B-018

エンティティの振舞いに着目したZによる仕様記述と状態遷移規則の比較に基づく誤り検出法

A Error Detecting Method based on the Comparison of Z Specification and State Transition Focusing Entities' Dynamics

楊 洋‡ Yang Yang　織田 健† Takeshi Oda

1 始めに

ソフトウェア開発の生産性や品質を向上させるため、要求仕様の誤りを減らすことが有効である。形式的仕様記述言語を用いると、曖昧さは除かれ、矛盾は数学的に検出できるが、要求仕様中の欠落は検出できない。そこで、我々は状態遷移規則と形式的仕様を比較することにより欠落を防ぐ手法を提案した [1]。この従来手法は欠落の含む誤りを検出でき、誤りの原因もスキーマ単位まで提示できるが、検証には複雑な入力が必要で、具体的に誤りの発生箇所と修正方法を提示できなかった。

本研究では入力を簡単化でき、具体的に誤りの発生箇所と修正方法を推論できる手法を提案する。

2 研究の背景

2.1 仕様記述言語 Z

Z は集合論と述語論理に基づく仕様記述言語である。システムの抽象状態を表す状態スキーマとシステムでの操作を表す操作スキーマによりシステムを記述する。状態スキーマの中にシステムの状態が常に満足しなくてはならない制約 (システム状態不変条件、以下では単に状態不変条件) が記述され、操作スキーマでは操作前後の変数間の関係を記述することで操作による集合の変化を表す。操作スキーマには操作を適用するための制約条件を示す述語 (制約式) と操作後の変数の値を規定する述語 (規定式) がある。なお、以下では Z により記述された仕様を単に Z 仕様とする。

2.2 Z 仕様の比較対象とする状態遷移規則

比較に用いる状態遷移規則はシステムを構成するオブジェクトを表すエンティティの種類ごとに用意し、Z 仕様と比較するため、遷移条件とエラーの出力を Z 記法で記述する。なお、事前に用意した Z 仕様と比較する状態遷移規則を要求定義からの状態遷移規則とする。

2.3 従来の誤り検出法

従来手法では、同一の誤りが混入されないように、Z と異なる性質の状態遷移規則を導入した。Z 仕様において、要素がある集合に属しているかいないかという静的な側面から操作スキーマにより集合間に要素の移動が発生したこととしてとらえ、その要素移動を状態遷移規則における状態遷移 (動的な側面) と比較することで、不一致を誤りとして検出した。さらに効果を高めるため、Z 仕様と状態遷移規則を情報交換のない状態で 2 つの開発者グループで記述し、それぞれの要素を対応付けて比較を行う (図 1)。

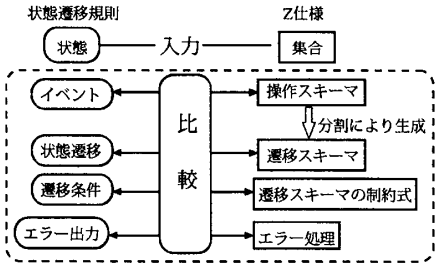

図 1：従来の誤り検出法

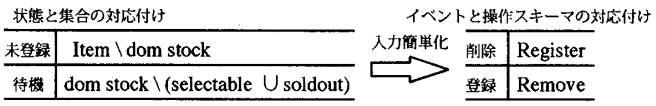

図 2：入力簡単化の例

しかし、従来手法においては、Z 仕様の誤りをスキーマ単位まで検出でき、比較対象となる状態遷移規則の誤りも検出できたが、状態と集合を対応付ける際に、人間は状態に対応する集合を Z 仕様から抽出する作業が極めて複雑で、誤りが混入する可能性がとても高い。さらに誤りが検出されてもその誤りの具体的な発生箇所と修正方法を推論できない。

3 入力簡単化

3.1 着目点

状態と集合の対応付けより、イベントと操作スキーマの対応付けが単純で、誤りの混入する可能性が低い。図 2 は自販機システム例の一部であり、検証者にとっては、状態と集合の対応付けよりイベントと操作スキーマの対応付けの方が簡単である。

Z 仕様中のデータ型 S に対して、状態スキーマ中の状態不変条件はデータ型の集合関係を一意に定めているので、状態不変条件から互いに素の集合を導出でき、これらの集合をデータ型 S の実体であるエンティティの状態に対応しているではないかと考えられる。多くの例を検証した上で、状態不変条件から導出した互い素の集合は状態遷移規則の状態に対応していることが発見した。例えば、あるシステム状態不変条件が定めた集合関係をベン図 (図 3) で表現すれば、S_1, S_2, $S_3 \setminus (S_1 \cup S_2)$ の 3 つの集合が導出でき、エンティティの状態に対応する。これにより、イベントとスキーマの対応付けから状態と集合の対応付けを推論できる。但し、状態や集合が欠落した場合、イベントと操作スキーマの対応付けから正しく状態と集合の対応付けを推論できない。そこで、イベントとスキーマを対応付ける前に、状態と集合の数が一致

