動作シミュレーションを用いて生成した拡張状態列による仕様理解支援

江上侑希† 高木智彦‡ 古川善吾‡

香川大学大学院工学研究科信頼性情報システム工学専攻†
香川大学工学部信頼性情報システム工学科‡

1. はじめに

ソフトウェア開発において仕様書は試行錯誤を繰り返しながら作成される．そのため，変更される仕様書における振舞いの差異を明らかにすることは仕様を理解するに当たって重要である．また，ソフトウェアの仕様書はオブジェクト指向の普及に伴い，モデルの統一表記法である UML(Unified Modeling Language)によって記述されることが一般的である．そこで，本研究では，UML で記述された仕様書のバージョン間における違いを理解するために，動作シミュレーションにより状態列を組み合わせた拡張状態列を提案する．拡張状態列によって各バージョンにおける仕様書の違いが現れるため，仕様の理解性が向上する．

2. 拡張状態列

状態の実行順序である状態列の生成には動作シミュレーションを用いる．動作シミュレーションとは，UML で記述された仕様に基づいてシステムの振る舞いを逐次実行するとともに，その実行過程をリアルタイムで人間に分かりやすく表示するものである．動作シミュレーションを用いることにより，開発者は任意に状態列を生成できる．

拡張状態列はステートマシン図のバージョン名と状態を保持し，動作シミュレーションにより各バージョンの状態列を組み合わせて作成される．拡張状態列の生成法について述べる．図 1 の V1 のステートマシン図が V2 のステートマシン図に修正されたとする．動作シミュレーションを用いて，V1 のステートマシン図における状態列を生成する．V1 から生成された状態列を記憶しておき，V2 の修正が影響する状態まで V1 の状態列をステップバックする．その後，ステップバックした状態から V2 のステートマシン図における状態列を生成する．これにより，V1 と V2 の拡張状態列が生成される．生成された拡張状態列は図 2 である．拡張状態列の状態には(バージョン名:状態)を記述する．また，拡張状態列は列と定義しているが，図 2 のように分岐しているように見える．列と定義する理由は，拡張状態列は各々のバージョンに分離させると状態列であり，拡張状態列のリーフから辿れば列であるためである．

拡張状態列は列にすることによって，考える範囲が狭くなるので理解性が高まる．しかしながら，各バージョンにおいて複数個所の修正がある場合は，拡張状態列も複数個生成する必要がある．また，拡張状態列は修正が行われる状態からステップバックにより状態系列を戻して修正や分析を行うため，事前に存在している状態列に限定した範囲の中で分析を行うことができる．そのため，拡張状態列は仕様の修正方法を検討する場合にも有効である．

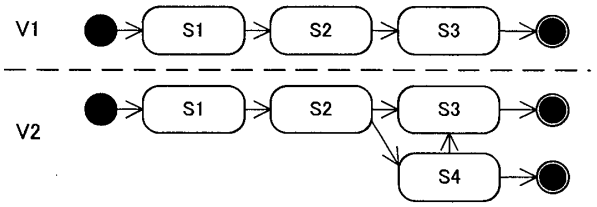


図 1: 各バージョンのステートマシン図

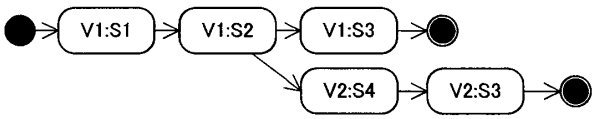
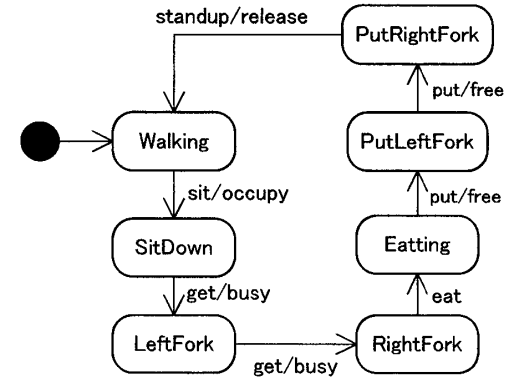
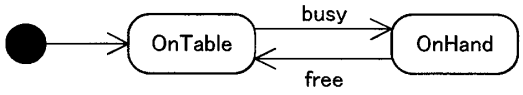


