ハイブリッド制約処理系 HyLaGI による微小誤差を用いたモデル解析

若槻 祐彰　松本 翔太　伊藤 剛史　和田 努　上田 和紀

時間経過に伴い状態が連続変化したり，状態や方程式系自体が離散変化したりするシステムのことをハイブリッドシステムと言う．HydLa は，ハイブリッドシステムを制約階層によってモデリングする宣言型言語である．HyLaGI は数式処理に基づいた HydLa の処理系であり，記号パラメータを含むシステムを扱える．例えばボール同士の衝突における三体同時衝突のような，特異点の近傍の運動に興味がある場合に，特異点から微小値 ε だけずれた運動を計算し，ε を限りなく 0 に近づけることで解析を行うことができる．HyLaGI はそのような解析をサポートする機能を有しており，特異点の解析以外にも誤差の解析に利用することも可能である．本研究では本機能を利用していくつかの例題の解析を行うことで，その有用性を示す．

Hybrid systems are dynamical systems that are composed of continuous behaviors and discrete behaviors. HydLa is a constraint-based modeling language for Hybrid systems. HyLaGI is an implementation of HydLa. HyLaGI can simulate systems that have simbolic parameters. For example, when we are interested in movements that are close to a singularly point, we can simulate movements that are away from the singular point by a symbolic parameter ε and then analyze movements by taking the limit of ε. HyLaGI supports such simulation using ε. In this paper, we show that this function is useful for analyzing a number of examples.

1　研究背景

1.1　ハイブリッドシステム

ハイブリッドシステム [3] とは，時間の経過に伴い状態が連続変化したり，状態や方程式系自体が離散変化したりするシステムのことである．物理現象やコンピュータ制御されたシステムの多くは，ハイブリッドシステムとしてみなすことができる．このように物理学だけでなく，制御工学や生命工学などのさまざまな分野においてシステムをモデリングする際に利用することができる．

1.2　ハイブリッド制約言語 HydLa

HydLa [6] [5] とは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．システムの挙動を方程式や論理式を用いた制約によって記述する．制約間に優先順位をつけることで，システムを簡潔に表現することができる．

例えば，地面を跳ねる質点をモデリングすると図 1 のように表すことができる．

```
1  INIT <=> y=10 & y'=0.
2  FALL <=> [](y''=-10).
3  BOUNCE <=> [](y-=0 => y'=-4/5*y'-).
4  INIT, FALL << BOUNCE.
```

図 1　床で跳ねる質点のプログラム

ここで，変数 y は質点の位置を表している．INIT は，質点の初期位置が 10，初速度が 0 であることをを表す制約である．FALL は，時刻 0 以降において質点の加速度が -10 であることを表す制約である．

Model Analysis by using micro errors in Hybrid Constarint Processing System HyLaGI

Yoshiaki Wakatsuki　Shota Matsumoto　Takeshi Ito　Tsutomu Wada, 早稲田大学大学院基幹理工学研究科, Graduate School of Fundamental Science and Engineering, Waseda University..

Kazunori Ueda, 早稲田大学理工学術院, Faculty of Science and Engineering, Waseda University..

BOUNCE は，時刻 0 以降において質点の位置の左極限値が 0 の場合，質点の速度は現在の速度の左極限値に $-4/5$ をかけた値である，つまり質点が地面に衝突した時に，速度が減衰して跳ね返ることを表している制約である．この => 記号の左辺をガード条件といって，この論理式が真のときに右辺の方程式が有効になる制約を表している．ここで地面に衝突した瞬間は，FALL と BOUNCE の二つの制約が矛盾する．そのような場合にどちらの制約を優先するのかを制約階層 [1] を用いて表したのが，4 行目の INIT, FALL << BOUNCE である．ここでは FALL と BOUNCE が矛盾した場合 BOUNCE を優先するように記述してある．

1.3 HydLa 処理系 HyLaGI

HyLaGI [4] は数式処理を用いた HydLa の処理系であり，浮動小数点誤差のないシミュレーションが可能である．また記号実行が可能で，システム内の不定値をパラメータとしてそのまま扱うことができる．

例えば，図 1 をシミュレーションした結果として図 2 のように出力する．3,20 行目の PP は離散変化するフェーズ (Point Phase) を表し，11 行目の IP は連続変化するフェーズ (Interval Phase) を表している．それぞれのフェーズの出力内容は，その時に採用されなかった制約 (unadopted modules) や衝突した制約 (unsat mod, unsat cons)，成立 (positive) もしくは不成立 (negative) に変化したガード条件，またその時の時刻 (t) と各変数 (y,y',y'') の値となっている．

