社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

確率的モデル検査ツールを用いた実時間ネットワークシステムの検証手法の提案およびネットワークシミュレータ NS-2 との比較

伊藤　明彦†　　長岡　武志†　　岡野　浩三†　　楠本　真二†

† 大阪大学大学院情報科学研究科
〒 565–0871 大阪府吹田市山田丘 1–5
E-mail: †{a-ito,t-nagaok,okano,kusumoto}@ist.osaka-u.ac.jp

あらまし　実時間ネットワークシステムにおいてはネットワークで要求される QoS を達成することが重要とされる．そこで，確率的振る舞いを持つシステムに対するモデル検査ツール PRISM に着目し，これを用いた実時間ネットワークシステムの詳細なモデル化方法，さらにそれらの結果を利用した形式的検証手法について述べる．また本稿では，提案手法の有用性を明らかにするために例題に適用し，ネットワークシミュレータ NS-2 によってモデル化に対するシミュレーション結果の比較，考察についても述べる．
キーワード　実時間ネットワークシステム，モデル検査，PRISM，シミュレーション，形式的検証

Verification for the Real-time Network Systems with the Probabilistic Model Checker and its Comparision with the Network Simulator NS-2

Akihiko ITO†, Takeshi NAGAOKA†, Kozo OKANO†, and Shinji KUSUMOTO†

† Graduate School of Information Science and Technology, Osaka University
Yamada-oka 1–5, Suita City, Osaka, 565–0871 Japan
E-mail: †{a-ito,t-nagaok,okano,kusumoto}@ist.osaka-u.ac.jp

Abstract　For distributed systems, managing QoS parameters is a focal point of their quality. In this paper, focusing on PRISM, a model checking tool for probabilistic real-time systems, we present a new method to analysis real-time network systems detailed, and to implement formal verification with the use of this result. The paper also describes an application example and evaluation of the method with network simulator NS-2.
Key words　Real-time Network System, Model Checking, PRISM, Simulation, Formal Verification

1. ま え が き

ストリーミングサービスの動画像データ配信に代表されるような，リアルタイム系のシステムは QoS 制御のような時間に基づいた配信が必要不可欠と言える．このようなシステムの開発者はシミュレータ [2] や数学的解析 [3] により，システムのパフォーマンスをあらかじめ予測しなければならない．シミュレータを利用した場合，システムの性能をある程度予測することが可能であるが，定性的解析とは言えず数学的な保証にはならない．また，一般に多くの計算リソースを必要とする．一方，数学解析は数学的，論理的に正しく，一般に計算時間を必要としないが，解析対象が限られるのでシステム全体の現実的な定性的解析は困難である．したがって，本手法ではモデル検査を利用することで，システム全体の性能解析を目指す．

確率モデル検査ツール PRISM [1] は時間要素と確率要素をともに取り入れたモデルを対象にしており，実時間ネットワークシステムのようなシステムのモデル化に極めて有用と判断できる．そこで本稿では，PRISM を用いて，インターネット環境における，ルータ間のバッファ溢れやバッファ蓄積による転送遅延を考慮したネットワークシステムの性能調査を行う．

インターネット環境に用いられる代表的なプロトコルである TCP は，正確性に重点を置く代わりに，ある程度の遅延を許容するためにリアルタイム系のシステムでは利用されることがない．リアルタイム系のシステムでは RTSP (Real Time Streaming Protocol) が用いられる．RTSP はインターネットやイントラネットなどの TCP/IP ネットワーク上で，音声や動画などをストリーミング配信するためのプロトコルである．RTSP の輻輳制御機構として，近年では TFRC (TCP-Friendly Rate Control) [4] が注目されてきている．

