平成 24 年度

学士学位論文

Web アプリケーションを対象とするモデル検査法の効率化に関する研究

On improvement of the efficiency of model checking for Web applications

1130309　浦部 未来

指導教員　高田 喜朗

平成 25 年 3 月 1 日

高知工科大学 情報学群

要 旨

Web アプリケーションを対象とする
モデル検査法の効率化に関する研究

浦部 未来

今日，計算機アプリケーションは一般に広く普及している．このアプリケーションの開発において，仕様通り動くことを保証することは重要である．その正しさの保証を行うものの 1 つにモデル検査がある．

モデル検査は，その検査の効率化が必要である．Zhang らの研究ではハードウェアに対するモデル検査の効率化手法が提案されている．Web アプリケーションについて，この手法が適用可能であるか調べることを本研究の目的とする．その第一段階として本研究では，Web アプリケーションの画面遷移仕様をペトリネットでモデル化する方法を検討する．

結果，例題に対しペトリネットでモデル化することができた．今後，Zhang らの手法が適用出来るか吟味する必要がある．

キーワード　モデル検査，ペトリネット，Web アプリケーション

Abstract

On improvement of the efficiency of model checking for Web applications

URABE Miki

Recently, computer applications are widely spread in general. In the development of an application, to ensure that the application works as specified is important. Model checking is one of the techniques to guarantee the correctness of computer applications. Efficient techniques for the hardware model checking has been proposed by Zhang et al. The purpose of this research is to investigate whether this technique can be applied for Web applications. As a first step, in this paper we consider how to model in a Petri net the screen transition specification of a Web application. As a result, we could model a sample specification in a Petri net.

key words model checking, petri net, Web application

目次

第 1 章

はじめに

現在，計算機アプリケーションは一般的利用者の身の周りに広く普及している．アプリケーションの開発において，アプリケーションが仕様通りに動くことは重要であり，これを保証しなければならない．その仕様通りに動くことを保証するための方法のひとつとしてモデル検査がある．

モデル検査とは，形式手法に基づく自動検証法のひとつである．検査対象をモデル化し，状態空間を網羅的に探索することで不具合の発見や期待する状態への到達性などを確認する．モデル検査では，検査対象のモデル化をどのように行うかによって検査の計算量や精度が変化する．モデルが大きすぎると状態爆発を引き起こし調べられなくなるが，抽象化しすぎると正しい検査結果を導き出せなくなる．また，単純な状態空間の大きさだけでなく，半順序簡約などの効率化法が有効に働くかどうかによっても計算量が大きく変化する．

Zhang ら [1] が，ハードウェアに対するモデル検査において半順序簡約の効果を高める手法を提案している．多くのモデル検査ツールがモデル化に有限オートマトンを使用しているが，この手法ではモデル化にペトリネットを使用している．

本研究では，Zhang らの効率化法が Web アプリケーションを対象としたモデル検査に関して適用出来るかどうか調べることを目的とする．その第一段階として本研究では，Web アプリケーションのためのペトリネットによるモデル化法を検討する．以降、本論文では、2 章では Web アプリケーションのモデル検査の対象について説明する．3 章では，ペトリネットについての解説をし，4 章では，ペトリネットを利用した Web アプリケーションのモデル化について説明する．

第 2 章

Web アプリケーションのモデル検査

2.1 概要

Web アプリケーションの設計では，まず必要な処理データ内容や画面遷移仕様などを最も基本的な要求仕様として作成する．その後，仕様をもとに入出力データの処理，ユーザ認証，イベント管理といった詳細なプログラム処理の流れ (フローチャート) を基本的な設計仕様として作成する．この要求仕様とフローチャートの間での食い違いがあると，要求通りのシステムが実現出来なくなる．そのため，フローチャートが要求仕様を満たさなければならない．フローチャートが要求仕様を満たしているかどうか調べるためにモデル検査が有効であると考えられる．

本研究では [2] を参考に，Web アプリケーションの設計の中でも画面遷移仕様に着目し，画面つきフローチャートをペトリネットでモデル化することを検討する．

2.2 画面つきフローチャート

本研究での画面つきフローチャートについて説明する．図 2.1 は画面つきフローチャートの例である．フローチャートが Web アプリケーションに関する各処理を表している．その各処理に対し画面を割り当てることで画面つきフローチャートを形成している．この例では，各処理の右上に書かれている (a)，(b)，(c) が画面を表している．これをモデル検査が

2.2　画面つきフローチャート

行えるようモデル化する.

