自動微分システムのためのテストプログラム生成

平野 勇次† 久保田 光一‡
中央大学大学院 理工学研究科 情報工学専攻†‡

要約: 自動微分システムは，プログラムを変換して別のプログラムを生成するものである．本研究では，この自動微分システムをテストするための入力となるプログラムの生成ツールを作成した．これは自動微分システムのためのさまざまなプログラムを簡単な入力から生成するツールであり，これにより自動微分システム構築の際のデバッグを容易にする．
キーワード: 自動微分，テストプログラム

1 背景

プリプロセッサタイプの自動微分システムはプログラムで表現された関数からその偏導関数値を計算するプログラムを生成するので，システムの構築には通常のコンパイラと同等の字句解析・構文解析処理が必要である．それだけでなく，生成されたプログラムが正しい偏導関数値を計算できているかを確認する作業も大きな手間となる．そのため，システムが正しく動作するかどうかを検証するためのテストデータとなるテストプログラムを作ることは，プログラムの構文上のテストだけではなく，数値計算上の数多くの条件を考慮しなければならず，容易ではない．

2 目的

本研究では，自動微分システム構築を容易にすることを目的として，自動微分システムを対象としたテストプログラム生成ツールの作成をする．

3 自動微分 [1]

自動微分とは関数の偏導関数値を計算するための技術で，ある関数値を計算するプログラムが与えられたときに，その偏導関数値を計算するプログラムを導出することである．導出方法は大きく分けてボトムアップ型とトップダウン型の 2 つがある．

3.1 ボトムアップ型自動微分

ボトムアップ型自動微分は基本演算を 1 回実行するたびに合成関数の微分則に従い偏導関数値を計算する．簡単な例を以下に示す．

```
ans = (v1 + v2) * v3 - 4 * v1;
```

のような式が与えられたとき，まず式を

```
z1 = v1 + v2;
z2 = z1 * v3;
z3 = 4 * v1;
ans = z2 - z3;
```

のように基本演算に分解する．また，各変数に関してその偏導関数値を格納する変数 (v1 なら dv1) を用意する．

そして，その基本演算ごとに作業用偏導関数値を計算することで，最終的な偏導関数値が計算される．

```
z1 = v1 + v2;
dz1 = dv1 + dv2;
z2 = z1 * v3;
dz2 = dz1 * v3 + z1 * dv3;
z3 = 4 * v1;
dz3 = 4 * dv1;
ans = z2 - z3;
dans = dz2 - dz3;
```

このとき dv1 の初期値を 1 とし，そのほかの偏導関数値を格納する作業用変数の初期値を 0 とすれば，v1 に関するの偏導関数値が計算される．

4 生成ツールの概要

テストプログラム生成ツールの概要および，それを用いた自動微分システムのテストの流れは図 1 のようになっている．

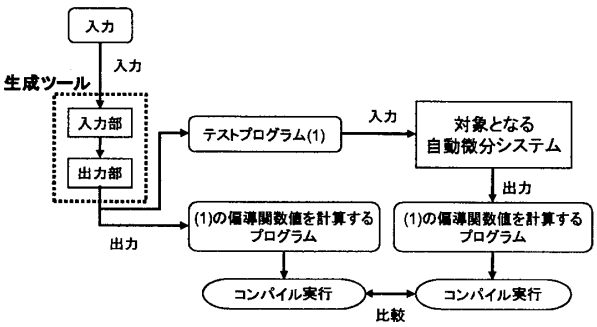

図 1 概要およびテストの流れ

本ツールではどのようなテストプログラムを生成するかを指示する入力を元にして，式計算を行うテストプログラム (1) と，その偏導関数値を計算するプログラム (2) の 2 つを生成する．そして自動微分システムに対して (1) を入力し，それにより生成されたプログラムのコンパイル実行結果と (2) のコンパイル実行結果とを比較することで，自動微分システムが正しく動作しているかを検証する．なお，現在のところ生成するテストプログラムは C 言語であり，テスト対象となる自動微分システムも C 言語を対象としている．

5 テストプログラムの持つべき要件 [4]

自動微分システムに対して行うテストとして，大きくわけて構文チェックと意味チェックの 2 つがある．本研究では，この 2 つのテストを簡単に行うことができるようなテストプログラムを生成する．

5.1 構文チェック

構文チェックでは，自動微分システムが正しく C 言語を読み取れているかを調べる．項目としてあげられるのは主として次の 2 つである．

- 変数や関数が宣言通りに処理され，それらを使用した式を読み取れているか
- 各構文の入れ子構造を正しく読み取り，処理できているか

Test Program Generator for Automatic Differentiation System
† Yuji HIRANO, Information and System Engineering Course, Graduate School of Science and Engineering, CHUO University
‡ Koichi KUBOTA, Information and System Engineering Course, Graduate School of Science and Engineering, CHUO University

5.2 意味チェック

意味チェックでは，自動微分システムが生成した偏導関数値を計算するプログラムの出力結果が正しいかどうかを調べる．そのためには，入力となるテストプログラムの偏導関数値の正しい値を知っていなければならない．