本稿では輻輳制御機構 TFRC を対象とし，その振る舞いを PRISM を用いてモデル化する．PRISM を用いた検証例題については文献 [5] などプロトコルレベルでの検証はあったが，システムのスループットの解析に踏みこんだ例題はない．本稿はこのレベルのケーススタディを与える．さらに，本研究の先行研究 [9] では固定値として与えられていたパケット損失率をルータのバッファレベルまで詳細にモデル化することで，常に変動するネットワークの性質について厳密に調べる．さらに，その詳細設計モデルより得られたパケット損失率や転送遅延時間などのデータを用いて，送信スループットを検証する簡単化モデルを構築する．また，本稿では PRISM によって得られた結果とネットワークシミュレータ NS-2 を用いてシミュレーションを行った結果を比較し，構築されたモデルの正当性について評価する．

以降，2. ではモデル検査に利用するツール PRISM，さらに実験結果の比較対象として用いるネットワークシミュレータ NS-2 や，ネットワークシステムに関連する実時間ネットワークシステムのプロトコルに関して説明や考察を行う．3. では例題の説明を行い，4. では NS-2，PRISM 上でどのようにモデル化を行うかについて述べる．5. において 4. で述べたモデル化方法について実際に例題に適用し，評価を行う．また，6. において手法の一般化に関する考察を行う．最後に 7. でまとめる．

2. 準　　備

2.1 確率的モデル検査ツール PRISM

本節では，本手法に用いられる確率的モデル検査ツール PRISM [1] について説明する．

モデル検査とは，時相論理式で表現された調べたい性質を，状態遷移系が満たすかどうかを判定するものである．PRISM は状態遷移系として主にマルコフ連鎖を対象とし，Discrete-time Markov chains (DTMC)，Continuous-time Markov chains (CTMC)，そして Markov decision processes (MDP) の 3 種類のマルコフ連鎖を与えることができる．

また PRISM は時相論理式として DTMC と MDP に対応した Probabilistic Computation Tree Logic [6] (PCTL) と CTMC に対応した Continuous Stockastic Logic [7] (CSL) の 2 つを入力として与えることができる．

一般に PRISM は与えられたモデルと論理式に対して，その式が真となる確率を返す．他にいくつかの不確定変数について論理式を満たす値を返すこともできる．

2.2 ネットワークシミュレータ NS-2

本稿におけるシミュレーション実験の比較対象として使用する The Network Simulator (NS) は，ネットワークの研究を対象に開発されたネットワークシミュレータであり，大規模で複雑なネットワーク上で発生するトラフィックの解析といった，ネットワークの動作検証などに幅広く用いられている [10]．NS の後継バージョンである NS-2 の使用言語は，2 種である．ネットワークプロトコルの動作など，コアの部分は C++，ユーザがシミュレーションとしてプロトコルの設定やリンクの設定などシナリオを構成するためのフロントエンドとして，OTcl (Object-oriented Tcl) が使用されている．

ただし，ユーザ独自のプロトコルなど NS-2 で未実装の機能に対しては，C++言語で記述された NS-2 のエンジン部分をユーザ自身で変更していく必要がある．

2.3 実時間ネットワークシステムのプロトコル

2.3.1 RTSP

RTSP はユーザデータの発信元と宛先のエンドノード (端末やコンテンツサーバ) 間の代表的プロトコルの一つである．RTSP は SETUP，PLAY，RECORD，PAUSE，TEARDOWN の 5 つの状態を持っており，エンドノードとのパケット交換により，これらを実現する．一方，実際のストリームデータの転送は RTP (Real-time Transport Protocol) が行う．RTSP は RTCP (RTP Control Protocol) のレポートメッセージを受け取り，TFRC のレート制御方式を用いて，RTP のスループットを定める．