†電気通信大学 情報通信工学科
‡電気通信大学 電気通信学研究科 情報通信工学専攻

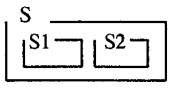

図 3：状態スキーマから状態に対応する集合の関係ベン図

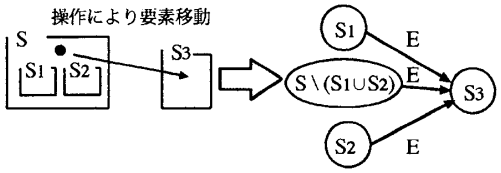

図 4：集合の要素移動から状態間の状態遷移の導出

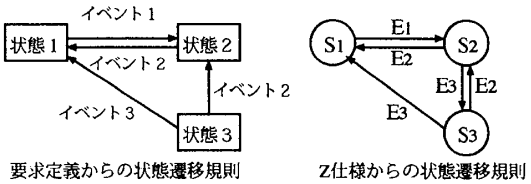

図 5：誤りの例

しているかどうかを検証する必要がある。

3.2 具体的な手順

まず検証したい Z 仕様中のデータ型と要求定義からの状態遷移規則の対応付けを検証者が行う。次に、状態スキーマの中のシステム不変式からそのデータ型の互いに素の集合を導出し、2 つのモデルの状態数が一致していれば、検証者は全てのイベントに対応する操作スキーマを入力する。

操作スキーマによる要素の移動をエンティティの状態間の状態遷移ととらえるので、図 4 のように、集合 S から集合 S_3 へ操作スキーマによる要素移動が発生すれば、集合 S_1, S_2, $S \setminus (S_1 \cup S_2)$, S_3 に対応する状態の間に、操作スキーマに対応するイベント E により状態遷移が発生したと考えられる。このように、検証したいデータ型の実体であるエンティティの状態遷移規則を Z 仕様から導出できる。この Z 仕様からの状態遷移規則を要求定義からの状態遷移規則と比較すれば、従来手法と同様に誤りを検出できる。

4 誤りに関する推論

4.1 着目点

状態間の状態遷移は状態に対応する集合間の要素移動を表現しているので、2 つの状態遷移規則を比較することで、イベントにより生じた状態遷移規則が不一致であれば、状態遷移の有無から操作スキーマによる集合間の要素移動の有無を推論できる。

4.2 具体的な手順

2 つの状態遷移規則が一致していなければ、どちらかのモデルに誤りが存在すると仮定し、その誤りの具体的な発生箇所と修正方法を推論する。

4.3 誤りの推論

状態の数が不一致であれば、要求定義からの状態遷移規則の状態数が妥当でないか、Z 仕様の状態スキーマの中にシステム状態不変式が欠落したかを推論できる。要求定義からの状態遷移規則が正しいと仮定すれば、集合間の関係から欠落したシステム状態不変式を提示できる。

状態間の状態遷移に不一致があれば、その状態遷移のイベントに対応する操作スキーマに誤りがあると判定できる。図 5 では 2 つの状態遷移規則の間に、状態 S_2 から S_3 への状態遷移が一致していない。要求定義からの状態遷移規則にイベント 3 が起こした状態 2 から状態 3 への状態遷移が欠落したか、Z 仕様中のイベント E_3 に対応する操作スキーマに欠落があると判定できる。ある状態 S_2 から S_3 へイベント E_3 により状態遷移が発生しないことは S_2, S_3 に対応する集合の間に要素移動がないということなので、イベント E_3 に対応する操作スキーマに以下のような制約式が欠落したことを判定できる。

$$\text{集合間を移動する要素} \notin S_2$$

5 考察

本手法では最初に状態と集合の数を検証し、その後イベントと操作スキーマを検証者が対応付ける。この作業は単純で、誤りの混入する可能性が低いと考えられる。

状態と集合の数を検証することで、集合の包含関係から状態スキーマの中のシステム不変条件に関する誤りを検出できる。操作スキーマの中の制約式に関する誤りがあれば、その操作スキーマに対応するイベントが起こした状態遷移の不一致が発生するので、その不一致である状態遷移から誤った制約式を推論できる。操作スキーマの規定式に関する誤りが発生すれば、その操作スキーマに対応する状態遷移の不一致を検出できるが、本手法では各状態に対応する集合間の要素移動により誤りを推論するので、操作スキーマに制約式が欠落したか、規定式が欠落したかを判定できない。従って規定式に関する誤りを本手法で検出できるが、修正方法を推論できない。

Z 仕様を記述する時に、状態不変条件は明示されない場合がある。これをシステムの暗黙な制約という [2]。ある操作が発生し、その操作により集合間の要素移動が発生する場合、Z 仕様側において、その要素移動の発生する集合の変化という静的な側面に着目するが、本手法では集合間の要素移動を状態遷移としてとらえ、ある状態遷移が発生したら、遷移元と遷移先の状態を必ず存在する。このように、システムの暗黙な制約が存在しても、2 つのモデルの状態数を検証する時に、そのシステムの暗黙な制約を欠落として検出できる。

6 終わりに

本稿では Z 仕様中のデータ型の実体であるエンティティの振舞いに着目し、イベントと操作スキーマの対応付けにより、入力を簡単化した。エンティティの動的な側面を抽出し、この側面を状態遷移規則と比較することで、Z 仕様の誤りと状態遷移規則の誤りの検出でき、各状態間の遷移の有無により欠落した式を推論する手法を提案した。今後は提案手法を実装していく必要がある。

参考文献

[1] 平岡 雅也 織田健. Z による仕様記述と状態遷移規則の比較による誤り検出法. FIT2004 第 3 回情報科学技術フォーラム.

[2] 張 漢明, 荒木啓二朗. 操作仕様記述におけるシステム状態不変条件の抽出. 日本ソフトウェア科学会 FOSE'95.