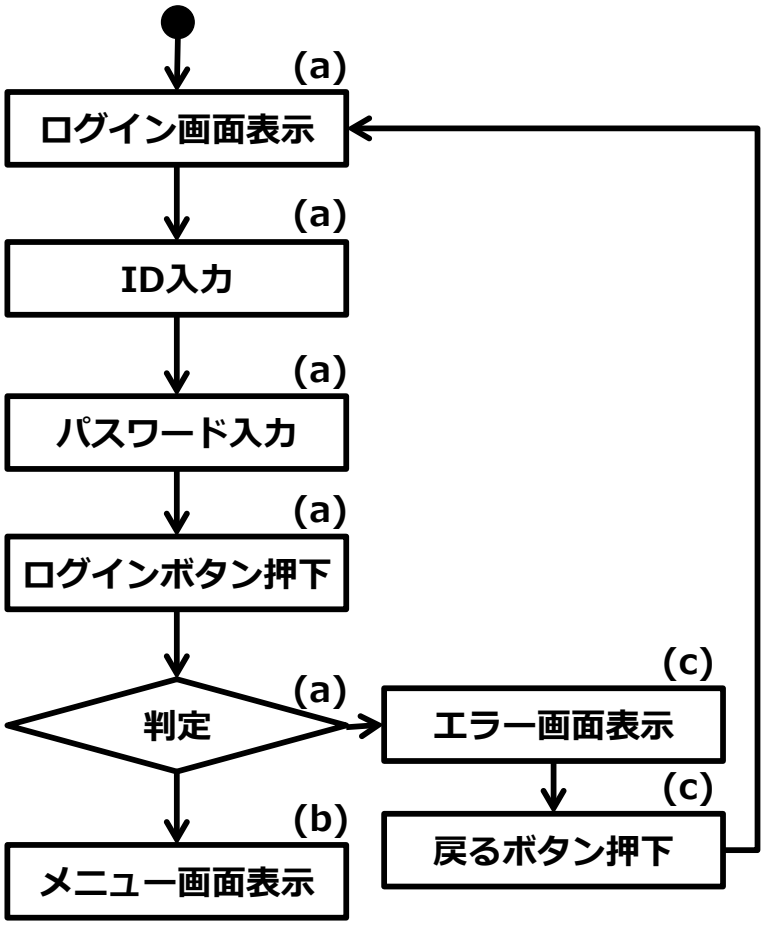

図 2.1　画面つきフローチャート

第 3 章

ペトリネット

3.1 概要

ペトリネットは，1962 年ドイツの Carl Adam Petri によって発表された．チューリング機械と有限オートマトンの中間のモデル化能力を持つ．多くのシステムに適用可能であり，並行的・非同期的・分散的・並列的・非決定的・確率的な動作を特徴とするシステムに関して記述力がある．

3.2 定義

ここでは，[4] を元にペトリネットの定義について説明する．

ペトリネットは，プレース (place) とトランジション (transition) という二種類のノード (node) からなる有向二部グラフである．トランジションは棒や箱で表されるノードであり，事象を表している．プレースは円で表されるノードであり，条件を表している．これらを結ぶアーク (arc) は条件と事象の間の関係を表す．プレース，トランジション，アークによってシステムの構造を表現する．アークには正整数の重みが付けられる場合があり，重みはアークの本数で表すか，アークに重みを併記して表す．重みが付けられないアークは重みが１であるとみなす．プレースの上には，非負整数個のトークン (token) が置かれる．トークンは点で表され，条件の成立を表す．このトークンの配置をマーキング (marking) と呼び，システムの状態を表す．

このペトリネットを形式的に表すと，$N = (P, T, F, W, m_0)$ となる．

$$
\begin{cases}
P = \{p_1, p_2, \ldots, p_{|P|}\} & \text{プレースの有限集合 } (|P|\text{:プレースの数}) \\
T = \{t_1, t_2, \ldots, t_{|T|}\} & \text{トランジションの有限集合 } (|T|\text{:トランジションの数}) \\
F \subseteq (P \times T) \cup (T \times P) & \text{アークの集合} \\
W : F \longmapsto \{1, 2, \ldots\} & \text{アークの重み} \\
m_0 : P \longmapsto \{0, 1, 2, \ldots\} & \text{初期マーキング}
\end{cases}
$$