2.3.2 輻輳処理機構 TFRC

リアルタイム系輻輳制御である TFRC は，RTP と TCP との公平性を実現することを目指したレート制御方式である．同じネットワーク環境下 (パス，遅延，ロス率) において TCP フローに対して同じ平均帯域を得られるようなレート制御を行うことで，既存 TCP フローに大きな悪影響を与えることなく通信を行うことができ，ネットワークの系全体の利用率を向上することができる．TFRC は RTCP のレポートメッセージを用いてレート制御を行う．レポートには，受信された RTP パケットのシーケンス番号を調べることでわかるパケット損失量や，タイムスタンプと受信時刻から見積もられたジッタ (転送遅延のゆらぎ) などの情報が含まれている．RFC3448 (Request For Comment) には次のスループット推定式が記載されている．

$$X = \frac{s}{R\sqrt{2*b*p/3}+(t\_RTO*((3*\sqrt{3*b*p/8})*p*(1+32*p^2)))}$$

X の単位は [byte/seconds] である．TFRC は X の値が Bandwidth など出しえる最大のスループットよりも小さい場合，この X をスループットとして設定するように RTSP に指示を出す．R[seconds] はラウンドトリップタイム，p[%] はパケット損失率，s[byte] はパケットサイズ，b[number of times] は TCP 肯定応答回数であり，t_RTO[seconds] は TCP のタイムアウト値を示す．

3. 実時間ネットワークシステムの例題

本章では，実時間ネットワークシステムの例を示すと共に，そのネットワークシステムの性質について述べる．

実時間ネットワークシステムの例として，動画像データ配信システム [8] を用いる．この動画像データ配信システムを，図 1 に示す．図 1 では，中央左のルータが FTP サーバ

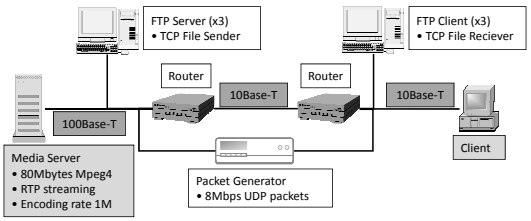

図 1　実時間ネットワークシステムの例題
Fig. 1　Real-time Network System

3 台，動画像サーバ，パケットジェネレータに接続されている．中央右のルータは FTP のクライアント 3 台と，動画像サーバからデータを取得するクライアントに接続されている．各ノードがルータを介して通信を行うことで，擬似的なインターネット環境を構築している．動画像サーバは約 80MBytes の動画データを 1Mbps の送信レートでクライアントに送信する．その際，動画像サーバの RTSP は TFRC によるレート制御を行う．その輻輳制御の影響について調べるために，時刻 100 秒より FTP サーバ，クライアント間で TCP セッションによる通信を開始する．パケットジェネレータは常に 8Mbps の UDP トラフィックを送出する．

4. PRISM と NS-2 のモデル化

本章では，図 1 の例題に対して本稿で構築した PRISM, NS-2 それぞれのモデルについて述べる．

4.1 PRISM でのモデル化

本節では，実時間ネットワークシステムを PRISM でモデル化を行う方法について説明を述べる．

まず実時間ネットワークシステムのプロトコルやバッファ制御の特性を調べ，詳細にモデル化する．そこで作られた DTMC の詳細設計モデルに対し，確率的モデル検査ツール PRISM に入力し，シミュレーションによってシステム上のスループットや，パケット損失などを調べる．さらに，シミュレーションによって得られたパケット損失やラウンドトリップタイムの分布を用いて簡単化モデルを構築し，モデルの検証を実行することで，メディアサーバの送信スループットの定性を調査する．

4.1.1 詳細設計モデルの構成

詳細設計モデルでは，パケット単位の転送のモデルを構築する．そのためには，ルータによるバッファ制御や，送られるパケット数を決定する通信プロトコルの輻輳制御の挙動だけでなく，パケット損失率や遅延時間の計算について考慮した上でモデル化を行う．

4.1.2 時間軸の管理

詳細設計モデルでは，実時間ネットワーク環境を想定して構築している．具体的には，整数変数を用意し，離散的な時間軸に沿って各ノードがパケットを送受信するようにモデル化を行う．

