B-009

形式的仕様記述における制約条件の欠落推定

岡野 純平 †
電気通信大学大学院情報理工学研究科情報学専攻

織田 健 ‡
電気通信大学大学院情報理工学研究科情報学専攻

1 はじめに

ソフトウェア開発手法の 1 つに形式手法がある。形式手法は要求を数学的に記述することで仕様内の無矛盾性を証明できるため、高信頼なソフトウェア開発を行える。しかし、システムが常に満たすべき条件 (不変条件) の記述に誤りがある場合、証明が適切に行えず利用者の考える振る舞いと異なるソフトウェアが開発されてしまう。

本研究では形式手法の 1 つ B-Method とソフトウェアの振る舞いを視覚的に記述する状態遷移図の 2 つの異なる仕様記述を比較することで、B-Method のモデルと利用者の想定する振る舞いが一致しない場合においてモデル内の誤りを推定する手法を提案する。

2 背景と目的

2.1 B-Method

形式手法 B-Method は、要求を集合論と述語論理を用いて記述し、それを段階的に詳細化することで実装を導出する [1]。以降 B-Method の仕様をモデルとする。モデルは大きく分けて不変条件と操作の 2 つで構成され、モデルの操作実行後に不変条件を満たすか数学的に証明することで無矛盾性を保証できる。

2.2 本研究の目的

利用者の想定している振る舞いと一致しない原因として以下の点が挙げられる。

- 関係、関数における型宣言の誤り
- 不変条件の欠落
- 余分な事前条件の記述
- 操作の欠落

これらの誤りは数学的な矛盾が存在しない場合があるため、B-Method の検証では発見できない。

本研究の目的は B-Method と状態遷移図の 2 つの記述を用いることで、上記のような証明では発見できない誤りを発見することである。

3 モデルと状態遷移図のマッチング

2 つの仕様記述の整合性は、B-Method のモデルから状態遷移図を導出し、2 つの状態遷移図が一致するかで評価する。本章ではそのための手法を提案する。

3.1 本手法における状態遷移図の位置づけ

利用者はエンティティ毎に状態遷移図を記述する。エンティティは仕様内の実体を示し、モデルの集合の型に対応する。各エンティティはモデルの型と、エンティティ内の状態はその型の部分集合と、遷移は操作と対応させることで状態遷移図とモデルのマッチングを行う。

†Jumpei Okano, The University of Electro-Communications department school of Informatics
‡Takeshi Oda, The University of Electro-Communications graduate school of Informatics

3.2 モデルから状態遷移図の導出

モデルから状態遷移図を導出する手順は以下である。

手順 1 型とエンティティの対応付け (人の手)
手順 2 モデルの型に内包された集合の包含関係の計算
手順 3 互いに素な集合をモデルの状態として導出
手順 4 状態遷移図内の状態とモデルの状態数の確認
手順 5 モデルの操作による状態遷移の導出
手順 6 状態のマッチング
手順 7 状態遷移のマッチング

手順 1 でモデルで定義された型と状態対遷移図のエンティティを対応させる。手順 2,3,4 で状態数について 2 つの仕様を比較し、仕様の検証を行う。その後、手順 5,6,7 で状態と操作のマッチングを行い、再び仕様の検証を行う。

そのため本研究では手順 5,6,7 以降のマッチングでは状態数は一致しているものとして扱う。

3.2.1 状態に対応する集合の導出

モデルから導出する状態は互いに素な集合に対応する。よって集合間の関係の計算が必要である。

モデルにおける集合とは利用者が定義した関数以外の変数及び関数の定義域と値域に当たる。集合間の関係は関数の定義域や値域を導出するルール、及び集合に関する不変条件を用いることで求められる。その後、重なり合った集合間の差や積集合を計算することで、状態に当たる互いに素な集合を導出できる。

3.3 2 つの状態遷移図のマッチング

ここでは手順 5,6,7 のマッチング手順を記述する。遷移の導出は事前条件が含む状態を始点、操作実行後の事後状態が含む状態を終点として、要素の移動が発生しうる全ての組み合わせを遷移として導出する。

状態のマッチングはモデルから導出された状態と利用者が記述した状態遷移図の状態の全ての組み合わせに対して、一致する遷移が多い組み合わせを結果とする。遷移のマッチングは各操作名とイベント名に対して行い、遷移の数及び遷移元と遷移先の組み合わせの一致が一番多い組み合わせを結果とする。以降の遷移の欠落や余分な遷移とは各イベント毎に対してのものである。

4 モデルの誤り推定

モデルと状態遷移図のマッチングによって 2 つの仕様に整合性が取れない場合どちらかの仕様に誤りが存在すると考えられる。状態遷移図に誤りがある場合は利用者にその個所を通知するだけで解決を図れるため、本研究ではモデルに誤りがある場合にその原因がどこに起因するかを推定する手法の提案をする。

モデルの誤り検出は状態数の不一致、遷移の不一致の順番で行う。そのため遷移の一致を比較する際には状態数は一致しているものとして扱う。

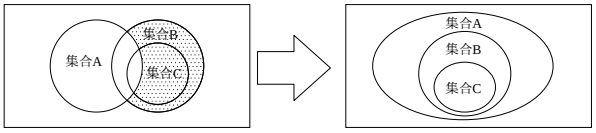

図 1:$B \subseteq A$ の欠落における排除される状態

4.1 状態数が一致しない場合の誤り

状態数が一致しない場合、その原因として集合間に余分な重なりが生じていることが挙げれる。そのため集合間の関係に以下の条件の欠落が存在すると考えられる。

条件 1 $A \subseteq B$, $B \subseteq A$
条件 2 $A \cap B = \emptyset$
条件 3 $A = B$

以降ではそれぞれの条件を条件 1,2,3 として示す。

4.2 欠落した不変条件推定手順

2 つの状態遷移図の状態数が一致しない場合、不変条件の欠落が考えられる。その場合、集合間に余分な重なりが生じるため状態数は必ず増える。そのためモデルの状態数の方が多くなる。状態数が一致しない場合は以下の手順で欠落した不変条件を推定する。