$|P|$ 個のプレースからなるペトリネット全体のマーキングはベクトルの形で，$m_0 = [m_0(p_1)m_0(p_2)\cdots m_0(p_{|P|})]^T$ のように表すこともできる．トランジション t に向かうアークを t の入力アーク，t の入力アークが出て行くもとのプレースを t の入力プレースという．t から出て行くアークを出力アーク，t の出力アークが入って行く先のプレースを t の出力プレースという．プレース p の入出力アーク，入出力トランジションもまた同様に定義する．

図 3.1 のペトリネットは,

$P = \{p_1, p_2, p_3, p_4\}$

$T = \{t_1, t_2, t_3\}$

$F = \{(p_1, t_1), (p_2, t_1), (p_3, t_2), (p_4, t_2), (p_4, t_3), (t_1, p_3), (t_1, p_4), (t_2, p_2), (t_3, p_1), (t_3, p_4)\}$

$W((p_1, t_1)) = 3, W((p_3, t_2)) = 2, W((p_2, t_1)) = \cdots = W((t_3, p_4)) = 1$

$m_0(p_1) = 3, m_0(p_2) = 2, m_0(p_3) = 1, m_0(p_4) = 1$

と表される．

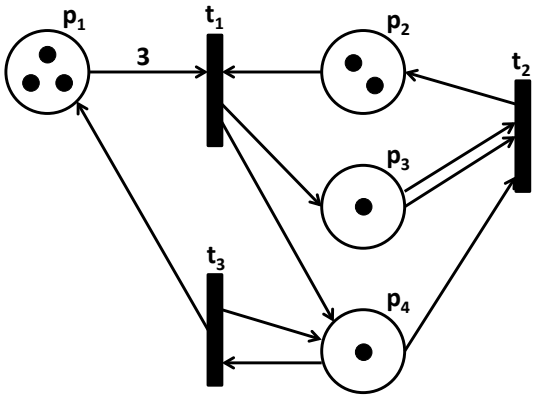

図 3.1 ペトリネットの例

マーキングをベクトルの形で表すと，$m_0 = [3211]^T$ となる．t_1 の入力プレースは

$\{p_1, p_2\}$, 出力プレースは $\{p_3, p_4\}$ である．p_1 の入力トランジションは $\{t_3\}$ ，出力トランジションは $\{t_1\}$ である．

3.3　発火可能性

ペトリネットのマーキングはトランジションの発火 (firing) によって遷移する．トランジション t がマーキング m で発火可能 (fireable, enabled) であるとは，t のすべての入力プレースが入力アークの重み以上の個数のトークンを持つことであり，$m[t\rangle$ で表す．発火可能なトランジションは発火してもしなくてもよい．発火可能なトランジション t が発火すると，各入力プレースから入力アークの重みの数だけのトークンを取り去り，各出力プレースに出力アークの重みの数だけのトークンを与える．この結果のマーキングを m' とすれば，$m[t\rangle m'$ と表される．

前記のトランジション発火の規則に対しては，各プレースにいくらでも多くのトークンを置くことが可能であると仮定している．このようなペトリネットを無限容量ネット (infinite capacity net) と呼ぶ．しかし，多くの自然システムのモデル化では各プレースが保持できるトークン数の上限を考慮するのが普通である．このようなペトリネットを有限容量ネット (finite capacity net) と呼ぶ．有限容量ネットでは，各プレースが保持できるトークン数に容量が割り当てられており，容量とはプレースが任意の時点で保持できるトークンの最大数である．プレースを p とすると容量は $K(p)$ と表す．有限容量ネットでは，発火可能なトランジション t に対し，t の各出力プレース内のトークンが，t の発火後，各プレースの容量 $K(p)$ を超えてはならないという制約がつく．

例

図 3.1 のペトリネットで，t_1 と t_3 は発火可能である．t_2 は p_3 に入力アークの重み 2 以上のトークンがないので，発火可能ではない．t_1 が発火した結果のマーキングを m_1 とすれば，$m_1 = [0122]^T$ である．

第 4 章

ペトリネットによる Web アプリケーションのモデル化とモデル検査

4.1　画面つきフローチャートのモデル化

ここでは，画面つきフローチャートのペトリネットでのモデル化について説明する．

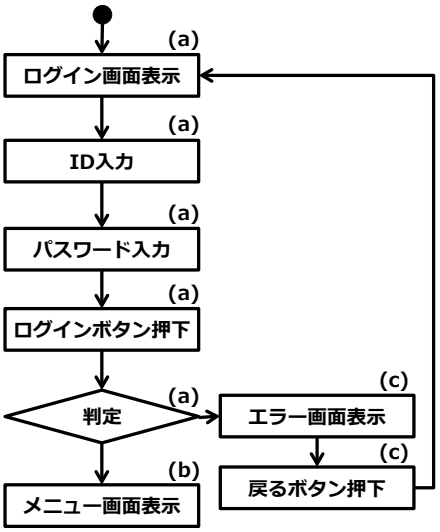

図 4.1　画面つきフローチャート (再掲)