時間経過はパケット送信等のイベントに基づいて行う．まず各モジュールがイベント発生時刻を登録しておく．イベントが登録されると，現在の時刻から最も直近のイベント発生時刻まで時間を経過させる．経過後，登録されたイベントが発生し，イベントを発生させたモジュールは次のイベント発生時刻を再び登録する．時間軸管理のためのモジュールは，変数を 2 個用い，10 行で記述している．

4.1.3 ルータによるバッファ制御

ルータによるバッファ制御については，受信したパケットを蓄積するキューとしてモデル化を行うため，現在のキュー長を整数変数によって管理することによって，キューに対する操作であるエンキュー，デキューの処理を実行する．

エンキューでは，各ノードが送出するパケット数の累計を管理する変数によってパケット損失かどうかを判断する．デキューは，時間軸の管理によって時刻が進められた際のイベント処理と同時に行う．つまり，進められる時間とリンクの転送速度からその時間内に送出可能なパケット数を計算し，キュー内のパケット数を減らす．バッファ制御に関するモジュールは，変数を 9 個用いて，約 100 行で記述している．

4.1.4 通信プロトコルの輻輳制御

ネットワーク上のパケット量は，各通信プロトコルにより決定される．詳細設計モデルでは，リアルタイム分散システムに用いられる各プロトコルの振る舞いについて考慮した設計になっている．以下今回用いる TCP と RTP プロトコルの輻輳制御を考慮したモデル化について説明する．

TCP セッションの振る舞いは，TCP のスロースタートフェーズ，輻輳回避フェーズの振る舞いについてモデル化している．具体的にはスロースタートフェーズの閾値を管理する変数を用意し，スロースタートフェーズの閾値を超えるまで，送信パケット数を倍々に増やしていく．閾値を超えた場合，輻輳回避フェーズとして送信パケット数を 1 ずつ増やしていく．パケット損失の発生時には，スロースタートフェーズの閾値を送信パケット数の半分に，ウインドウサイズを 1 に設定し，4 ラウンドトリップタイム経過後に送信イベントを行う．TCP プロトコルに関するモジュールは，一つの TCP のノード当たり，変数を 6 個用いて約 30 行で記述している．

モデル上の RTP プロトコルでは，ラウンドトリップタイムとパケット損失率をバッファ制御部によって計算し，スループット推定式によって送信レートを算出する．しかし，算出された送信レートが高い場合，モデルの複雑化を避けるために送信レートに合致するように一度に複数のパケットを送信するようにモデル化している．RTP プロトコルに関するモジュールは，変数を 6 個用いて約 40 行で記述している．

4.1.5 ラウンドトリップタイムとパケット損失率の計算

ラウンドトリップタイムは，パケットがサーバからクライアントまでを一往復するのにかかる時間である．モデル上では，リンクを通過するのに必要な物理的な伝播遅延と，ルータによるノード内遅延から求めている．ノード内遅延は，キュー内滞留パケット数と，デキューによる送出速度

から計算する．値の急な変動を防ぐため，TFRC の仕様 [4] に従い，その時点までのラウンドトリップ値と新しく計測されたラウンドトリップ値に対し，重み付けをして加算することで，値を更新する．

パケット損失率は，パケット損失間隔の逆数として求められる [4]．ただし，値の急な変動を防ぐため，パケット損失間隔の履歴を用いて計算する．その中で最近の履歴ほど大きい重みを付随させて加算し，値の更新を行う．モデル上では，各履歴を残しておくため，履歴数分の整数変数を定義している．

ラウンドトリップタイムとパケット損失率の計算に関するモジュールは，変数を 25 個用いて約 40 行で記述している．

4.1.6 簡単化モデルの構成

詳細設計モデルでシミュレーションを行ってスループットを計測することは可能だが，検証を行うことが困難である．その理由は，詳細設計モデルでは，リアル時間単位，パケット単位の解析を行っているのでモデルが複雑になりすぎているためである．本稿ではメディアサーバにおける送信スループットを検証するため，詳細設計モデルのシミュレーション結果を元に簡単化モデルを構成する．