2 誤差を用いたモデル解析

HydLa と HyLaGI を用いることで，物理現象やコンピュータ制御されたシステムのふるまいをシミュレーションすることができる．しかし，現実にモデルのふるまいを考えると与える初期値に誤差が入ることが考えられる．この場合，誤差の影響により，システムのふるまいが誤差を含まない場合のふるまいと大きくことなる場合がある．そのためシステムの性質を解析する際に，モデルに誤差が与える影響を調べることは大変重要である．このような誤差解析だけでなく，ある値とその近傍を対象としたシミュレーション

```
1  ---------Case 1---------
2  ---------1---------
3  ---------PP 1---------
4  unadopted modules: {}
5  positive :
6  negative :
7  t : 0
8  y : 10
9  y' : 0
10 y'' : -10
11 ---------IP 2---------
12 unadopted modules: {}
13 positive :
14 negative :
15 t : 0->2^(1/2)
16 y : 10+t^2*(-5)
17 y' : t*(-10)
18 y'' : -10
19 ---------2---------
20 ---------PP 3---------
21 unadopted modules: {FALL}
22 unsat mod : {BOUNCE, FALL}
23 unsat cons : {y''=-10, y'=(-4)/5*y'-}
24 positive : y-=0=>y'=(-4)/5*y'-
25 negative :
26 t : 2^(1/2)
27 y : 0
28 y' : 2^(1/2)*8
29 # number of phases reached limit
```

図 2　床で跳ねる質点のプログラムの実行結果

ンや無限大や無限小を扱う場合にも微小な誤差は利用可能である．本研究では微小値を表すパラメータ ε を特別に処理する機能を設計・実装した．

微小値の処理を含んだアルゴリズムは図 3 のようになる．通常の HydLa モデルのシミュレーションアルゴリズム [4] に加えて，高次項の削減 (*ReduceEpsilon* (2.1 節))，高次項の削減により起こる問題の解決 (*checkMinT, ShiftTme* (2.2 節))，不適切な場合分けの削減 (*GetElement* (2.3 節))，差分値の計算 (*DiffEpsilon* (2.4 節))，極限値の計算 (*LimitEpsilon* (2.5 節)) が追加されている．以降でこれらの処理について説明する．

2.1 高次項の削減

微小誤差を扱う際に，誤差が十分に小さいと考えると，ε の 2 次以降の項に関しては興味がなく，無視し

Require: HydLa プログラム HydLaProgram, シミュレーション終了時刻 MaxT
$MS := TopologicalSort(SolveCH(HydLaProgram))$
$M_{all} := MaxModuleSet(MS)$
$V := GetVariables(HydLaProgram)$
$T := 0$, $S := ture$, $CP := ture$
while $T <_{CP} MaxT$ **do**
// PP
$S := SubstituteMinTime(S,T)$
$(S, CP, E, \_, \_) :=$
$CalcMCS(S, MS, E, CP, T, CheckConsistencyPP)$
if $S = false$ **then**
$break$
end if
$(S, CP) := AddParameters(S,CP,V)$
$S := ReduceEpsilon(S)$
// IP
$(S, CP, E, A_+, A_-) :=$
$CalcMCS(S, MS, E, CP, T, CheckConsistencyIP)$
$S := SolveDifferentialEquation(S)$
if $S = false \vee \neg IsUnique(S, V)$ **then**
$break$
end if
$S := ReduceEpsilon(S)$
$getMinT := false$, $tmpMinT := 0$
while $!getMinT$ **do**
$(MinT, CP) := GetElement(CompMinTime($
$\{FindMinTime(S \wedge CP \Rightarrow g) | (g \Rightarrow c) \in A_-\}$
$\cup \{FindMinTime(S \wedge CP \Rightarrow \neg g) | (g \Rightarrow c) \in A_+\}$
$\cup \{FindMinTime(S \wedge CP \wedge M_-) | M_- \in (M_{all} \backslash M)\}$
$\cup \{FindMinTime(S \wedge CP \wedge \neg M_+) | M_+ \in M\}$
$\cup \{(MaxT - T, true)\}))$
$getMinT := checkMinT(MinT, CP)$
$(S, CP) := ShiftTime(S, CP, MinT)$
$tmpMinT := tmpMinT + MinT$
end while
$MinT := tmpMinT$
$T := MinT + T$
end while
$DiffS := DiffEpsilon(S)$, $S := LimitEpsilon(S)$