1. モデルから導出した状態数と利用者が記述した状態数の差を計算し、評価値とする。
2. 各集合間に対して条件 1,2,3 を欠落した可能性のある条件として列挙する (モデル内にある条件は除く)。
3. 2 の手順で導出された条件の追加により、排除される集合を列挙する。
4. 2 の条件を組み合わせ、3 に含まれる集合の種類が評価値なる組み合わせを欠落した条件と推定する。

手順 2 では各集合の組合わせに対してそれぞれ条件 1,2,3 を求める。その際、条件 2 において集合の組合わせは 2 つとは限らず、3 つ以上の場合も考慮する。これは $A \cap B \cap C = \emptyset$ は $A \cap B = \emptyset \wedge B \cap C = \emptyset$ と一致しないためである。以上により欠落可能性のある条件を列挙する。

手順 3 では手順 2 で導出された条件を追加した場合、その条件によって排除される状態を導出する。条件 1 で $B \subseteq A$ の状態を追加した場合、$B - A$ という集合に含まれる状態が排除される。例えば図 1 では集合 $C - A$ と $(B - C) - A$ が排除される。また条件 2 の場合はその条件に内包されている状態が全て排除される。$A \cap B = \emptyset$ では $A \cap B$ 内に存在する状態が排除される。条件 3 で $A = B$ を追加する場合、$A - B$ と $B - A$ に含まれている状態が排除される。

手順 4 では、欠落した条件の組み合わせを推定する。評価値は 2 つの状態遷移図の状態数の差のため、手順 3 で求めた排除される状態の数と評価値が一致する状態の組み合わせが欠落した条件となる。この際、複数の組み合わせが残る場合が存在する。その場合は利用者にその条件の組み合わせを提示し、B-Method の検証器を用いることで各条件の組み合わせを不変条件に組み込んだ際に無矛盾性が保証できるかを検証することで決定する。

4.3 遷移の不一致による誤り推定

モデルから導出した状態遷移図の遷移が一致しない場合は以下のケースが考えられる。

1. 自己遷移による不一致
2. モデルから導出した遷移の方が少ない

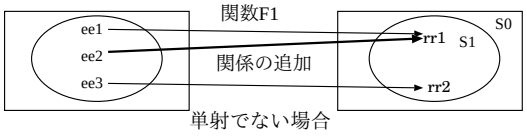

図 2:関数の減算における自己遷移の有無

3. モデルから導出した遷移の方が多い

状態遷移図には存在しない自己遷移が導出された場合、それは関数の型定義に誤りがあると考えられる。モデルの型定義が単射でない場合、その関数に写像を追加、もしくは減算してもその値域が変化しない場合が存在するため自己遷移が発生する。図 2 の場合、関数 $F1$ に $\{ee2 \mapsto rr1\}$ を追加しても値域の数は変わらないため自己遷移が発生する。しかし単射の場合自己遷移は発生しない。そのため自己遷移が不一致の場合は、本来単射で記述すべき関数をそうでない関数で記述していると考えられる。

ケース 2 では事前条件が多すぎる場合と操作そのものが欠落している場合が考えられる。操作そのものが欠落している場合は名前の対応関係が取れないため発見は容易なので、ここでは割愛する。事前条件が多すぎる場合、操作が実行される状態が狭まるため本来あるべき操作が実行されず、事後状態が不変条件を満たすことになるため証明では発見できない。そのため遷移が足りない場合は遷移の始点にあたる状態を確認し、その集合をガードしている条件を探索することで余分な事前条件を発見できる。

ケース 3 では事前条件の不足が考えられる。事前条件が不足した場合は条件が緩くなり、余分な遷移が導出される。しかし本手法では不変条件の誤り発見の後、モデルの無矛盾性を再検証する。そのため事前条件に欠落がある場合は、不変条件の誤りを発見した後に B-Method の証明器で事前条件の欠落は解決できる。今後はこの仮定が妥当かどうかを検証することが課題である。

5 考察

本研究ではモデルと状態遷移図に整合性が取れない場合のモデルの誤りを推定する手法の提案を行った。

関数、関係における誤りや不変条件の欠落の推定は本手法で網羅できていると考えている。また余分な事前条件の発見についてはアウトラインの提案をしたのみなので具体的なアルゴリズムの構築が今後の課題である。

操作の欠落に関してはマッチングの際に名前の対応づけられない遷移が現れると考えられる。そのため、操作の欠落の発見は容易である。ただし遷移のマッチングについては具体的なアルゴリズムが考案できていないため、現段階では操作の欠落についての提案は行えていない。

6 終わりに

本研究では B-Mehtod の証明では発見できない仕様の誤りについて状態遷移図とのマッチングにより誤りの原因を推定する手法を提案した。しかし遷移のマッチングの具体的なアルゴリズムの提案、及び全体を通した評価実験はまだ行えていないため今後の課題となる。

参考文献

[1] J-R Abrial, *The B-Book*CAMBRIDGE UNIVERSITY PRESS