1N-4

文字列一致による数学的等価性判定可能なモデル分割手法

三鍋孝介†
電気通信大学大学院情報理工学研究科総合情報学専攻

織田 健‡
電気通信大学情報理工学部総合情報学科

1 はじめに

ソフトウェア部品を用いた開発手法は信頼性の確保の手段として古くから提案されているが、部品作成の手間、部品検索の手間、部品再利用の手間が問題となっている。我々は形式手法を用いた部品の仕様を使うことで部品検索の手間を削減、また既存のソフトウェアを分解することで部品製作の手間を減らす手法について提案してきた[1]。本稿では部品の仕様をソフトウェアモデルの分解により得る方法について提案する。

2 背景・目的

2.1 形式手法 B Method

形式手法とは数学を基盤とする形式的な言語を用いて仕様を記述することで齟齬をなくし、ミスを減らそうとする手法である。B Method はその形式手法の一つで、ソフトウェアの仕様を集合論や命題論理を用いてモデルとして記述する[2]。このモデルを拡張してゆき、最終的に手続的に記述された実装を作る。どちらも数学的に書かれているため、B Method ではモデルの無矛盾性と、モデルと実装間の整合性を証明することができる。つまり、無矛盾なモデルと整合性のとれた実装を作ることができればその実装は無矛盾な信頼性の高い実装であると言える。また、定理証明器による支援が用意されている。

2.2 ソフトウェア部品

我々の手法の部品は、B Method で記述された実装と仕様のセットになっている。この部品は部品の機能をモデルから読み取ることができ、証明によって部品の信頼性を保証することができる。そして部品の粒度は多くのソフトウェアで再利用できるように細かくする。細粒度なおかつ形式手法によって記述される部品は制作に手間がかかるが、B Method で記述されたソフトウェアのモデルと実装を分割することで部品とその仕様を得る。

2.3 字面統一した仕様による検索

前述のソフトウェア部品には数学的に書かれた仕様がついているため、新規ソフトウェアの要求を数学的に記述すれば定理証明によって健全性の高い検索が可能である。ただし、粒度の小さい部品一つ一つに対して定理証明を行う場合その計算時間は膨大なものとなってしまう。そこで、本研究では仕様や要求に対して構文書き換えを行い、おなじ機能はすべて同じ字面になるようにした上で、テキストの全文一致検索を行う。これにより健全性を保ったまま効率的な検索が可能となる。

A Model slicing method for mathematical equivalence check by string matching

†Kousuke Minabe, The University of Electro-Communications Graduate school of Informatics

‡Takeshi Oda, The University of Electro-Communications Graduate school of Informatics

	演算子	定義
1	$seq_1(s)$	$seq(s) - \{[]\}$
2	$u - v$	$\{x \mid x \in u \& x \notin v\}$
3	$s \leftarrow t$	$\{x \mid x \in s \leftrightarrow t \,\&$ $dom(x) \quad = s \,\&\, dom(x^{-1}) \in t)\}$
4	$s \leftrightarrow t$	$\mathbb{P}(s \times t)$
a	$u \in \{x \mid x \in v\}$	$u \in v$
b	$u \in \{v\}$	$u = v$

表 1: プリミティブ化規則（抜粋）

2.4 モデル分割の難点

モデルは実装の動作を表す操作と、操作に関わる制約の二つからなる。図 1 は実際の B Method の記述を抜粋したものだが、INVARIANT(不変条件) や PRE(事前条件) 等が操作の制約に当たる。制約によりモデルは高い信頼性を得ているため、高信頼なモデルを分割により得るには操作の分割と必要な制約条件の抽出の二つのステップが必要となる。操作は決まった粒度毎に分割し、制約条件は操作に使われている変数に注目して抽出する。このとき、信頼性の保証のため注目した変数以外を含む式は抽出しない。しかし制約条件には明示されていないが推論によって成り立つ条件 (暗黙の条件) が存在し、他の変数を使った式から推論される暗黙の条件の中に必要な制約条件が含まれる場合がある。そのため、予め暗黙の条件を全て明示した上で操作の分割を行う必要が有る。

2.5 目標

本稿では 2.2 節の部品の仕様をソフトウェアのモデルを分解して得る手法について述べる。その手法で得たモデルは以下の条件を満たさなければならない。

- 同じ機能を表現するモデルは全て同じ字面である
- 必要な条件を取り逃さない

3 モデル分割手法

モデル分割では、プリミティブ化と推論の二つを行い暗黙の条件を全て書き出した後モデルを分割する。この後、字面を等しくするため標準化を行う。

3.1 暗黙の条件の書き出し

プリミティブ化はモデル内の式に使われる演算子に対して演算子の定義に沿って書き換えを行い、演算子の種類を制限する。また、可換な演算子は全てどちらか一方に制限する。こうすることで異なる演算子による表現がなくなるほか、一つの演算子が持っていた意味を全て明示することができる。変換は表 1 のような変換規則を再帰的に式に適用することで行う。変換を再帰的に行った場合二重否定や括弧入れ子になるため、同時にそれらを解消する変換も行う。表 1 の a, b がこのような式の単純化のためのルールである。演算子の種類は減る一方、演算子の数は増える方向に変換するため、文の長さは冗長になってしまい人が読むには不向きとなるが、分割操作は計算機が行うため問題は無い。この後、推論を行い残りの暗黙の条件を導く。この推論には推移律や包含関係などに基づいた推論を行う。この推論では式の数が増

```
INVARIANT
    users ⊆ 0..maxUser &
    userName ∈ users → seq(0..255) &
    [] ∉ ran(userName) &
OPERATIONS
    uid ← registUser(name) =
    PRE
        max(users) ≤ maxUser - 1 &
        name ∈ seq1(0..255) · · · (a)
    THEN
        ANY newUser
        WHERE
            newUser ∈ 0..maxUser &
            newUser ∉ users
        THEN
            users := users ∪ {newUser} ||
            userName := userName < +
                {newUser ↦ name} · · · (b)||
            uid := newUser
        END
    END;
```