図 3　微小な値を扱う HyLaGI アルゴリズム

ても良い場合がある．このような場合のために ε に関する n 次以上の高次項を削減する機能がある．この機能は，HyLaGI が数式処理を用いてシミュレーションを行っているため，複雑な式の計算などは処理が重くなり計算できなくなる場合もあるので，興味のない高次項を削減することで計算を軽くするという目的がある．

まず，各フェーズにおいてそれぞれの変数は ε を含んだ式 $f(\varepsilon)$ で表される．この $f(\varepsilon)$ 中に現れる ε の高次項を削減するために，ここでは $\varepsilon = 0$ の近傍での n 次近似を用いることとする．これは，$f(\varepsilon)$ を $\varepsilon = 0$ においてテイラー展開したのちに，$n+1$ 次以降の高次項を切り捨てて近似する方法である．ここで *ReduceEpsilon* の処理は $n > 0$ の時のみ適用する，

2.2　高次項の削減による問題点の解決

高次項の削減により，各変数値は数式処理で求めた厳密な値からずれることになる．それによって，円状の枠の内部を運動していたボールが円の外側に出てしまったり，地面を跳ねていたボールが床を貫通してしまったりすることが考えられる．これらのような近似しない数式処理シミュレーションの結果と定性的に異なるふるまいを，望ましくないふるまいとして適切に処理する必要がある．

例えば，円状の枠の内部を運動するボールの例題に関して考えるとプログラムは図 4 のようになる．ここで枠に衝突した瞬間を近似した際に，ボールの位置がずれることを考える．すると，枠の内部にずれたときは，近似しなかった場合と同様に枠で跳ね返ったのと同様の運動をするが，枠の外側はすぐにもう一度枠にぶつかって外側に跳ね返ってしまう．これは円状の枠の内部を運動するボールのシミュレーションとしては望ましくない．

```
1  INIT <=> x=eps & y=4/5 & x'=4 & y'=-1.
2  RUN <=> [](x'' = 0 & y'' = 0).
3  EPS <=> 0 < eps < 1/10 & [](eps'=0).
4  BOUNCE <=> [](((x-)^2+(y-)^2=1 =>
5       x'=x'--(x-*x'-+y-*y'-)*2*(x-) &
6       y'=y'--(x-*x'-+y-*y'-)*2*(y-)).
7  INIT, EPS, RUN << BOUNCE.
```

図 4　円の内部を跳ねる質点

本研究では次の離散変化時刻と成立するガード条件を用いて対処する手法を考案した．この手法は，(i) 衝突後の処理において次の離散変化時刻を求めた際に，その時刻の極限 $\varepsilon \to 0$ を求めると 0 になり，かつ (ii) その際の離散変化の原因となるガード条件が，衝突の原因となったガード条件と等しい場合，これを無視する方法である．

近似により高次項が削減されて起こる各変数値の

変化は，切り捨てられる ε の $n+1$ 次以降の項からなるため，極限 $\varepsilon \to 0$ を考えると，ずれは限りなく小さくなっていくことになる．したがって (i)(ii) の場合に離散変化を無視する理由は，近似による変化によって，定性的に異なるふるまいを起こしてしまうからとなる．

この方法では，三体衝突のような特異点の近傍の解析の場合や $1/\varepsilon$ のような間接的に無限大を扱っている問題の場合も，望ましいふるまいかどうかを適切に判定できる．本研究ではこちらの手法を用いて対処した．

本研究の過程で，別の手法として衝突後の状態とガード条件との関係を用いて動作を検査する手法についても考えた．しかし，この手法では望ましい運動との比較が必要なので，どうなると望ましい運動なのかあらかじめわかっておく必要がある．この円状の枠の内部を運動する例題だと，誤差 $\varepsilon = 0$ の場合が望ましい場合の運動だと考えられる．ところが，三体同時衝突のような特異点の近傍における運動の解析をする場合，そもそも $\varepsilon = 0$ が特異点なので，単純に $\varepsilon = 0$ を代入した状況でガード条件との関係を比較すると，ボール同士がめり込んだりするなど，意図しない動作を起こしてしまう．また，$1/\varepsilon$ のような間接的に無限大を扱っている問題の場合，そもそも $\varepsilon = 0$ の場合を計算できないことがある．以上のように，$\varepsilon = 0$ の場合が必ずしも望ましいふるまいになるとは限らず，望ましいふるまいを正確に求めることができない問題点があるため，本研究では採用しなかった．