6 実装

実装は，Dev-C++ 4.9.9.2 を用いて C 言語で行った．

6.1 入力部

本ツールでは，型や構文といったものをテストプログラム中で網羅するのではなく，入力の際に使用したい型や構文を指定する形をとっている．そのため入力では，使用者がテストプログラムに必要な変数や関数，構文といったものを S 式で記述する．入力項目は次のようなものとなる．

- 使用する関数の型・名前・引数
- 関数内で使用する変数の型と数
- 関数内で使用する構文とその深さ
- 再帰等の複雑な関数の形の使用・不使用
- 使用する大域変数や，構造体で使用する変数の型と数
- 計算結果を出力するかどうか，ループの回数をカウントするかといったその他の機能

入力例を図 2 に示す．

```
1:(program              //program の中ですべて記述する
2: (func                //関数について記述をする
3:  (type double)       //型を指定する
4:   (name f1)          //名前を指定する
5:   (input             //入力変数 (引数) を記述する
6:    (int 2))          //int 型の変数を 2 つ
7:   (output            //出力変数 (戻り値) を記述する
8:    (double 1))       //double 型の変数を 1 つ
9:   (const             //構文を記述する (construct の略)
10:   (for 2(while 3))))) //for 文が 2 つ，その中に
                              while 文が 3 つ入れ子になる
```

図 2 入力例

6.2 出力部

出力部では，入力から使用する変数・関数や構文といったものを読み取り，それを元にテストプログラムを生成していく．

6.2.1 式・構文の生成

変数や関数はそれぞれ記号表を用いて管理する．記号表には変数ならその名前や型，式中で使用された回数，関数なら名前や型の他に，引数の型とその数，関数の中で使用する構文の種類と数といったものが格納される．

計算式は，構文の途中に必ず一つは存在し，使用される演算子や変数・関数はランダムに生成される．構文の条件式に関してはランダムではなく条件式のための変数を別に用意しており，それを用いて必ず一定回数ループするようになっている．

6.2.2 自動微分

テストプログラムを生成するのと同時にその偏導関数値を計算するプログラムも生成する．自動微分にはボトムアップ型を用いる．

偏導関数値を計算するプログラムはテストプログラムとほとんど同じ構成になっているが，計算式は大きく異なっている．

7 実験

実験として，吉岡が昨年度実装した自動微分システム [2][3] に対して，生成ツールを用いたテストを行った．

今回本ツールでは入力に以下のようなものを与え，そこから複数のテストプログラムを生成した（式生成はランダムなので，生成するたびに実行結果の異なるプログラムが出力される）．double 型の関数が 10 個で，それぞれが入力変数として double 型や int 型，そのポインタといったものを持つ．また，各関数内で (吉岡のシステムは入れ子の深さが 8 までしか対応していないためその範囲内で)，for や while といった構文が入れ子になって使用されている．また，いくつかの関数では，関数の中で別の関数が呼び出され，式の中で使用している．

このテストプログラムを吉岡のシステムに通したところ，途中でエラーが起きることなく偏導関数値を計算するプログラムが生成された．また，その生成されたプログラムのコンパイル実行結果と，本ツールが生成したテストプログラムの偏導関数値を計算するプログラムのコンパイル実行結果を比較した．その結果，ほとんどの値は一致したが，以下の場合において正しい値が出力されないことを確認した．

- 入力が int 型の変数のときに値が大きくなった場合，桁あふれが起きて正しい値が出力されない
- 計算途中で，`z1 = f1(x) - f1(x);` のように，計算結果に影響しない無駄な関数呼び出しがあり，かつその実引数 x がポインタだった場合，正しい値が出力されない

これより，基本的なプログラムに対しては吉岡のシステムは自動微分システムとして正しく動作していたが，上記のような特殊な例について，正しい偏導関数値を計算するプログラムが生成されないことを確認した．

8 まとめと今後の課題

自動微分システムを対象としたテストプログラム生成ツールを作成した．また本ツールを用いて既存の自動微分システムのテストを行い，基本的な処理が正しく行われていることを確認するとともに，正しいプログラムが生成されない例も見つけることができた．

これにより，自動微分システム構築の際のテストプログラムを作る手間を軽減することができたといえる．

今後の課題としては，本ツールでは 1 つの入力から 1 つのテストプログラムを生成したが，たとえば取り得る構文の入れ子構造を網羅したテストプログラムを生成するために，1 つの入力から複数のテストプログラムを生成できるようにする，といったことが考えられる．

謝辞

本研究を進めるにあたり研究室の同輩，後輩にはお世話になった．ここに記して感謝の意を表す．

参考文献

[1] 久保田光一，伊理正夫，“アルゴリズムの自動微分と応用”，コロナ社，1998 年
[2] 吉岡毅，“C プログラム遡行手続きによる高速自動微分の実装”，中央大学大学院理工学研究科情報工学専攻修士論文，2006 年
[3] 吉岡毅，久保田光一，“高速自動微分のための C 言語プリコンパイラ”，FIT2006 第 5 回情報科学技術フォーラム論文集，1，pp105-106，2006 年
[4] 内山 裕貴，引地 信行，石浦菜岐左，永松 祐二，“C コンパイラ用テストスイートおよびその生成ツール testgen”，VLSI 設計技術研究会，2007 年 1 月 17-18 日