あらかじめ詳細設計モデルより，ラウンドトリップタイムとパケット損失率を計測しておく．次に，輻輳制御時の二つのパラメータに対する離散的な確率密度分布を作成する．そして，各区間に対する確率密度を，各区間中央値に対する生起確率として近似する．次に，モデル上でその生起確率によってラウンドトリップタイム，パケット損失率の値を決定し，2. で述べたスループット推定式にパラメータとして渡す．

簡単化モデルはモジュール数 1，70 行程度で記述している．

4.2 NS-2 におけるモデル化

NS-2 で今回のシミュレーションを行うためのネットワークシナリオの構成としては，NS-2 も PRISM と同様に図 1 の構成に従った．

図 1 の構成における 100Base-T や 10Base-T によって各ノードのリンク設定を行い，FTP サーバと FTP クライアント，パケットジェネレータに対応するノード上の通信プロトコルは，NS-2 エンジンにおいてであらかじめ用意されている TCP，UDP のプロトコルを用いている．

本稿の例題として，NS-2 のモデルでは，輻輳制御機能を持った RTP プロトコルで動作を行うメディアサーバ，そしてその RTP パケットを受信し，適切なフィードバックパケットを送信する RTP の受信プロトコルを設定する必要がある．これらのプロトコルについては，未登録のモジュールであるため，追加的に実装しなければならない．

そこで本稿では，NS-2 の C++で記述されているエンジン部に対し，RTP パケットの送受信プロトコルを TRFC3448 [4] で定められている仕様に従うように，実装を行った．追加，変更ファイル数は 8 ファイル，追加，変更メソッド数は 12，追加，変更を加えたファイルに記述した行数は合計で約 500 行である．

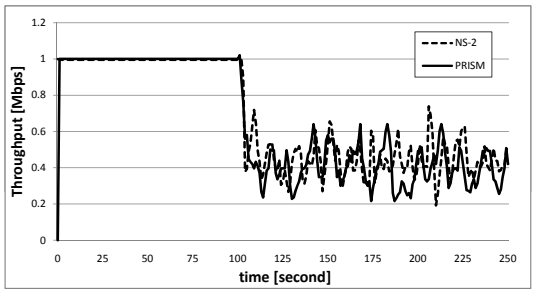

図 2　シミュレーション結果:送信スループット
Fig. 2　A Simulation Result : Send Throughputs

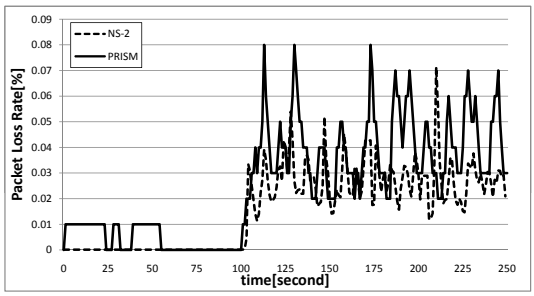

図 3　シミュレーション結果:パケット損失率
Fig. 3　A Simulation Result : Packet Loss Rates

5. 例題適用

考案した検証手法に対して例題を適用した．なお，実験環境は OS が Windows Vista 64 ビット版，CPU が Core2Duo E6550 2.33GHz，およびメモリが 2.00GB である．

5.1 実験方法

PRISM モデル，NS-2 のモデル共に実験シナリオではパケットサイズを 500 (Bytes)，TCP の肯定応答回数を 1，TCP のタイムアウト値を 4 ラウンドトリップタイムと設定している．また，ルータのキューサイズを 32KB，64KB，128KB とした 3 通りのシナリオについて実験を行った．