2.3 不要な場合分けの削減

微小誤差 ε は，HyLaGI 内では記号パラメータ “eps” として扱われている．HyLaGI は記号パラメータの値によって，動作の定性的な違いがある場合，場合分けを行ってシミュレーションする．微小誤差 ε も同様に場合分けが行われるが，ε は十分に小さいものとして仮定しているため，本手法では，ε の範囲が 0 もしくはその近傍を含まない場合の削減を行っている．

2.4 差分値の計算

微小誤差が与える影響を考える時に，運動が進むにつれて誤差の影響がどのように拡大していくのか，または収束していくのかは，システムの性質を解析する上で重要な問題である．そこで，微小誤差を含んだ変数値の変化だけではなく，初期時刻における誤差がどう影響していくのかは，大変重要である．

ここでは，ε を含む元の式と $\varepsilon = 0$ を代入した式，つまり誤差 ε が 0 だった場合との差分値の計算を行う．この処理はシミュレーションの最後に行われるため，場合分けなどの問題は発生しない．

2.5 極限値の計算

微小誤差が限りなく 0 に近づいた場合，つまり極限 $\varepsilon \to 0$ において各変数がどうなるのかも，システムの重要な性質である．そこで，各フェーズにおいてそれぞれの変数値の極限 $\varepsilon \to 0$ を求めた値も出力している．ここで $1/\varepsilon$ などは無限大となるため，間接的に無限大を扱ったシミュレーションが可能となる．

例えば，図 5 のようにすると撃力を微小値を用いて表現できる．これは静止している x1 に ε 時間だけ $1/\varepsilon$ の加速度が加わり，その後等速運動をするプログラムとなる．このようなプログラムを実行し，シミュレーションの後でこの極限 $\varepsilon \to 0$ をとることにより，瞬間的に無限大の力が加わったシミュレーションができる．

```
1  INIT <=> x1 = 0 & x1' = 0.
2  TIMER <=> time = 0 & [](time' = 1).
3  EPS <=> 0.0<eps<0.1 & [](eps'=0).
4  FORCE(x) <=> [](time<eps => x''=1/eps)
5             & [](time>=eps => x''=0).
6  INIT, TIMER, EPS, FORCE(x1).
```

図 5 撃力を扱う HydLa プログラム

3 例題による評価

3.1 円の内部を跳ねる質点

ここでは，高次項の削減の説明 (2.2 節) で利用した図 4 のプログラムに関して，実行にかかる計算時

間がどう変化するかを調べた．ここでこのプログラムの解析において，誤差 ε は十分に小さく，その 2 次以降は無視して良いという仮定のもとで，高次項を削減することを考える．

まず高次項の削減を行うことにより，ボールが円周に衝突した瞬間の位置と時刻において，2 次以降の項がもたらしていた微小な違いは削減されてしまう．そのため，各変数は微小にずれることになる．しかし，高次項 (2 次以降) を削減しても誤差の 1 次の項が残っているため誤差がどのような影響を与えているかは解析することができる．

ここで計測したのは，実行時間の変化となる．図 6 は高次項の削減を行わない場合の各フェーズの計算にかかる時間で，図 7 は高次項の削減を行った場合の各フェーズの計算にかかる時間となる．

この図 6 を見ると，高次項を削減しない場合ボールが円の枠に衝突するたびにだんだんと数式が複雑化するため，徐々に計算に時間がかかるようになっていくことがわかる．しかし，図 7 では高次項の削減を行っているため，ε の 2 次以降の項が削減されるので，数式の複雑化が起こらないようになっている．そのため，各フェーズにかかる時刻は，ある一定以上には増えていないことがわかる．

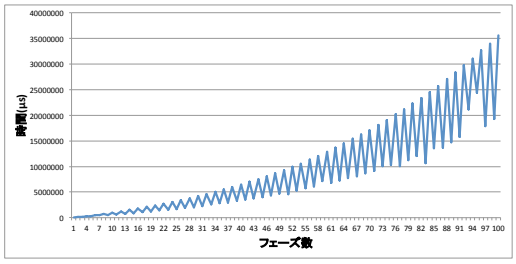

図 6　高次項の削減をしない場合の各フェーズの計算にかかる時間

3.2　玉突き衝突

これは xy 平面上の y 軸上に距離 1 の間隔で半径 1 のボールが並んだモデルである．そこにボールをぶつけて玉突き衝突を起こしていく．ぶつけるボールは x 軸方向に微小にずれていて，その誤差の影響がどのように伝達していくかを見ていく．プログラムは図 8 のようになる．

