実世界情報システムプロジェクト～アテンティブエンバイロンメント研究グループ～

プログラムの MLD システム表現と変数領域の検出

多治川　友之，田原　鉄也，新　誠一
東京大学大学院　情報理工学系研究科　システム情報学専攻

概要

近年のソフトウェアの複雑化や高度化によって、信頼できるソフトウェアを効率よく作り出す手法の開発が求められている。その１つとして、計算機による自動化や補助は非常に有効といえる。本研究ではプログラムの検証作業のために、従来の離散事象や論理構造に立脚したモデルではなく、状態方程式を拡張した区分多項式(PWP)システムや混合論理ダイナミカル(MLD)システムでプログラムを表現できることを示した。これによって、変数の値が取りうる領域を求める問題と、プログラムの特徴点を求める問題を混合整数計画問題として定式化することができた。

1 はじめに

現在の世の中のシステムの多くはコンピュータによって支えられている。一般に開発直後のプログラムには 100 行当たり１から 3 個のバグが潜んでいるといわれ[3]、その利便性と引き換えにソフトウェアの不具合によりシステム自体が停止してしまう危険性も増大している。そこで、本論文ではプログラムの検証を効果的に行う方法を提案する。

今までにもプログラムの検証問題は数多く考えられてきた。その目標は「プログラムが仕様と完全にあっているかを自動で検証する」ことである。その従来手法では、プログラムと仕様をモデル化してモデル同士を推論によって比較することで検証を行う。モデルの種類によって大きく２種類にわかれており、１つは仕様を時相論理でプログラムをオートマトンやペトリネットで表現したモデル検証系と呼ばれる手法である。もう１つは仕様を述語論理でプログラムを関数型プログラムで表現した証明検証系と呼ばれる手法である。

この従来手法の大きな特徴は、モデル化が非常に複雑なことである。論理的構造を意識したモデル化が必要であるため、一般の開発者には向かない。また、表現できるソフトウェアの範囲も限られている。従ってこれらの手法は、一般にはほとんど使われていないのが現状である。

現在は、テストケースを用意しておき仕様書による理想の結果と実際のプログラムを実行した結果を比較することで、バグを見つけ修正して質を高めていく。この方法では完全にプログラムが正しいことを証明できるわけではないが、値の比較という開発者にとって直感的にわかりやすい作業で行えるため、現実では広く行われている。

そこで本研究では、プログラムを値の変化が表現できるようなモデルに変換することで、値の比較という作業にも数学的な補助ができないかを考えた。そのモデルとして用いたのが区分多項式(PWP)システムや混合論理ダイナミカル(MLD)システムである。これによって、変数の値が取りうる領域を混合整数計画問題として解くことで数学的にその値を保証した。また、境界値や極値などの特徴的な点を取って結ぶことで、プログラムの入出力特性を少ない点のグラフで表現できることを示す。

2 ハイブリッドダイナミカルシステム

プログラムは x = y + 4 や z = x * 2 のような連続的な演算要素と、if～then～else～や while～do のような離散的な演算要素からなる。従って、プログラムの適切なモデル化には連続的な要素と離散的な要素の２つを表せるモデルが必要であり、そのようなモデルをハイブリッドダイナミカルシステムという。

ハイブリッドダイナミカルシステムには、離散的な要素をベースにしたものと連続的な要素をベースにしたものの二種類がある。前者のベースとなっているのは、オートマトンやペトリネットといった状態遷移を表現したもので、前章であげた検証問題などのように計算機科学ではよく用いられている。一方で、後者のモデルでは、状態方程式が基本となっており制御問題を数理計画

問題として定式化するときなどに使われている。本研究では後者のモデルを用いており、その中には離散要素を全て連続値関数で近似することで回路の検証問題に用いた前川らの例[4]があるが、ここでは離散的な要素も表現可能な区分多項式(PWP)システムや混合論理ダイナミカル(MLD)について説明する。

2.1 区分多項式(PWP)システム

区分多項式システムとは、通常の多項式による状態方程式を離散的につなぎ合わせたモデルで、

$$\begin{aligned} x[t+1] &= f_i(x[t],u[t]) \\ y[t] &= g_i(x[t],u[t]) \end{aligned}, for(x[t],u[t]) \in \chi_i \quad (1)$$

$$\chi_i = \left\{(x[t],u[t]) : \wedge_{j=1}^{n_i} c_i^j(x[t],u[t]) \le 0\right\}$$

