関数の簡約化を用いたプログラムの正当性の自動検証アルゴリズム

2Q−05

平田雅之　岡野浩三　谷口健一
大阪大学 大学院基礎工学研究科 情報数理系専攻

1　まえがき

モデル検査などの形式的検証では，システムを(拡張)有限状態機械で表し，与えられた性質を満たすことを証明する[1-4]．一般に拡張有限状態機械(EFSM)の状態数が大きくなると現実的な時間で検証することが不可能となる．そのため状態数を少なくする必要がある．そこで，(1)プログラムの手続きおよび関数をそれぞれ単独で1つのEFSMで表し[5]，(2)いくつかの手続きおよび関数についてはEFSMを簡約化し，(3)その呼出し部の遷移を簡約化したEFSMで置換をおこない，最終的に状態数を減少させた1つのEFSMを作成し，検証を行うという手法を提案する．

2　検証の手順と検証性質

入力としてC言語のプログラムおよび検証したい性質を表す記述(CTL)が与えられる．検証手順としては，まずC言語のプログラムをEFSMに変換する．またそれにともなってCTL記述もEFSMに対応するように変換する．さらにCTL記述に現れる状態を含まない関数に対しては，5の関数の簡約化を用いることにより，関数に対応するEFSMをさらに状態数が少ないEFSMへと変換し，その関数呼出し部にあたる遷移と置換をおこなう．

本稿では，再帰関数を許していないので，関数呼出し部を含まない関数から置換を順次行うことにより，全ての関数をEFSMに置換することが可能である．

置換されたEFSMと検証したい性質を表すCTL式からプレスブルガー文を作成し[3]，そのプレスブルガー文の真偽判定を行うことにより，与えらたC言語のプログラムが性質を満たすかどうかを判定する．

検証できる性質としては，「プログラムの全ての入力と経路に対してある状態に到達することが可能であるか」や「プログラムのどんな入力や経路に対してもある状態に到達することがないか」などに限定しているので，手続きや関数を簡約化する際に関数がもつ情報の一部が失われたとしても，検証の結果が真であればプログラムでもその性質が成り立つことが保証される．

3　対象となるC言語のクラス

今回検証の対象とするC言語に対して次のような制限を加えた．

扱うことができる変数の型としてはint,long int,short int,charである．これらの型はすべてプレスブルガー文の整数変数として変換される．

また，構文としては大小比較式，加減算,if文,while文,for文,switch文を認める．その他手続きおよび関数における引数については値渡しのみを認める．また再帰関数は認めないものとする．

4　C言語から関数呼出しを含むEFSMへの変換

ここでは,C言語から関数呼出しを含むEFSMへの変換方法について述べる．

基本的な変換方法としては,C言語の命令文実行後の各状態にEFSMの各状態を割り当てる．さらに遷移後の変数値はC言語の代入文を用いて表す．

if文のEFSMへの変換は次の通りである．

if文の条件式を判定する状態から他の2つの状態(s_0,s_1)への遷移を構成し，各遷移の判定条件式としてif文の条件式を用いる．s_0,s_1より先の実行文についてはs_0,s_1それぞれ独立に部分EFSMに変換する．switch文を含む場合でも同様にEFSMに変換できる．

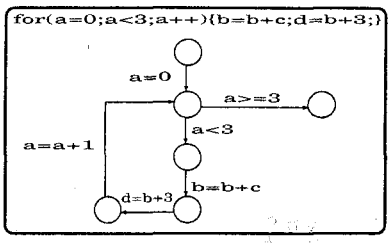

図 1: for文の変換

Automated Verification of Validity of Programs using Reduction of Functions
Masayuki Hirata, Kozo Okano and Kenichi Taniguchi
Graduate School of Engineering Science, Osaka University
Toyonaka-shi, Osaka, 560-8531, Japan

for 文の EFSM への変換は図 1 のようになる.

while 文の EFSM への変換は,for 文同様に閉路をもつ部分 EFSM となる.

5 関数の簡約化と置換

ここでは関数の簡約化の方法および関数呼出し部の置換方法について述べる.

まずは関数の簡約化について述べる. 対象関数内で値が変化するすべての変数を,引数および標準入力等の外部入力からなる式で以下のように表す.