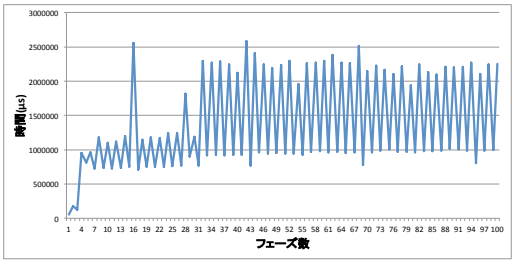

図 7　高次項の削減を行う場合の各フェーズの計算にかかる時間

```
1  INIT(x,y,cx,cy,dcx,dcy) <=>
2    x=cx & y=cy & x'=dcx & y'=dcy.
3  MOVE(x,y)<=>[](x''=0) & [](y''=0).
4  COLL(xa,ya,xb,yb) <=>
5    [](((xa--xb-)^2+(ya--yb-)^2=4)=>
6      xa'=(xa'-*(yb--ya-)^2+(yb'--ya'-)
7         *(xb--xa-)*(yb--ya-)
8         +xb'-*(xb--xa-)^2)/4 &
9      ya'=(ya'-*(xb--xa-)^2+(xb'--xa'-)
10        *(yb--ya-)*(xb--xa-)
11        +yb'-*(yb--ya-)^2)/4 &
12     xb'=(xb'-*(ya--yb-)^2+(ya'--yb'-)
13        *(xa--xb-)*(ya--yb-)
14        +xa'-*(xa--xb-)^2)/4 &
15     yb'=(yb'-*(xa--xb-)^2+(xa'--xb'-)
16        *(ya--yb-)*(xa--xb-)
17        +ya'-*(ya--yb-)^2)/4 ).
18 EPS <=> 0<eps<0.1 & [](eps'=0).
19 INITS{INIT(x1,y1,eps,0,0,10),
20       INIT(x2,y2,0,3,0,0),
21       INIT(x3,y3,0,6,0,0),
22       INIT(x4,y4,0,9,0,0)}.
23 MOVES{MOVE(x1,y1),MOVE(x2,y2),
24       MOVE(x3,y3),MOVE(x4,y4)}.
25 COLLS{COLL(x1,y1,x2,y2),
26       COLL(x1,y1,x3,y3),
27       COLL(x1,y1,x4,y4),
28       COLL(x2,y2,x3,y3),
29       COLL(x2,y2,x4,y4),
30       COLL(x3,y3,x4,y4)}.
31 INITS, MOVES << COLLS, EPS.
```

図 8　玉突き衝突の HydLa プログラム

このプログラムに対し HyLaGI の従来のシミュレーションを行うと，急激に数式が複雑化してしまい，4

つ目のボールに衝突するまでのシミュレーションができないプログラムとなる．しかし，高次項を削減することで図 9 のようにシミュレーションを行うことができる．

図 9 では，一つ目のボールが右に少しずれた位置から運動し始めるため 2 つめのボールは左，3 つめのボールは右，のように左右交互にずれが発生するふるまいになることがわかる．

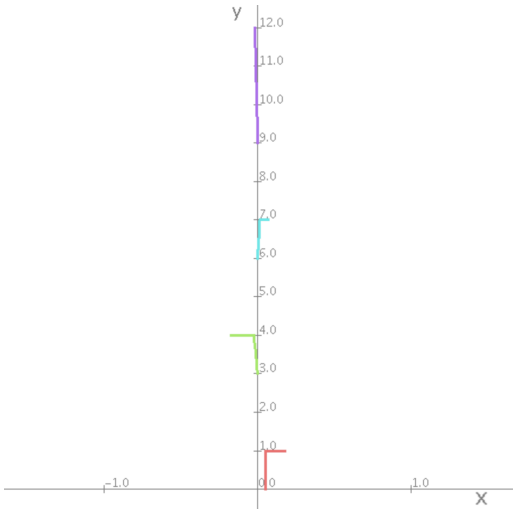

図 9　ボールが玉突き衝突していく例題

3.3　ニュートンのゆりかご

これは x 軸上を一直線に半径 0.5 のボールが 4 つ隙間なく並んでいて，そこに 5 つ目のボールをぶつけていく例題となる．ボールは等速直線運動を行い，衝突は完全弾性衝突である．両端には壁があり，壁の間を 5 つのボールが運動するようなプログラムとなる．

通常の方法では n 体同時衝突 ($n > 2$) がおこり，シミュレーションができないプログラムになるが，図 10 のようにボールの大きさに微小値を加えることで，ε の大きさの隙間を開けて並んでいるように近似すると，二体衝突が連続で起きるようになり，シミュレーションを行えるようになる．

