反しないもののうち，極大無矛盾な制約の部分集合として与えられる．

入力: $HydLa$: basic HydLa program,
$MaxT$: maximum simulation time

1: $MS := TopologicalSort(HydLa)$ // list of candidate sets of constraints
2: $V := GetVariables(HydLa)$
3: $T := 0$ // current time
4: $S := true$ // current set of constraints
5: $P := true$ // constraints on symbolic parameters
6: $G_p := \emptyset$ // guards that caused the previous event
7: $E := \emptyset$ // expanded always consequents
8: **while** $T <$ $MaxT$ **do**
9: // Point Phase (PP)
10: $S := Subst(S, T)$
11: $(S, P, E, \_, \_) :=$
$MCS(S@G_p, MS, E, P, T, CheckConsistencyPP)$
12: **if** $S = false$ **then**
13: break
14: **end if**
15: $(S, P) :=Enclose(AddParameters(S, P, V))$
16: // Interval Phase (IP)
17: $(S, P, E, A_-, A_+) :=$
$MCS(S@G_p, MS, E, P, T, CheckConsistencyIP)$
18: $S := SolveDifferentialEquation(S)$
19: **if** $S = false$ **then**
20: break
21: **end if**
22: $(MinT, P, G_p) := GetElement(CompareMinTime($
$(\bigcup_{(g \Rightarrow c) \in A_-} FindMinTime(Subst(g, S), P, G_p))$
$\cup (\bigcup_{(g \Rightarrow c) \in A_+} FindMinTime(Subst(\neg g, S), P, G_p))$
$\cup \{(MaxT - T, true)\}))$
23: $T := MinT + T$
24: **end while**

図 3: HyLaGI によるシミュレーションアルゴリズムの概略

3. 精度保証シミュレーション環境

本研究では，HydLa で記述されたハイブリッドシステムモデルの精度保証シミュレーションを行う環境を提供するため，記号実行に基づく HydLa のシミュレータである HyLaGI と，その web フロントエンドである webHydLa を開発および公開している．

3.1 HyLaGI

HyLaGI は C++で実装されており，[12] の URL にてダウンロードできるほか，後述する webHydLa を通じて利用することができる．現在は Mathematica をバックエンドの制約ソルバとして利用しているほか，精度保証数値計算のために kv ライブラリ [5] を利用している．Mathematica は制約の連言の無矛盾性判定，常微分方程式の求解，離散変化時刻に関する最小化問題，算術式や論理式の変形に用いている．さらに kv ライブラリを用いることで，精度保証数値計算の代表である区間演算 [10] と，パラメタの線形項を残しながら精度保証数値計算を行う技術であるアフィン演算 [3] を行っており，Mathematica の記号的計算と連携させることで全体としてパラメタを保存した記号的シミュレーションを実現している．

HyLaGI が用いているシミュレーションアルゴリズムの概略を図 3 に示す．アルゴリズムの詳細については [7] および [8] を参照されたい．このアルゴリズムは基本 HydLa [11] のプログラムを入力とし，軌道の集合を出力する．基本 HydLa のプ

ログラムは，元の HydLa プログラム中の個々の制約間の優先度に関する仕様を，その仕様を満たす制約の部分集合を要素とする半順序集合に変換することで得られる（この半順序集合中の順序関係として，部分集合間の包含関係を用いる）．別の言い方をすると，制約集合 S が制約 C を含むならば，S は必ず C より高い優先順位を持つような制約集合のみを，半順序集合中の要素として選ぶ．