図 2: 拡張状態列

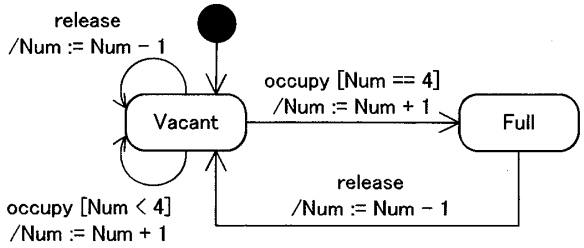
Specifications Understanding Support by the Extended State Sequences Generated with a Behavior Simulation
Yuuki Egami , Tomohiko Takagi , Zengo Furukawa
Faculty of Engineering, Kagawa University



(a)哲学者のステートマシン図



(b)フォークのステートマシン図



(c)座席のステートマシン図

図 3: 哲学者問題のステートマシン図

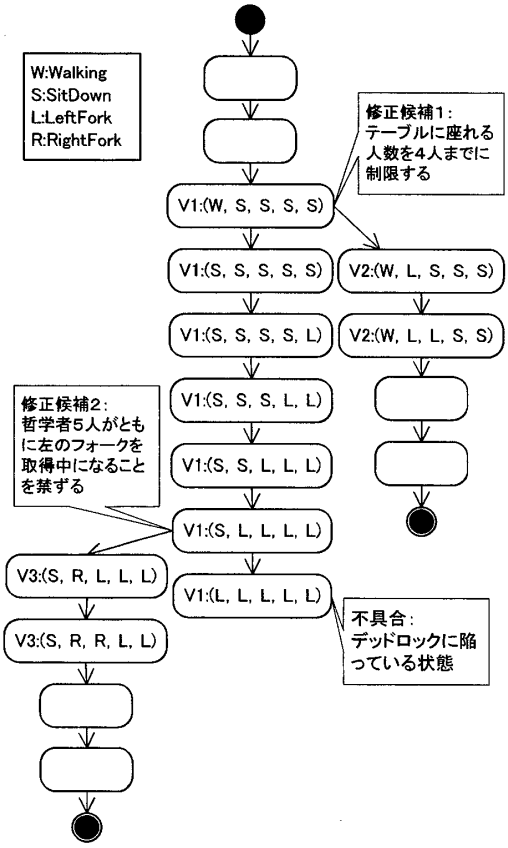


図 4: 哲学者問題の拡張状態列

3. 適用例

適用例として，「食事をする哲学者問題」[1]を例に説明する．哲学者，フォーク，座席のステートマシンはそれぞれ図 3 に示すように定義できる．これを元に作成した，5 人の哲学者が動作する場合のプロダクトマシン（並行動作するステートマシンを 1 つのステートマシンとして合成したもの）が，図 4 の V1 のラベルが付いた部分である．括弧の中は，図 3 の(a)で表される哲学者クラスから作成された 5 人の哲学者オブジェクトの状態の組を表している．V1 は 5 人の哲学者が左のフォーク取得中である V1:(L, L, L, L, L)の状態になり，デッドロックに陥る．そこで，V2 および V3 では以下の修正が行われた．

- V2:哲学者が席に着くのを 4 人に制限する
- V3:5 人の哲学者全員が左のフォーク取得中になってはならないという制約を追加する

以上の内容は図 4 の拡張状態列において明快に示されており，ゆえに本手法が仕様の理解支援に役立つことが期待できる．

4. おわりに

本稿では，動作シミュレーションを用いた仕様化支援のために，ステートマシン図から生成される状態列の組み合わせである拡張状態列について提案した．拡張状態列では仕様であるステートマシン図のバージョンによる違いが明らかになるので，仕様の理解が行いやすくなる．また，拡張状態列は修正が行われる状態から状態系列を戻して修正や分析を行うため，事前に存在している状態列に限定した範囲の中で分析を行うことができる．これにより，仕様の理解や分析が行いやすくなる．また，拡張状態列を保存することにより，変更の履歴が残るので設計のトレースが容易になる．

今後の課題として，拡張状態列はバージョン間における状態の意味が異なる可能性があるので，別途に検証を行う．また，今回検討した仕様化支援機能を実装することである．実装後は今回検討した機能の有効性を定量的に示す予定である．

参考文献

[1] Edsger W. Dijkstra: Co-operating SequentialProcesses, in Programming Languages, ed. F. Genuys, Academic Press, pp. 43-112, 1968.