という式で表せる。各領域 χ_i 内は多項式不等式 c_i による制約の積で表現され、多項式関数 f_i, g_i によってシステムの特徴が決まる。

代表的な例はスプライン関数（図１）である。３次のスプライン関数で、点の座標とその点での傾きを指定して３次関数で補完したものである。

2.2 混合論理ダイナミカル(MLD)システム

次に、混合論理ダイナミカル(MLD)システムについて述べる。これは Bemporad らが提案したシステム[5]で、前節の PWP システムのような切り替え要素を離散変数で表現することで、純粋な混合整数数理問題に変換していることが特徴である。なお、Bemporad らの論文では関数が線形のときのみ論じていたが、ここでは拡張して多項式として扱う。

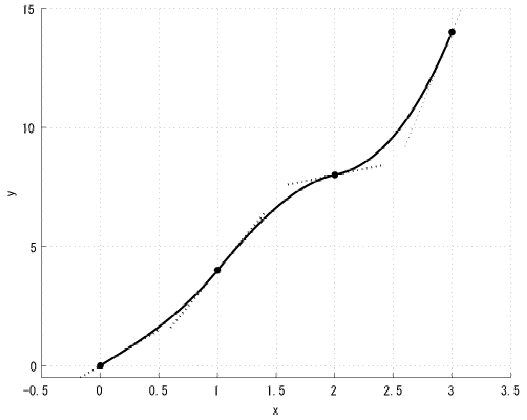

図１　スプライン関数

まず、論理演算と離散変数の関係を論じる。関数 $f : \mathrm{R}^n \to \mathrm{R}$ に対し命題 $X = \{f(x) \le 0\}$ を考える。x がある有界な領域内にあるとして、関数 $f(x)$ の最大値と最小値を M, m とする。この時、命題の成立非成立を、ある 0-1 変数 δ に対応させると

$$\{f(x) \le 0\} \leftrightarrow \{\delta = 1\} \Leftrightarrow \begin{cases} f(x) \le M(1-\delta) \\ f(x) \ge \varepsilon + (m-\varepsilon)\delta \end{cases}$$

と等価に変換できる。

これを利用すると、PWP の条件式 $c_i^j \le 0$ の成立非成立を、離散変数 δ_i^j と

$$\delta_i^j = 1 \leftrightarrow c_i^j \le 0$$

のように、対応付けると、この対応は

$$\begin{aligned} f(x) &\le M_i^j(1-\delta_i^j) \\ f(x) &\ge \varepsilon + (m_i^j - \varepsilon)\delta_i^j \end{aligned} \quad (2)$$

という不等式に変換される（ε は微小整数）。各領域はこの重ねあわせなので、離散変数 δ_i を

$$\delta_i = \prod_{j=1}^{n_i} \delta_i^j$$

と定義すると、この対応は

$$\begin{aligned} &-\delta_i^1 + \delta_i \le 0 \\ &\quad \mathrm{M} \\ &-\delta_i^{n_i} + \delta_i \le 0 \\ &\delta_i^1 + \Lambda\ \delta_i^{n_i} - \delta_i \le n_i - 1 \end{aligned} \quad (3)$$

という不等式に変換される。

これを用いると、PWP システムの式(1)は(2),(3)の不等式制約がついた

$$\begin{aligned} x[t+1] &= \sum_{i=1}^{n_i} \delta_i f_i(x[t],u[t]) \\ y[t] &= \sum_{i=1}^{n_i} \delta_i g_i(x[t],u[t]) \end{aligned} \quad (4)$$

という式で表現できる。これを MLD システム表現と呼ぶ。

3 プログラムの変換と解析

ここでは、プログラムを PWP システム及び MLD システムに変換し、解析を行っていく。

3.1 プログラムの変換

本論文で想定したプログラムの構文を BNF 記

法で表現したのが図２である。

さらに、図３のような前条件を定義しておくと、本研究では、このプログラムを実行した後の変数状態が全てPWP形式で表現できることを示した。さらに、入力変数と内部変数の最大値と最小値を与えておくことで、PWPシステム内にある条件式の最大値と最小値を求めて MLD システム(2),(3),(4)にも変形できる。

