論文の内容の要旨

論文題目　動的ウェブアプリケーションを対象とした
モデル検査技術に基づくシステムテスト手法
Model-Checking Based System-Testing Techniques
for Dynamic Web Applications

氏　名　谷田　英生

近年，ウェブブラウザ内で動作するウェブアプリケーションは，動的なアプリケーションの実装を可能とするAjaxなどの技術が普及すると共に，急速にその適用分野を拡大した．また，それと共に実装されるアプリケーションの規模・複雑度も増大し，効率の良く網羅的な検証手法の確立が待ち望まれている．現在，ウェブアプリケーションの開発においてはテストケースを用いた検証が行われることが多いが，テストによる検証手法は網羅的でなく，稀にしか再現しないコーナーケースの不具合を残してしまう可能性がある．そのため，一般的なプログラムの検証を目的として，プログラムを実際に動作させることなく解析して不具合が存在しないかを調べる静的コード解析や，プログラムの挙動を有限状態機械としてモデル化してその正しさを数学的に証明するモデル検査などの検証技術が提案されている．

そのうち，モデル検査は，従来プログラムの挙動を表すモデルを人手で作成する必要があったために，開発の初期段階の上流工程において仕様の正しさの検証に用いられることが多かった．しかし，昨今では，プログラムの実装を元にモデルを自動的に作成して，実装の正しさを自動的に証明するために使用可能とする技術が提案されている．一方，そのような技術は単一計算機上で動作するようなスタンドアロンプログラムを対象としている．そのため，ウェブアプリケーションシステムの一部分を構成するサーバ・クライアント上のプログラムを個別に検証するために適用することは可能であるが，実際にウェブアプリケーションが動作するときと同様にネットワークを介してサーバ・クライアント上のプログラムが協調動作した際の挙動を検証対象とすることは出来ない．

そこで，本論文では，ウェブアプリケーションシステム全体の挙動をモデル検査の対象とすることを可能にする手法を提案する．なお，本論文では，ウェブアプリケーションのさまざまな挙動のうち，画面間遷移に関するものを検証対象とする．本論文で提案する主な手法は，検証対象アプリケーションを特殊なクローラによって自動操作して画面間遷移モデルを抽出する手法，抽出されたモデルを用いてアプリケーションの挙動が期待する性質（プロパティ）に合致するかを検証する手法，クローラによってモデルを抽出する際にアプリケーションの一部機能を動作させるために必要となるデータ入力を自動的に生成し画面間遷移モデルに含まれるアプリケーションの挙動を拡大する手法である．

本論文は，第2章では動的ウェブアプリケーションの実装に一般的に用いられる技術に関する紹介を行う．第3章では動的ウェブアプリケーションの検証に適用可能であると考えられる既存の検証技術を取り上げ，それらの制限に関する議論を行う．

第4章では，ウェブアプリケーションの検証に用いる画面間遷移モデルの定義を行ない，その自動抽出手法を提案して評価する．まず，アプリケーションの画面間遷

移に関する検証を行うために必要な情報に関する議論を行なって，モデルの構造を定義する．画面間遷移モデルは有限状態モデルの一種であり，画面間遷移モデルでは，クライアントのブラウザ内でクリックなどの任意の操作を行った際に，表示内容に対応する動的HTML文書が変化した場合に状態遷移が生じたとみなす．また，操作を行ったのちに表示された動的HTML文書が，過去に表示したものと同一のものであれば，同一状態に到達したと見なす．なお，動的HTML文書は，ブラウザ上で動作しているプログラムによる文書の構造の追加・削除などの書き換えを反映したHTML文書を指す．