これらのパラメータを用いて，NS-2 と詳細設計モデルを用いてそれぞれシミュレーションによって動画像サーバの送信スループット，パケット損失率について計測する．また，詳細設計モデルのシミュレーションによって得られたラウンドトリップタイムとパケット損失率の確率分布を元に簡単化モデルを構築し，メディアサーバの送信スループットを検証する．

5.2 シミュレーション結果

ここでは，詳細設計モデルのシミュレーション結果と，NS-2 のシミュレーション結果を比較する．ルータのキューサイズを 64KB とした場合の送信スループット計測値を図 2 に，パケット損失率計測の結果を図 3 にそれぞれ示す．

図 2，3 が示すように，ルータのキューサイズが 64KB である場合に送信スループット，パケット損失率共にネットワークシミュレータの計測値に近い値を導出することがで

表 1　シミュレーション結果比較

キューサイズ	32KB		64KB		128KB	
環境	NS-2	PRISM	NS-2	PRISM	NS-2	PRISM
最大値	607	588	738	640	995	980
最小値	203	48	193	216	33	144
平均値	242	372	443	401	473	367
分散	11	13	10	11	12	12

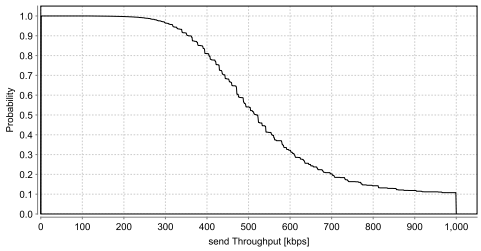

図 4　検証結果:送信スループット値の生起確率

Fig. 4　A Verification Result : Occurrence Probability of Throughputs

きた．次に，キューサイズが 64KB の場合に加え，32KB，128KB の場合のシミュレーションにおける送信スループット値の最大，最小，平均と分散値を表 1 に示す．

全体として最大値は近い値を算出しているが，複雑なネットワークであるため，32KB や 128KB のキューサイズの場合，平均値や最小値は多少異なった値になっている．これは，詳細設計モデル構築の際に，パケットを複数まとめて送信する処理など，複雑化を防ぐ構造が実際の NS-2 で実行される処理と異なっているためである．しかし，全体としてある程度近い傾向にあるのではないかと考えられる．

5.3　簡単化モデルによる検証結果

詳細設計モデルのシミュレーションによって得られたデータを解析し，各パラメータの確率密度分布を生成する．その分布より送信スループット値を検証する，簡単化モデルを構築した．なお，キューサイズは 64KB として計測を行っている．それぞれの分布については，スペースの都合上省略する．

送信スループット値の検証で用いる検証式は，“P =?[Throughput $< xU$ Throughput $> x$]” としている．これは，スループット値が x となる確率について調べる検証式である．図 4 が簡単化モデルでの検証結果であり，グラフにおける横軸が送信スループット値で，縦軸がその送信スループット値の生起確率である．メディアサーバの送信スループットが横軸の送信スループットとなる確率が，縦軸の確率値に対応している．

図 4 を見ると，116kbps までが生起確率 1 となり，1000kbps で 0 になっている．これは，116kbps までは理論上必ず保証されるという最小値を示し，1000kbps 以降は生起することが無い，つまり最大値を示す結果である．また，116kbps よりも高い送信スループット値になるにつれ，少しずつ確率が少なくなっていき，1000kbps においては 0.1 程度の確率となっている．これは，RTSP の輻輳制御時において，横軸に対応する送信スループット値が保証される確率を示している．以上の結果より，ラウンドトリップタイム，パケット損失率の確率分布を用いた簡単化モデルにより，送信スループットの定性値を求めることができた．

5.4　実験時間

実験時間は，NS-2 のシミュレーションが約 3 分，詳細設計モデルに対する PRISM の 1 サンプルシミュレーションが約 2,3 秒，そして簡単化モデルに対する PRISM の検証が約 38 秒であった．なお，詳細設計モデルにおける，ラウンドトリップタイムとパケット損失率のシミュレーションによるデータ解析は，それぞれ約 10 時間程度である．