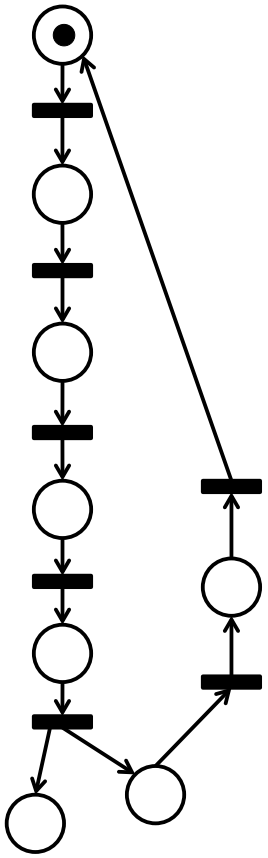

図 4.2　例題のペトリネット

図 4.1 (図 2.1 の再掲) は今回の例題の画面つきフローチャートである．これをペトリネットにモデル化すると図 4.2 となる．各処理をプレースで表している．画面の情報については検査時に付加する．

4.2　モデル検査器 SPIN の入力言語への変換

SPIN は，G.J.Holzmann 博士が中心となって開発，公開しているモデル検査ツールである．SPIN では，検査対象のシステムを Promela (Process Meta Language) と呼ばれる記述言語で記述する．記述された検査対象は有限オートマトンとして解釈され，そのモデルを網羅的に探索することで検査を行う．また，効率を上げるために半順序簡約を使用している．

Promela は，元来通信プロトコルの検証のために開発された．検査対象のシステムは非同期並行動作を行うプロセスの集まりとして記述される．文法は C 言語に似ており，各プロセスの振る舞いを手続き的に記述する．

SPIN でペトリネットモデルの検査を行うためには，ペトリネットを Promela で記述する必要がある．ペトリネットから Promela への変換規則に関しては，[3] を参考とした．

まず，プレースはトークンの情報，すなわちシステムの状態を表すため，通信チャンネルの配列とする．トークンはチャンネルの中のメッセージとして表す．トランジションはシステムの状態を変換するものである．そのためそれぞれプロセスとして表現する．

4.3　考察

今回，画面つきフローチャートについて，例題をモデル化した．このモデル化からモデル検査を行おうとすると各処理分のプレースとそれのほぼ同量のトランジションがあり，ただそのまま検査にかけると効率が悪いと考えられる．この例題では示せていないが，並列に動くような処理，トークンを複数使用することで記述する仕様に対してはペトリネットでの記述のほうが容易であると考えられる．これらに対し，Zhang らの手法が適用出来るのならば，画面仕様のモデル検査のペトリネットでのモデル化は有効であるといえる．

第 5 章

まとめ

本研究では，Web アプリケーションを対象としたモデル検査について，Zhang らの効率化法が適用出来るか検討するためにペトリネットによるモデル化法を検討した．ペトリネットでのモデル化，SPIN によるモデル検査が行えることを確認した．今後，Zhang らの効率化法が使用できるのか吟味する必要がある．

謝辞

本研究を行うにあたり，様々な面でご指導を頂きました本学情報学群高田喜朗准教授に深く感謝いたします．高田喜朗准教授の支えなしではこの研究を提出まで進めることは出来ませんでした．ありがとうございました．

また，副査を引き受けてくださった横山和俊教授，松崎公紀准教授に深謝いたします．

参考文献

[1] Y. Zhang, E. Rodriguez, H. Zheng, C. Myers, “An Improvement in Partial Order Reduction Using Behavioral Analysis”, IEEE VLSI, pp.100–107, 2012.

[2] 崔, 河本, 渡邊, “画面遷移仕様のモデル検査”, コンピュータソフトウェア, Vol. 22, No. 3, pp.146–153, 2005.

[3] 孫, 塚原, 飯島, “オブジェクト指向ペトリネットによるワークフローのモデル化と分析”, 情報システム学会, 第 5 回全国大会・研究発表会, 2009.

[4] 椎塚, “実例ペトリネット その基礎からコンピュータツールまで”, コロナ社, 1992.