このようなモデルを，実際のウェブアプリケーションの操作に一般ユーザが使用するウェブブラウザを制御する特殊なクローラを使って構築する．クローラは指定された探索深さまで，指定された種類（<a/>など）の全てのHTML要素に対して指定された種類（クリックなど）のイベントの発行を自動的に行なって，その結果を観測してモデルを構築する．また，ウェブアプリケーションの使用画面の収集を目的としてクローラを提案した従来研究に比較して，検証対象の挙動への到達に必要なデータ入力などを指定できるguided crawlingという機能を提案する．Guided crawling機能への入力は複数のguidance directiveであり，directiveはモデル抽出中の各状態においてdirectiveが有効化されるかの真偽値を返す手続きと，有効化された際に行うデータ・イベント入力を返す手続きから構成される．Guided crawlingにより，実際の開発現場で検証の対象となる多くのウェブアプリケーションを自動でクロール対象とすることが可能となり，モデル抽出の対象とすることが可能となる．

さらには，モデル抽出やモデル検査を短時間で行うためにモデルを簡略化する手法も提案する．簡略化の内容としては以下の二つを提案する．第一に，自動的に指定された種類の全要素に対してイベント発行する際に，検証対象の機能に関係しないとユーザに指定された一部の要素をイベントの発行対象から除外して探索の幅を狭くする．第二に，各状態が等価であるかどうかを判定する際に，ユーザに指定された動的HTML文書の一部構造を切り落としてから，等価性を判定する．これにより，似た内容を表示している複数の状態を同一状態とみなしてモデルを構築することが可能である．

提案したモデル抽出手法は，実際に幅広く使用されているアプリケーションを含めた複数の例題を用いた評価を行った．Guided crawlingの機能を用いたユーザに指示された入力操作を行わなければモデル抽出が不可能であったようなアプリケーションについても，画面間遷移モデルを抽出することができた．また，モデルの簡略化を行った場合のみモデル抽出の処理が数日程度の処理時間で完了できた例題も存在し，モデルに対する簡略化を行う手法の有効性も示された．使用した例題やパラメータにも依存するが，生成されたモデルには最大で1,580状態と4,039状態遷移が含まれる．モデルの抽出に要した時間は，最大で46時間程度となっている．

第4章の最後では，画面間遷移モデルに抽出されるアプリケーションの挙動に関する議論を行う．そして，モデルを用いた検証手法で対象とすることが可能なアプリケーションが満たすべき性質（プロパティ）に関する議論を行う．

第5章では，第4章の画面間遷移モデル抽出手法によって得られたモデルに基づく検証手法を提案する．まず，開発現場で検証対象とされるような画面間遷移に関する仕様の具体例をいくつか示して，それらの仕様との整合性を検証するために画面間遷移モデルを用いたモデル検査を行うと，より網羅性の高い検証を実現可能であることを説明する．そして，既存のソフトウェア実装のモデル検査を用いた形式的

検証技術との差異点と，その長所・短所に関して議論する．

その後に，調査を行った開発現場でウェブアプリケーションの画面間遷移に要求されるような仕様例をもとに，ウェブアプリケーションの検証に特化したモデル検査手法を提案する．提案する検査手法においては検証対象アプリケーションが満たすべき性質を表現するプロパティ内に用いる原子命題として，現在のウェブアプリケーションの開発現場でサーバ・クライアントが協調して動作した際の挙動をテストする際のアサーションに近い複雑な式を用いることが可能である．具体的には，提案手法は各状態に対応する動的HTML文書を常に同一の真偽値にハッシュする任意の手続きを原子命題として使用することが可能である．そのために，従来のテストによる検証技術から，提案するモデル検査に基づくモデル検査手法への移行は容易であると考えられる．また，テストによる手法に比較して提案手法はより広い実行コンテキストでのアプリケーションの挙動を網羅的に検証することが可能である．