- 変数値に引数および定数が入力されたとき

$$a{=}3 \quad \rightarrow \quad a{=}3$$

- 外部入力が行われるとき

$$getchar(a) \quad \rightarrow \quad a{=}input$$

なお外部入力が関わる演算が行われるときは外部入力のみで代用する.

$$\left.\begin{array}{rcl} f & = & input \\ a & = & f+x \end{array}\right\} \rightarrow a = input$$

今回 **2** で挙げたような性質のみを対象として検証を行うので, 上記のような値の変更を行っても, 検証結果が真であれば, 元のプログラムおいてもその性質が成り立つことが必ず保証できる.

- 変数の値が分岐により決定されるとき

分岐が静的に決定するときは分岐条件を含めて式を構成する.

$$\begin{array}{l}\text{if (a>0) b=1}\\ \text{else b=0}\end{array} \quad \rightarrow \quad b = \left\{\begin{array}{ll} 1 & a > 0 \\ 0 & a \leq 0 \end{array}\right.$$

分岐が動的に決定するときは分岐先でとりうる値全てを条件なしでとるものとする.

$$\begin{array}{l}\text{if (a>0) b=1}\\ \text{else b=0}\end{array} \quad \rightarrow \quad b = \left\{\begin{array}{l} 1 \\ 0 \end{array}\right.$$

これらの作業により, 関数の返り値に相当する変数の値を関数の引数および外部入力の式で表すことができ, **6** で述べるクラス制限よりその式は有限長になる.

次に関数呼出し部の置換方法について述べる. 呼出している関数における返り値を上記の方法により求め, その値が入力されるような部分 EFSM で関数呼出しの部の遷移を置換する. 図 2 では, 返り値 r が左のように求まったときに, 関数呼出し部を右のように置換している.

6 自動検証のためのクラス制限

検証時に EFSM における閉路を無限回調べることがないように以下のような制限を加えた.

$funcA$ の返り値 r

$$r = \left\{\begin{array}{ll} r1 & cond1 \\ r2 & cond2 \\ input & cond3 \end{array}\right. \rightarrow$$

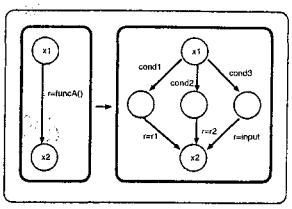

図 2: 関数の返り値の置換

for 文では, 制御変数の初期値は定数のみ, 条件式では, 定数との比較のみ, 制御変数の変更は定数の増減のみを許している. この制限により for 文の閉路は単純に定数回調べればよいことがわかる.

while 文でも, 条件式においては単一の変数と定数との比較のみを許している. さらに条件式に現れる変数は, while 文における閉路内において必ず外部入力をとらなければならない. この条件により閉路を脱出するかどうかはその変数値にのみ依存し, この閉路については高々2 回調べればよい [3].

7 例題

例題としてオークションシステムへの適用を考えている. このシステムでは, ふたりの参加者が互いに入札をくり返し、片方があきらめるまで入札をくり返す. このシステムで以下のような性質の検証を考えている.

- ふたりのうち高い金額の入札を行った人が落札する権利を得る
- 低い金額の入札を行った人が落札することはない

8 あとがき

本稿では, プログラム中の関数単位で検証できるアルゴリズムについて述べた. 今後の課題としては, ポインタの使用を許す等の C 言語のクラスの拡張が考えられる.

参考文献

[1] Clarke, E. M., Grumberg, O. and Peled, D. A.: *Model Checking*, The MIT Press (1999).

[2] Chan, W., Anderson, R., Beame, P. and Notkin, D.: Combining Constraint Solving and Symbolic Model Checking for a Class of Systems with Non-linear Constraints, in *Proc. of 9th Int'l Conf. on CAV*, Vol. 1254 of *LNCS*, pp. 316–327, Springer-Verlag (1997).

[3] 竹中崇, 岡野浩三, 東野輝夫, 谷口健一:整数入力値を保持するレジスタをもつ EFSM に対する記号モデル検査アルゴリズム, 第 13 回 回路とシステム軽井沢ワークショップ 論文集, pp. 555–560 (2000).

[4] Bouajjani, A., Jonsson, B., Nilsson, M.,and Tayssir-Touili:Regular Model Checking , in *CAV 2000*, Vol. 1855 of *LNCS*,pp. 403–418 (2000).

[5] Mauri, G., and Scheer, S.: Constructing Reliable Embedded Systems Using the Results of System Safty Analysis , in*Ada-Europe 2000*,Vol. 1845 of *LNCS*,pp. 173–184 (2000)