このプログラムのシミュレーションを行うと，図 11 のようになり，微小ながら時間差で順にボールが衝突していくことがわかる．従って，隙間なく並んだ

```
1  INIT(m,m0,x,x0,xv) <=>
2    m=m0 & x=x0 & x'=xv & [](m'=0).
3  CONS(x) <=> [](x''=0).
4  COL(x1,m1,x2,m2) <=>
5    [](x1- - x2-)^2 = (1-eps)^2 =>
6      x1'=((m1--m2-)*x1'-+2*m2-*x2'-)
7          /(m1-+m2-)&
8      x2'=((m2--m1-)*x2'-+2*m1-*x1'-)
9          /(m1-+m2-)).
10 COL_WALL(z) <=>
11   [](z-=0 | z-=6 => z'=-z'-).
12 EPS <=> 0<eps<0.1 & [](eps'=0).
13 EPS, INIT(m1,1,x1,1/2,1),
14 INIT(m2,1,x2,2,0), INIT(m3,1,x3,3,0),
15 INIT(m4,1,x4,4,0), INIT(m5,1,x5,5,0),
16 (CONS(x1), CONS(x2), CONS(x3),
17   CONS(x4), CONS(x5)) <<
18 (COL_WALL(x1), COL_WALL(x2),
19   COL_WALL(x3), COL_WALL(x4),
20   COL_WALL(x5),
21   COL(x1,m1,x2,m2), COL(x1,m1,x3,m3),
22   COL(x1,m1,x4,m4), COL(x1,m1,x5,m5),
23   COL(x2,m2,x3,m3), COL(x2,m2,x4,m4),
24   COL(x2,m2,x5,m5), COL(x3,m3,x4,m4),
25   COL(x3,m3,x5,m5), COL(x4,m4,x5,m5)).
```

図 10　ニュートンのゆりかごの HydLa プログラム

ボール 4 つにボールをぶつけていくと，反対側のボールに運動が伝わるという，ニュートンのゆりかごのふるまいをシミュレーションできた．

また，微小値の極限 $\varepsilon = 0$ を考えると図 12 のように理想的なニュートンのゆりかごも扱うことができた．

3.4　流される質点の運動

このモデルでは二次元空間での質点の運動を考える．図 13 のように質点が置かれている空間を，y 軸と並行な 3 つの領域 A, B, C に分割し，それぞれの領域では質点に対し異なる力が働くものとする．領域 A では x 軸方向に 0.5，y 軸方向に -0.2，領域 B では x 軸方向に 0.01，y 軸方向に -0.1，領域 C では x 軸方向に -1.5，y 軸方向に 0.3 の加速度で力が働いていると考える．質点の初期位置は点 $(5 + \varepsilon, -10)$ とし，初速度は 0 とする．プログラムは図 14 のようになる．