簡単化モデル自体の検証時間は数十秒程度であるが，簡単化モデル構成のための，詳細設計モデルのシミュレーションによるデータ解析に非常に時間が掛かっている．これは，詳細設計モデルにおけるラウンドトリップタイムやパケット損失率のデータについてより信頼性の高いデータを得るために，シミュレーションのサンプル数を出来る限り増やしているからである．

5.5　考　　察

シミュレーション結果では，ネットワークシミュレータ NS-2 と大差ない結果を得ることができた．これは，実時間ネットワークシステムを適切にモデル化を行えていることがわかる．キューサイズが 32KB のときに，PRISM のモデルではパケット損失率の挙動が大きく，NS-2 のモデルでは比較的小さい．これは，モデルの状態数の削減を図るために，TCP や RTP のモデル化を，パケットを一度に複数個送出するような設計にしたために，パケット損失率が急に増加してしまうからである．その影響もあって送信スループットの最小値は外れ値となっている．しかし，全体として平均値や分散は近い値となっている．

また，簡単化モデルの検証結果であるが，シミュレーション上の最小，最大の送信スループット値にと比較した場合，スループットの最小値は，116kbps よりも大きく，最大値も同様に 1000kbps よりも小さくなっている．よって，簡単化モデルの検証結果に関しても，ある程度信頼性のおける結果であると考えられる．最も重要な点は，シミュレーション結果があくまで 1 サンプルのシミュレーション結果である一方，検証結果における送信スループット値は定性的な値である，ということである．つまり,「理論的にどんな場合でもこれらの最小値，最大値に収まる」，という結果を示している．ただし，簡単化モデルは送信スループット値を検証するためのモデルであるため，他の特性に関して検証を実行したい場合は，その部分に特化した別のモデルを構成する必要がある．

6.　手法の一般化について

本章では，本稿で行った実時間ネットワークシステムの PRISM へのモデル化から検証に関して，一般的な検証までの方法論として用いるための考察を行う．

本稿の例題では，まず実時間ネットワークシステムの特性 (通信プロトコル，ルータのバッファ制御など) を細かくモデル化した詳細設計モデルを構築し，PRISM を用いてシミュレーションを行った結果，実際のネットワークシミュレータと同等の結果を得ることができた．詳細設計モデルでは，パケット損失やラウンドトリップタイムなど多くの特性について，シミュレーションによる解析は可能であるが，システムの振る舞いを詳細にモデル化しておりモデル自体が複雑になっているため，検証を実行することができない．そこで，重要な特性の一つである送信スループット値の検証を目的とし，2. で述べたスループット推定式に基づき，簡単化モデルを構成した．簡単化モデルでは，推定式に必要なパラメータの確率分布を詳細設計モデルのシミュレーションから解析し，ラウンドトリップタイムとパケット損失率を決定する確率的な遷移としてモデルに与えている．この簡単化モデル上で，送信スループット値の検証を行った．

本稿のケーススタディより，TCP や UDP トラフィックなどが混在する環境において，RTSP が提供可能な最大，最小スループットを検証する一般的な方法論を以下のように再構成できる．RTSP が定める RTP のスループット値は，2. において示したスループット推定式によって定められる．この推定式では，ラウンドトリップタイムやパケット損失率といったパラメータ設定に基づいてスループットが決定されるため，各パラメータが取りうる値の範囲を適切に定める必要がある．そこで，まず検証したいネットワーク上の各通信のトラフィックや，ルータのバッファ制御などを詳細に設計したモデルを構築し，必要なパラメータであるラウンドトリップタイム，パケット損失率の解析を PRISM のシミュレーションによって行う．それらの値のヒストグラムから各区間の確率密度を定め，その区間の確率密度を，区間中央値が生起する確率として近似する．この操作によってラウンドトリップタイムやパケット損失率が，取り得る値とその確率についてのデータを分析することができる．その分布を PRISM の簡単化モデル上に組み込み，ラウンドトリップタイムとパケット損失率を確率分布から決定させることで，送信スループット値を算出する簡単化モデルを構築する．