本アルゴリズムは 2 つのフェーズを交互に切り替えながら進行する．1 つ目のフェーズは離散変化を扱う Point Phase であり，2 つ目のフェーズは連続変化を扱う Interval Phase である．各フェーズでは，HydLa の意味論に従い制約の極大無矛盾集合（*maximal consistent set, MCS*）を計算して S に代入し，記号パラメタに関する条件を P に代入する (10 行目–16 行目)．*MCS* は各フェーズに対応した無矛盾性判定を行う関数を引数とする高階関数である．9 行目の *Subst* は S 中の各数式に現れる時刻変数 t に対し，現在時刻 T を代入する．Point Phase の最後に，その時刻で値が不確定な変数に対して記号パラメタを導入し，その値の区間包囲を *Enclose* 関数によって計算する（14 行目）．*Enclose* 関数はアフィン演算を用いることで数式の過大近似を行い，その結果が誤りを含むことが無いよう配慮している．アフィン演算では基本演算を実行するたびに，精度保証数値計算による計算誤差を包含するための記号パラメタが導入される．*Enclose* 関数ではこのパラメタ数の増大が計算精度に悪影響を与えないよう，kv ライブラリを用いてパラメタの削減を行う．Interval Phase では，再び *MCS* を求めるとともに，成立したガードの集合 A_+ と成立しなかったガードの集合 A_- を得る（16 行目）．その後，S 中の微分方程式を解き（17 行目）次の離散変化時刻の計算を *CompareMinTime* および *FindMinTime* によって行う．(21 行目)．*Subst* は微分方程式の解を S と g と $\neg g$ 中の各数式に対して代入している．*FindMinTime* では，ニュートン法の区間演算による拡張版である区間ニュートン法 [10] と中間値の定理を利用することで，パラメタに関する情報を残しつつも離散変化条件を表現する方程式の精度保証解を求めることを可能としている．本アルゴリズム中で *FindMinTime* と *Enclose* 以外の手続きはすべて記号的に計算され，すべての不確定量は P 中のパラメタとして扱われる．

HyLaGI の出力は，入力である HydLa プログラムの仕様を満たす軌道の集合である．軌道の集合は各時刻の不確定な変数値に対し，記号パラメタを用いることで表現する．例えば図 2 の例題プログラムに対しては，`x1` の初期値に対応する記号パラメタを導入する．

3.2 webHydLa

webHydLa は HyLaGI の web フロントエンドであり，[13] の URL に web ブラウザからアクセスすることで利用できる．webHydLa のスクリーンショットを図 4 に示す．webHydLa 上では，HydLa プログラムの編集や実行時オプションの設定のほか，シミュレーション結果を即座に 3 次元空間にプロットし，モデルのふるまいを確認することができる．さらに，webHydLa を用いることでユーザは有償かつ大規模なシステムである Mathematica をインストールすることなく，HyLaGI の機能を利用することができる．webHydLa はクライアント側が JavaScript で，サーバ側が Python で記述されており，サーバ側でシミュレーションを行い，その結果を受けてクライアント側が描画を行うという形で実装されている．

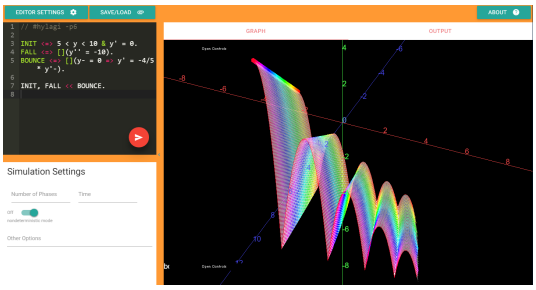

図 4: webHydLa のスクリーンショット

4. 例題による評価

本研究で構築した記号シミュレータである HyLaGI の性能を，図 6 に示すプログラムを用いて評価した．実験環境は OS : Gentoo Linux，CPU: Intel(R) Xeon(R) CPU E7- 4860 @ 2.27GHz，Memory: 256Gbyte となっている．本プログラムは 2 次元空間中で，放物線 $y = x^2$ の形状を持つ床の上で質点が反発係数 1 で跳ねるモデルに対応しており，その軌道例は図 5 に示してある．ボールは $x(0) = 2, y(0) = 8$ から落下を開始し，床との衝突を繰り返す．ここで水平方向の初速度を $0 \leq x'(0) \leq 0.00001$ とすることで，微小なパラメタをモデルに含めている．変数 `cont` は，離散変化を表すための補助変数として使用している．