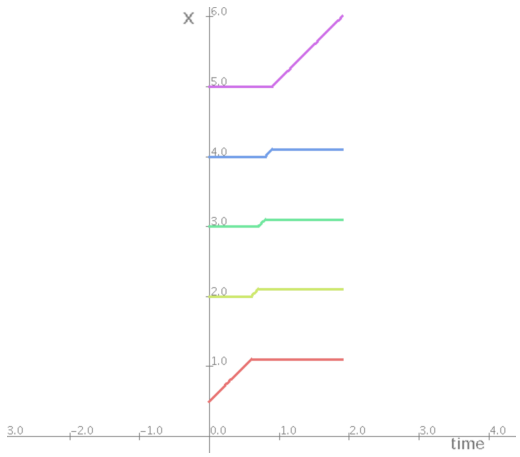

図 11　ニュートンのゆりかごのシミュレーション結果

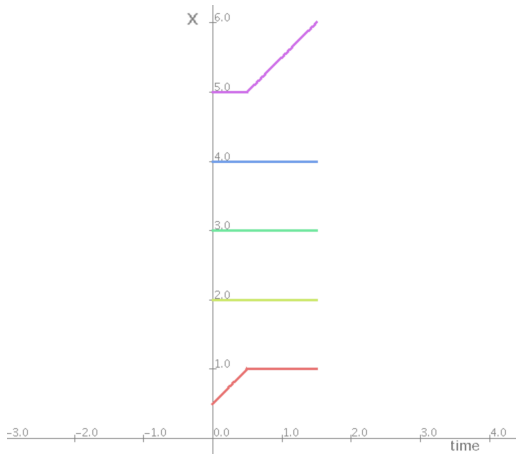

図 12　理想的なニュートンのゆりかごのシミュレーション結果

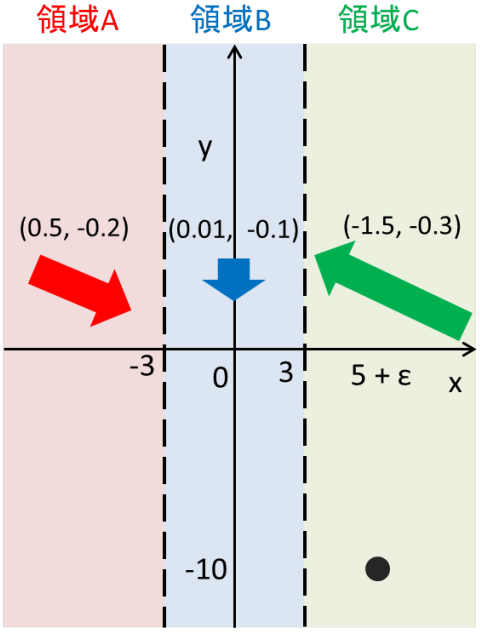

図 13　流される質点の運動における各領域

```
1  INIT <=>
2     x=5+eps & y=-10 & x'=0 & y'=2.
3  EPS <=> -0.2<eps<0.2 & [](eps'=0).
4  MOVEA <=>
5     [](x<=-3 => x''=0.5 & y''=-0.2).
6  MOVEB <=>
7     [](-3<x<3 => x''=0.01 & y''=-0.1).
8  MOVEC <=>
9     [](x>=3 => x''=-1.5 & y''=0.3).
10 INIT, EPS, MOVEA, MOVEB, MOVEC.
```

図 14　流される質点の運動の HydLa プログラム

この質点の運動の軌道は次の図 15 のようになる．

この軌道で初期値の誤差がどのような影響を与えるか分析する．微小値の与える誤差の差分値を計算すると図 16 のようになる．図 16 では，各時刻における x と y の本来の軌道 (誤差 ε=0 の場合) とのずれを表している．赤線が x で青線が y を表している．これは ε が 1 まで変化した場合の軌道のずれを描いてある．誤差の影響が小さくなっている部分を囲ってあるが，特定の時刻において影響が小さくなっていることがわかる．

このような流される質点の運動において，質点の運動に関してある特定の時刻で初期値の誤差 ε の影響が小さくなることが分析できた．

このように誤差の影響が収束するような点が分析できるため，物体の運動に不確定性がある場合に，システムの性質を利用した応用が考えられる．例えば，ロボットに物体をキャッチさせたい場合に，物体の運動の不確定性が収束するようなキャッチ可能な地点を計算する，などの応用が可能になる．