3.2 プログラムの解析

では、その MLD システム(2),(3),(4)を用いて解析を行っていく。まずは、変数領域が推定できる利点が挙げられる。変数領域は最大値と最小値によって与えられるため、

$$\max \sum_{i=1}^{n} \delta_i f_i(x,u), \min \sum_{i=1}^{n} \delta_i f_i(x,u)$$

$$\max \sum_{i=1}^{n} \delta_i g_i(x,u), \min \sum_{i=1}^{n} \delta_i g_i(x,u)$$

によって、混合整数計画問題として定式化できる。これによって、

- オーバーフローの危険性を確認できる
- 仕様で与えられた変数の値の範囲と一致するか数学的な検証ができる
- このプログラムを利用する際に安全かどうかの判断基準となる

という利点がある。

次に、プログラムの入出力特性を効果的に表せる点を数学的に求めることで、少ない点でもプログラムの特徴を表すことができる。

式:=入力変数、内部変数、定数と+,-,*による演算
ブール式:= 入力変数、内部変数、定数と<,>,<=,>=による演算
文:=skip | 内部変数:=式 | 出力変数:=式 |
　begin 文{ ;文 } end |
　if ブール式 then 文 else 文 |
　while ブール式 do 文 od
プログラム:=文

図２　プログラムの構文

前条件::= 出力変数:=定数 & 内部変数:=定数

図３　前条件

本研究では、その点として領域の境界値と極値を取った。グラフとして描くことを考えて、ここでは入力変数 u1,u2 に対する出力変数 y1 の関係を考え、他は固定値とする。また、u1,u2 の最小値をそれぞれ m_{u1}, m_{u2} と、最大値を M_{u1}, M_{u2} とする。

そのとき領域 i の境界値は、$\delta_i = 1$ を満たし、かつ

$$(m_{u1}, m_{u2}), (m_{u1}, M_{u2}), (M_{u1}, m_{u2}), (M_{u1}, M_{u2}),$$

$$u_1 = m_{u1} \wedge c_i^j(x,u) = 0, u_1 = M_{u1} \wedge c_i^j(x,u) = 0,$$

$$u_2 = m_{u1} \wedge c_i^j(x,u) = 0, u_2 = M_{u1} \wedge c_i^j(x,u) = 0,$$

$$c_i^{j_1}(x,u) = 0 \wedge c_i^{j_2}(x,u) = 0$$

を満たす点である。

また極値は、$\delta_i = 1$ を満たし、境界上での極値

$$u_1 = m_{u1} \wedge \frac{\partial}{\partial u_2} y_1(x,b) = 0,$$

$$u_1 = M_{u1} \wedge \frac{\partial}{\partial u_2} y_1(x,b) = 0,$$

$$u_2 = m_{u2} \wedge \frac{\partial}{\partial u_1} y_1(x,b) = 0,$$

$$u_2 = M_{u2} \wedge \frac{\partial}{\partial u_1} y_1(x,b) = 0$$

を満たす点か

$$\frac{\partial}{\partial u_1} y_1(x,b) = 0 \wedge \frac{\partial}{\partial u_2} y_1(x,b) = 0$$

を満たす点で与えられる。

3.3 実例

例として

```
double x1;

double func(double u1, double u2) {
  if (0.5 * (u2 - u1) >= x1 + 5)
    x1 = x1 + 5;
  else if (0.5 * (u2 - u1) <= x1 - 5)
    x1 = x1 - 5;
  else
    x1 = 0.5 * (u2 - u1);
  return x1;
}
```

というプログラムを考えると、このプログラムは

$$x1[t+1]=\begin{cases} x1+5 & u1,u2,x1\in\chi_1 \\ x1-5 & u1,u2,x1\in\chi_2 \\ 0.5u2-0.5u1 & u1,u2,x1\in\chi_3 \end{cases}$$

$$\chi_1 : x1+5-0.5u2+0.5u1\le 0$$

$$\chi_2 : -x1-5+0.5u2-0.5u1+\varepsilon\le 0 \wedge$$
$$-x1+5+0.5u2-0.5u1\le 0$$

$$\chi_3 : -x1-5+0.5u2-0.5u1+\varepsilon\le 0 \wedge$$
$$x1-5-0.5u2+0.5u1+\varepsilon\le 0$$

