動的更新が可能なソフトウェア開発手法の研究

阿部 友樹 (410002)

北陸先端科学技術大学院大学 情報科学研究科

2006 年 2 月 9 日

キーワード: 動的更新，状態遷移図，SPIN，LTL.

背景と目的

近年，ユビキタス環境が整いつつある．この環境においてサービスを提供するソフトウェアは永続的に動作することが求められる．しかしこれらのソフトウェアには，停止することが避けられない状況がいくつか存在する．その１つとしてソフトウェアの更新作業があげられる．ソフトウェアの更新はバグの修正や機能の拡張，セキュリティの強化などを行う上で必要とされる作業である．現在の一般的な更新手法はシステムを一時終了する必要がある．この間はサービスを停止しなければならないので，サービスの品質の低下に繋がる．また，あらゆるところに存在するシステムを手作業で更新するとなれば，莫大な人件費がかかる．これらの問題はユビキタス環境を整備する上で大きな問題となりうる．

そこで動的な更新手法が重要視される．動的な更新とは，システムを終了させることなく，安全に新しいシステムへ移行する技術のことである．動的な更新手法を確立するためには，以下の手法を明らかにする必要がある．

1. 移行しても不具合が発生しない正しいポイントの発見手法.
2. 発見したポイントから再開するための現実的な実装手法.

本研究では手法 2 はすでに実現されているものと仮定して，手法 1 の提案を行なった.

アプローチ

手法１を提案する際に，最も困難とされるのが「正しいポイント」の発見である．なぜなら「正しさ」の定義が明確でない上に，システムの広大な状態空間から再開可能なポイントを探し出すのは非常に困難であり，多大なコストがかかるためである．ここで再開可能とは更新において一時的な停止は伴うが，システムを終了させることなく移行することが可能であることを指す.

そこで本研究では多大なコストをかけない現実的な手法を提案するために，状態遷移図が利用可能なレベルまで再開可能なポイントを限定するというアプローチをとった．ポイントを限定することによって，有限な状態空間での再開可能なポイントの探索が可能となる．また探索する状態空間が有限であれば，モデル検査ツール SPIN を使用することができる．本研究では状態遷移図と SPIN を用いて手法 1 の提案に取り組んだ．上記の「正しさ」については，更新時に以下に示す 3 種類の不具合が発生しないことであると定義した．

- システムが停止，または暴走する．
- 不正な値の操作を行なう．
- ユーザに不利益が生じる．

また研究を進める上でいくつか仮定を置いたので，それを以下に記す．

- システムは状態遷移図通りに実装されている．
- 実行中の旧システムからは，現在の実行状態を抽出することができる．また新システムにその実行状態を入力することで，正しいポイントから再開することができる．
- 新旧システムはそれぞれ単独では不具合なく動作するシステムである．

提案した手法

本研究では前節で述べた「正しさ」の判定を段階的に行なう手法を提案した．その手法を提案する際に，状態の制約とサービスという情報を状態遷移図に追加した．状態の制約とは状態遷移図において、ある状態の入場動作を終えて状態内に滞在している時に、常に成り立つ条件の集合である．またサービスとはアクションや活動の一種で，ユーザにとって有益なシステムの動作を指す．この 2 つの情報を用いて行なう段階的な判定手法を以下に示した．

STEP1 状態の制約による候補の絞り込み．

STEP2 システムのデッドロック検査．

STEP3 LTL 式による仕様検査．

STEP1 では不正な値の操作を行う可能性を秘める状態からの再開を予防し，STEP2 では更新時にシステムが停止，または暴走しないかどうかを検査する．そして STEP3 でユーザに不利益が発生しないことを確認することで,「正しさ」の判定を行なう．

評価と展望

提案した段階的な検査手法を全てパスすることで，以下のことが保証される．

- 更新後に不正な値の操作を行なわないこと．
- 更新時，または更新後にデッドロックが発生しないこと．
- 新システムが旧システムの仕様を満足していること．
- 更新時にサービスの欠落，または重複提供を行なわないこと．

この手法における最大の利点は，再開可能なポイントを発見するためのコストを大幅に削減できることである．しかしその反面で判定基準が厳しすぎるために，再開できるはずのポイントを見逃しているという欠点も併せ持っている．今後は利点の恩恵を失わない程度に，判定基準を緩和するルールの制定が求められる．また欠点として，更新前と同等の変数，サービス，外部イベントを持っていなければ，この手法を適用することができないという制限がある．この手法をより有効にするためには，上記の 2 つの欠点を補うことが重要となる．

そしてこの手法を現実的に適用させるためには，仮定で示した状態遷移図通りにシステムを実装する方法や，動的にシステムを移行する現実的な実装方法の提案を行なう必要がある．