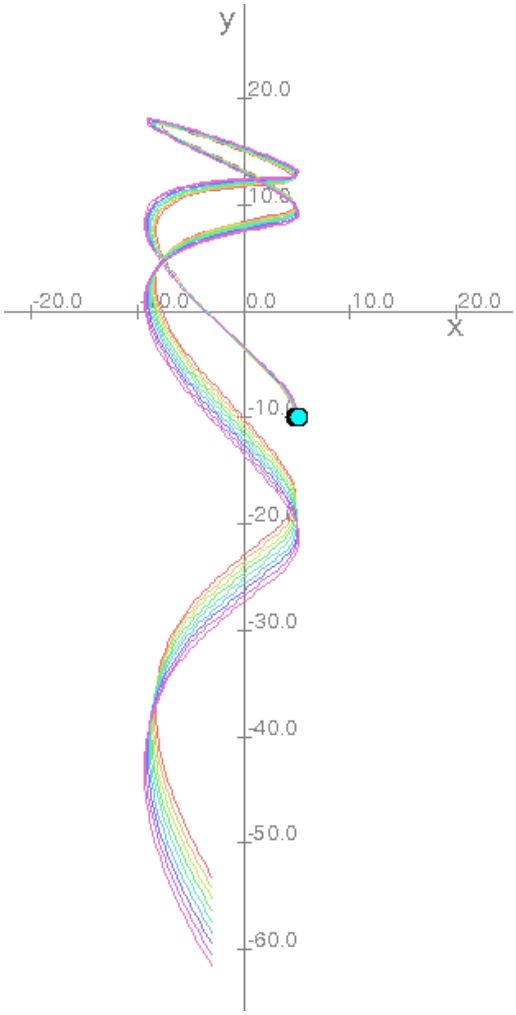

図 15　流される質点の運動の軌道

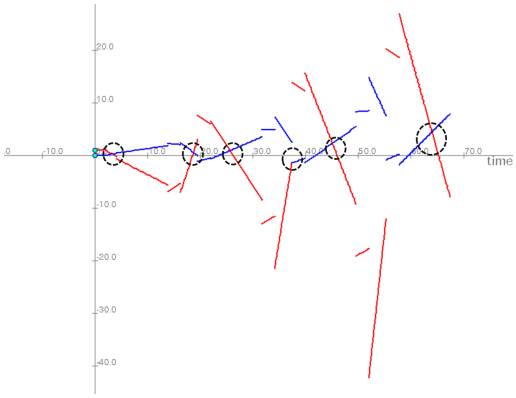

図 16　流される質点の運動の誤差の影響

4　関連研究

文献 [2] では，従来の HydLa，HyLaGI で直接扱うことのできない離散変化が連続しておこるモデルを扱っている．この文献中の例題 Fig.8,9 (ニュートンのゆりかご) や Fig.11,12 (ボールが左右からほぼ同時にぶつかるニュートンゆりかご)，Fig.14 (ボールの質量がすべて異なるニュートンのゆりかご) は，本論文の図 10 と同様に微小値を用いたモデリングを行い，解析することができた．また，Fig.15,16 (２つ並んだボールを左から押す例題)，Fig.28 (上下に重なった物体の下の物体に撃力を与える例題) は，微小値を用いることでモデルを HydLa プログラムで表すことができたが，HyLaGI の実装が完全ではないため，まだ解析は行えていない．原理的にはシミュレーション可能な例題であるため，HyLaGI を修正でき次第解析を行う予定である．

5　まとめと考察

5.1　考察

提案手法を用いることで，微小値であることを利用して高次項の削減を行えるため，だんだんと数式が複雑化していくシミュレーションにおいてその複雑化を軽減して計算時間の削減を行うことができた．また，同様に数式が複雑化することによって，通常だとシミュレーション不可能な例題もシミュレーションを行うことができた．高次項の削減は，興味のない微小値の n 次以上の項を削減するだけでなく，数式を簡単にすることで，HyLaGI のパフォーマンスの向上という効果もあった．

また，ニュートンのゆりかごのような特異点の解析の際に利用可能で，微小値が限りなく 0 に近いということを利用して，特異点近傍での運動を考察できた．同様に無限大や無限小を扱うような表現しにくいシステムもモデリング可能であると考えられる．

さらに，流される質点の運動のシミュレーションのように，初期値の誤差が軌道に与える影響を調べることができ，誤差の収束や拡大などシステムのもつ性質を分析した応用に利用できる．

5.2 まとめ

本論文では，HyLaGI において微小値としてパラメータ ε を特別に処理する機能の評価を行った．微小な値を扱う機能が HyLaGI のシミュレーション能力を向上できたことや実際にモデルに誤差が与える影響から，システムの解析に利用できることがわかった．

今後の課題として，4 章の文献 [2] 中の例題で，まだ解析できていない例題があるため，これを解析していきたいと思う．また，複数の微小値を用いた解析のために機能拡張を行い，その際にシステムの性質の解析や考察を容易にするために，それぞれの微小値について適切に出力する方法など考察していきたい．

謝辞 本研究を行うにあたり，例題作成に協力していただいた皆様，貴重な意見をいただいた上田研究室 HydLa 班の皆様に感謝する．本研究の一部は，科学研究費補助金 (基盤研究 (B) 26280024) の補助を得て行った．

参 考 文 献

[1] Borning, A., Freeman-Benson, B., Wilson, M. : Constraint Hierarchies, Lisp and Symbolic Computation, Vol.5, 1992, pp.221–268.

[2] Lee, Edward A.:Constructive Models of Discrete and Continuous Physical Phenomena, EECS Department, University of California, Berkeley, 2014, Feb, http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-15.html.

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

[4] 松本翔太, 上田和紀： ハイブリッド制約言語 HydLa の記号実行シミュレータ Hyrose, コンピュータソフトウェア, Vol.30, No.4, 2013, pp.18–35.

[5] 上田和紀, 石井大輔, 細部博史: 制約概念に基づくハイブリッドシステムモデリング言語 HydLa, SSV2008(第 5 回システム検証の科学技術シンポジウム), 2008.

[6] 上田和紀, 細部博史, 石井大輔 : ハイブリッド制約言語 HydLa の宣言的意味論, コンピュータソフトウェア, Vol.28, No.1, 2011, pp. 306–311.