図 1: 例題モデル（抜粋）

えていくことになるが、同じ意味の式は全て同じ字面になっているため、それらを削除すると、推論しても式の数が増えなくなる。増えなくなったとき推論が全て終わったと判断し、推論を停止する。

3.2 モデルの分割

モデルの分割は基本的に一操作毎に分割する。但し、if 文や select 文など条件分岐を含むものは、分岐後の操作が排他的であるときはそれぞれの場合にわけ、排他的でない条件分岐は分岐後の操作もまとめて一操作とする。その後、分割操作中の変数のみの制約条件を抽出する。

3.3 標準化

プリミティブ化により使える演算子を制限したため、同じ意味を表す式は同じ数の変数や同じ演算子を使う。そのため、字面の統一は構文の並び替えだけですむ。式を構文木に直し、各変数や各演算子に重みをつけて各項の重さを決める。より重い項が左に、より重い式が上になるように構文木を並び替える。演算子の重みは予め決めているが、変数の重みは同じにしておき演算子の重みによって式順が決定した変数から重みを変更する。この時同じ順番に出現する変数はその次に出現する順番で重みをつける。この作業を繰り返し変数の重みを決定する。

4 手法適用例

本節では提案手法の適用例を示す。分割するモデルは、分割で主に操作を行う部分を抽出した図 1 を用いる。

4.1 暗黙の条件の展開

まず暗黙の条件を展開する。例えば図 1 の (a) を展開すると、表 1 の 1 の推論より以下のように書き換わる。

$$name \in seq(0..255) - \{[\ ]\} \tag{1}$$

同様に 2, a, b の順に項書き換える。

$$name \in \{x \mid x \in seq(0..255) \& x \notin \{[\ ]\}\} \tag{2}$$
$$name \in seq(0..255) \& name \notin \{[\ ]\} \tag{3}$$
$$name \in seq(0..255) \& name \neq [\ ] \tag{4}$$

結果、(a) 式は $name \in \mathrm{seq}(0..255)$ と $name \neq []$ の二つの式になる。このような書き換えは他の式にも行われ、不変条件は図 2 のようになる。プリミティブ化後に制約条件から推論を行い、同じようにプリミティブな式を推論する。ここでは (c) と (d) 式に注目すると $userName \in 0..maxUser \leftarrow seq(0..255)$ を推論できる。

```
INVARIANT
    users ⊆ 0..maxUser & · · · (c)
    userName ∈ ℙ (users × seq(0..255)) & · · · (d)
    userName ∈ ℙ (0..maxUser × seq(0..255)) &
    dom(userName) ∈ ℙ (users)&
    dom(userName) ∈ ℙ (0..maxUser)&
    dou(userName^-1) ∈ ℙ (seq(0..255))&
    [] ∉ dom(userName^-1) &
```

図 2: プリミティブ化後の不変条件

```
INVARIANT
    v001 ⊆ 0..v002 ↔ seq(0..255) &
    dom(v001) ⊆ 0..v002&
    dou(v001^-1) ⊆ seq(0..255)&
    [] ∉ dom(v001^-1) &
OPERATIONS
    out ← op(v003) =
    PRE
        v003 ∈ seq(0..255)
        v003 ≠ []
    THEN
        ANY v004
        WHERE
            v004 ∈ 0..v002 &
        THEN
            v001 := v001 < + {v004 ↦ v003}
        END
    END;
```

図 3: 標準化後

4.2 操作分割

次に排他的な操作を分割する。今回の操作に使われている同時代入は排他的な操作なためそのまま分割するだけで良い。ここでは図 1 の (b) 式に注目する。ここで使われている変数や定数は userName, maxUser, user, name であるため、これらのみを含んだ式を抽出する。

4.3 標準化

必要な式を抽出した後、並び替えを行う。変数及び定数の出現順に v00x と名前を付け替え、操作名を op に、出力名を out にする。最終的に分割した図 1 の (b) 式に注目して分割したモデルは図 3 になる。このモデルでは userName, maxUser, name, newUser がそれぞれ v001, 002, v003, v004 に付け替えられている。

5 まとめ

本稿では形式手法を用いた部品合成を目的とした部品の生成のため、モデルを分割する手法について述べた。このモデルではプリミティブ化により演算子の種類を減らして推論することにより同じ構文になることを狙っている。無矛盾なモデルを作れるかは暗黙の条件が全て書き出せたかどうかにかかっている。今後はプリミティブ化のルールや推論のルールからどの程度の表現を同じ字面にできるかを明示することが課題となる。

参考文献

[1] 中村丈洋, 織田健. B method における自動コード合成フレームワークの提案. 情報処理学会研究報告. ソフトウェア工学研究会報告, Vol. 2010, No. 18, pp. 1–8, 2010.

[2] J-R Abrial. *The B-Book:Assigning Programs to Meanings.* thePress Syndicate of the University of Cambridge, 1996.