という PWP システムに変換される。変数の範囲を $0\le u1,u2\le 20, -10\le x1\le 10$ として MLD システムに変換するが、ここでは(2)(3)(4)の変形例で示した通りなので省略する。これを用いて解析を行うと、最大値と最小値は

$$-10\le x1[t+1]\le 10$$

で与えられる。これは MLD に変換する前に定めた仕様の変数の範囲通りであり、このプログラムの実行において変数の範囲が保たれることがわかる。また、u1,x1 に対する x1[t+1]のグラフを書くと図 4 のようになる。この例では各領域内のシステムは線形なので、極値をとる必要はないので省略されている。

3.4 問題点

以上でプログラムを PWP、MLD システムに変換して解析する手法を述べたが、ここには計算機による解探索上の問題が存在する。MLD システムに変換する際の条件式の最大値や最小値、変数の値の最大値、最小値の計算の際には、多項式計画問題、及び混合整数多項式計画問題を解く必要がある。また、極値や境界値を求める際も、多項式の求解問題に定式化される。

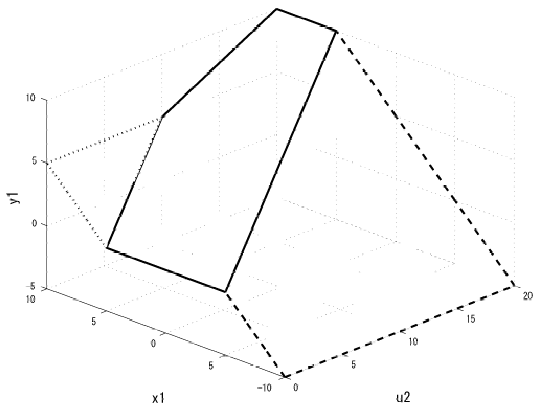

図 4　入出力関係

本研究で計画問題を解くのに用いたソルバーは MATLAB の Optimization Toolbox と Bemporad らが開発した MIQP.M[6]を用いているため、線形か 2 次までしか解けない。従って、計算機によって解析まで行うには、PWP システムに変換した際の各領域内のシステムの多項式が 2 次までである必要がある。

4 おわりに

本研究ではプログラムが状態方程式を拡張したシステムである PWP システム、MLD システムで表現できることを示した。さらに、MLD システム表現によって、変数の値の領域を求める問題を混合整数計画問題に定式化して、数学的な保証を加えることができた。最後に、プログラムの特徴を表す点として極値や領域の境界値を求める問題を多項式方程式の求解問題として定式化した。

現在は多項式が 2 次までのときしか解析できないが、今後はより高次の解析について、さらには解析のみならず設計などの諸問題にも適用していきたい。

参考文献

[1] T. Tajikawa, T. Tabaru and S. Shin : “Expression and Analysis of Algorithm via Dynamical System”, Proceeding of The 29th Annual Conference of the IEEE Industrial Electronics Society, pp. 1134-1139 (2003)

[2] 多治川, 田原, 新 :“プログラムの MLD システムへの変換と変数領域の検出”, 第 16 回計測自動制御学会 自律分散システムシンポジウム予稿集, pp. 91-96 (2004)

[3] 倉田, 石川 : “解説:無駄なく確実にテストする”, 日経システム構築, no. 124, pp.158-173 (2003)

[4] 前川, 萱嶋, 〆木, 新 :"論理値の関数近似を利用した順序回路テスト生成", 電子通信情報学会技術資料, FTS-91-62, pp. 7-12 (1992)

[5] A. Bemporad and M.Morari : “Control of systems integrating logic, dynamics, and constraints”, Automatica, vol. 35, no. 3, pp. 407-427 (1999)

[6] A. Bemporad and D. Mignone : “MIQP.M, A Matlab function for solving mixed integer quadratic programs”, ETH Zurich (2000). Code available at http://control.ethz.ch/~hybrid/