図 7 は各衝突における変数値区間の幅を示している．この図において，`Mean & AA d {5,6,7,8,9}` は提案手法を用いたシミュレーションに対応しており，末尾の数字はアフィン演算による近似後に，この数まで記号パラメタを削減したことを示している．`Newton & IA` はアフィン演算の代わりに通常の区間演算を用い，さらに中間値の定理を用いなかった場合に対応しており，提案手法との比較のために掲載してある．図 7 から，提案手法が `Newton & IA` に比べて結果の区間幅を小さく抑えられていることと，使用する記号パラメタの数が多いほど区間幅を狭く保てることがわかる．一方でいずれの場合でも区間幅が 1 を超えてしまった時点で，離散変化時刻の計算に失敗しそれ以降のシミュレーションはできない．

図 8 は各衝突に関する計算時間を示している．`Symbolic` は数値的な計算を用いず，すべての計算を解析的に行った場合に対応しており，3 ステップ目以降の計算時間が爆発してしまったため 2 ステップ目までで打ち切っている．提案手法は，ナイーブな区間演算と解析的な計算の中間の特性を持っており，ナイーブな区間演算に比べて時間はかかるが，ある程度以上に計算時間が爆発していないことが図 8 において確認できる．

5. まとめと今後の課題

本研究ではパラメトリックハイブリッドシステムの精度保証シミュレーションアルゴリズムを提案・実装し，可視化環境を含めた総合的なツールとして提供した．構築したアルゴリズムは記号シミュレーションをベースに，区間ニュートン法，アフィン演算，中間値の定理を用いた方程式の求解という 3 つの技術を連携させている．本手法により，モデルの軌道をパラメタに関する線形項を残した形で計算することができ，計算結果の精度は通常の区間演算を用いたものと比べて向上している．

今後の課題としては，解析解を持たない非線形微分方程式を，パラメタを含む線形微分方程式によって包囲して計算する

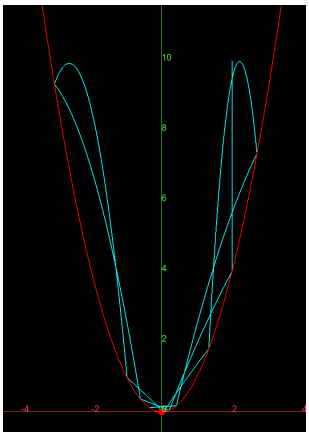

図 5: 放物線上で跳ねる質点の軌道例

```
INIT   <=> x = 2 /\ y = 8 /\ 0 <= x' <= 0.00001 /\ y' = 0.
FALL   <=> [](cont = 1 => x'' = 0 & y'' = -9).
BOUNCE <=> [](y- = (x-)^2 => cont = 0
  /\ x' = x'- - (4*(2*x- * x'- - (y'-))*x-)/(4*(x-)^2 + 1)
  /\ y' = y'- + (2*(2*x- * x'- - (y'-)))/(4*(x-)^2 + 1)).

INIT, FALL, [](cont = 1) << BOUNCE.
```

図 6: 放物線上で跳ねる質点のプログラム

手法を構築することで，シミュレーション能力をさらに向上させることを検討している．

謝辞

本研究の一部は，科学研究費 (26280024) の助成を受けて行った．

参考文献

[1] Chen, X., Abraham, E. and Sankaranarayanan, S. : Flow*: An Analyzer for Non-Linear Hybrid Systems, in Proc. International Conference on Computer Aided Verification, 2013, pp. 258–263.

[2] Chen, X., Schupp, S., Makhlouf, I. B., Abraham, E., Frehse, G. and Kowalewski, S. : A Benchmark Suite for Hybrid Systems Reachability Analysis, in Proc. 7th NASA Formal Methods Symposium, Springer, 2015, pp. 408–414.