上記のような複雑な式を用いた原子命題を用いるプロパティへのアプリケーションの整合性を検証する手法として，二つのアプローチを提案する．まず最初に，ウェブアプリケーションの画面間遷移モデルに特化した独自の検査器を用いて，検査を行う手法を提案する．モデル検査の際に用いるプロパティとして，時相的に複雑なものが使用される場合は非常に限られているという既存研究の報告にヒントを得て実際のアプリケーションの開発現場で使用されるテストケースを調査した結果，本研究では時相的に3つの種類に分けられるプロパティを検査対象とすれば，テストで検証対象とされているアプリケーションの性質をテストよりも広い実行コンテキストで検証可能であると判断した．よって，独自の検査器を用いた手法においては，この3つの時相的種類に分類されるプロパティを検証対象とする．また，さらに時相的に複雑なプロパティを検証可能とする既存のモデル検査器を用いた検証手法についても提案する．この手法においては，画面間遷移モデルと検証対象のプロパティが与えられた際に，検証対象のプロパティに含まれる原子命題が画面間遷移モデルの各状態においてとる値を評価しながら，既存のモデル検査器に入力可能な形式のモデルを構築する．そのように構築されたモデルを既存のモデル検査器による検証対象として使用する．

提案する検証手法は，モデル抽出の評価実験と同様の例題を用いて評価を行った．計31件のプロパティを用いて検証を行った結果，評価実験に用いた実際に幅広く使用されているアプリケーションを含む例題3件のうち，2件のアプリケーションについて複数の不具合を発見することが出来た．また，抽出されたモデルに基づく各プロパティのモデル検査は，いずれも10秒以内に終了した．そのため，提案する検証手法は，複数の例題を用いた評価実験においても，ウェブアプリケーション画面間遷移に関する不具合発見能力を持つことが示された．

第6章においては，第4章において提案した，検証対象とするアプリケーションの画面間遷移モデルを構築する手法において，データ入力を自動的に生成する手法を提案する．第4章において提案した画面間遷移モデルの構築手法においては，検証対象の挙動への到達に必要なデータ入力を手動で指定して，モデルを構築することを可能とした．このようなアプローチは，アプリケーションの検証対象となる挙動に特定のデータが必要となる場合に非常に有効と考えられる．一方，検証対象の挙動への到達のために空文字列でない何らかのデータを入力すれば良い状況も多くのアプリケーションで存在する．そのため，本研究ではモデル生成を行なっている最中にデータ入力が可能なフォームを伴うページに到達すると，ランダムに空文字列ではない入力データを生成して入力してから，イベントの発行を行なってモデルを抽

出する手法を提案する．提案した手法は，抽出されたモデルに含まれる状態・状態遷移数の他に，モデル抽出の際に実行されたブラウザ上で実行されるJavaScriptプログラムの実行カバレッジに基づく評価も行う．

そして，第7章においては本論文で提案・評価した各手法のまとめを行なって，今後の課題・考えられる将来の研究の方向性について述べる．本論文で提案した各手法は，現在一般的となったインタラクティブなユーザインターフェースを提供する動的ウェブアプリケーションのモデル検査技術に基づくシステムテスト手法を提供する．提案手法のうち，第4章に記したモデル抽出手法は，従来の研究に比較して検証対象の挙動への到達にデータ入力が必要となる場合にも，その挙動のモデルを抽出することが可能である．また，抽出されるモデルは，提案するモデル検査に基づく検証手法に適したものとなっている．第5章に記したモデル検査に基づく検証手法は，現在一般的に開発現場で使用されるテストに基づく検証手法に比較して，より広い実行コンテキストでの検証対象アプリケーションの挙動の妥当性を検証可能である．また，従来のモデル検査に基づく検証技術に比較して，現在開発現場で使用されているアサーションに近い形式でのプロパティの記述を可能とする．第6章に記した，モデル生成時のデータ入力を自動的に生成する手法は，より少ない工数で検証対象のアプリケーションの挙動をモデルに抽出することを可能とする．本論文で提案する各手法を使用すると，現在，開発現場で多用されているテストに基づく手法に比較して，より幅広い実行コンテキストで動的ウェブアプリケーションの挙動の妥当性を時相的論理により用いて検証することが可能で，不具合を残したままアプリケーションを現場で運用してしまう可能性を低減することが可能である．

今後の研究の方向性としては，より短時間で効率的に画面間遷移モデルを生成する手法や，ウェブブラウザで表示されている以外のサーバ・クライアントで実行されているプログラムの変数やデータベースに格納されている情報をモデルに含めて，検証可能とするアプリケーションが満たすべき性質の種類を拡張することが考えられる．