[3] de Figueiredo, L. H. and Stolfi, J. : Affine Arithmetic: Concepts and Applications, Numerical Algorithms, Vol. 37, Issue 1-4, 2004, pp. 147–158.

[4] Frehse, G., Guernic, C. L., Donzé, A., Cotton, S., Ray, R., Lebeltel, Olivier., Ripado, R., Girard, A., Dang, T. and Maler, O. : SpaceEx: Scalable Verification of Hybrid Systems, in Proc. International Conference on Computer Aided Verification, LNCS 6806, Springer, 2011, pp. 379–395.

[5] 柏木雅英：kv ライブラリ, http://verifiedby.me/kv/.

[6] Lunze, J. : Handbook of Hybrid Systems Control : Theory, Tools, Applications, Cambridge University Press, 2009.

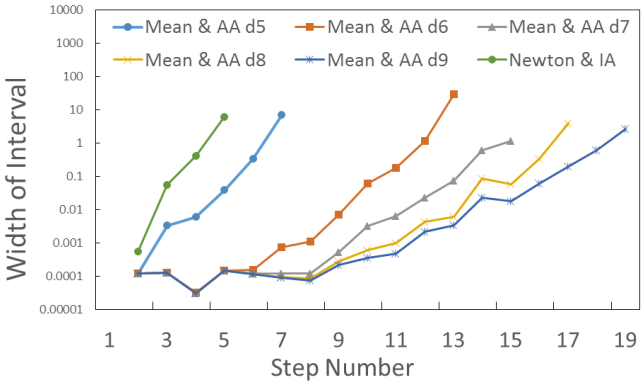

図 7: 図 6 のシミュレーションにおける区間幅

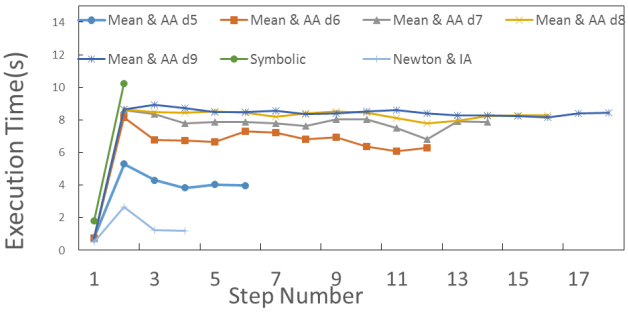

図 8: 図 6 のシミュレーションにおける計算時間

[7] Matsumoto, S. and Ueda, K. : Symbolic Simulation of Parametrized Hybrid Systems with Affine Arithmetic, in Proc. 23rd International Symposium on Temporal Representation and Reasoning, 2016, pp. 4–11.

[8] 松本翔太：Validated Simulation of Parametric Hybrid Systems Based on Constraints, 博士学位論文, 早稲田大学, 2017.

[9] Mimram, S., Bouissou, O. and Chapoutot, A. : HySon: Set-based Simulation of Hybrid Systems, in Proc. IEEE 23rd International Symposium on Rapid System Prototyping, 2012, pp. 79–85.

[10] Moore R. E., Kearfott R. B. and Cloud M. J. : Introduction to Interval Analysis, Society for Industrial and Applied Mathematics, 2009.

[11] Ueda, K., Matsumoto, S., Takeguchi, A., Hosobe, H. and Ishii, D. : HydLa : A High-Level Language for Hybrid Systems. Second Workshop on Logics for System Analysis, 2012, pp. 3–17.

[12] 早稲田大学上田研究室：HydLa, http://www.ueda.info.waseda.ac.jp/hydla/.

[13] 早稲田大学上田研究室：webHydLa, http://webhydla.ueda.info.waseda.ac.jp/.

[14] Zeng, Y., Rose, C., Brauner, P., Taha, W., Masood, J., Philippsen, R., O'Malley, M. and Cartwright, R. : Modeling Basic Aspects of Cyber-Physical Systems, Part II, in Proc. 11th IEEE International Conference on Embedded Software and Systems, 2014, pp. 550–557.