簡単化モデルの検証を行うことで，最大，最小スループット値だけでなく，送信スループットの取り得る値が生起する確率について検証することができる．

7. あとがき

本稿では，実時間ネットワークシステムを詳細にモデル化し，モデル検査ツール PRISM に与えることで，システムのスループット値の変動について視覚的に示した．NS-2 におけるシミュレーションによる計測値と比較し，両者が近いスループット値を導出していることを示した．さらに詳細設計モデルで得られたラウンドトリップタイムとパケット損失率を利用して簡単化したモデルを構築することで，詳細設計モデルでは困難であった，システムが出しえるスループットの定性的な最大値と最小値，および生起確率を算出することができた．

今後の課題としては，ラウンドトリップタイムとパケット損失率の相関関係に基づいた簡単化モデルの構成について考えている．本稿の簡単化モデルでは，二つのパラメータは確率分布に基づいて独立に決定される．しかし，実際のネットワークシステムにおいては，両パラメータはある程度相関があると考えられる．この課題に対応するためには，二つのパラメータを同時に決定するような確率分布を作成するなど，どちらかのパラメータが，もう一方に対しある程度依存した形で値を決定していくような構成としていく必要がある．

謝辞 本研究の一部は科学研究費補助金基盤 C (21500036) と文部科学省「次世代 IT 基盤構築のための研究開発」(研究開発領域名：ソフトウェア構築状況の可視化技術の開発普及) の助成による．NS-2 に関する多くの知識や示唆を頂いた大阪大学の廣森聡仁助教と本研究の一部を行った長井栄吾氏に感謝する．

文　献

[1] M. Kwiatkowska, G. Norman and D. Parker : “PRISM 2.0: A Tool for Probabilistic Model Checking,” In Proc. 1st International Conference on Quantitative Evaluation of Systems (QEST’04), pp.322–323, 2004.

[2] “The Network Simulator - ns-2,” available at http://www.isi.edu/nsnam/ns/.

[3] E. J. Kim, K. H. Yum, and C. R. Das : “Calculation of Deadline Missing Probability in a QoS Capable Cluster Interconnect,” IEEE International Symposium on Network Computing and Applications (NCA’01), pp.36, 2001.

[4] M. Handly, S.Floyd, J.Padhye, and J.Widmer : “TCP friendly rate control (TFRC): protocol specification,” Request for Comments (RFC)3448, 2003.

[5] M. Kwiatkowska, G. Norman, D. Parker and J. Sproston : “Performance Analysis of Probabilistic Timed Automata using Digital Clocks,” In Proc. Formal Modeling and Analysis of Timed Systems (FORMATS’03), Vol. 2791 of LNCS, Springer-Verlag, pp.105-120, 2003.

[6] H. Hansson and B. Jonsson : “A logic for reasoning about time and probability,” Formal Aspects of Computing, vol. 6, pp.512-535, 1994.

[7] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton : “Verifying continuous time Markov chains,” LNCS 1026, pp.499–513, 1996.

[8] Y. Taniguchi, A. Ueoka, N. Wakamiya, M. Murata, and F. Noda, “Implementation and Evaluation of Proxy Caching System for MPEG-4 Video Streaming with Quality Adjustment Mechanism,” in Proc. of The 5th AEARU Workshop on Web Technology, pp.27-34, 2003.

[9] 長井 栄吾, 岡野 浩三, 楠本 真二 : “確率的モデル検査ツール PRISM によるリアルタイム分散システムのネットワーク遅延を考慮した検証手法について,” 電子情報通信学会研究報告. SS, ソフトウェアサイエンス 105(597), pp.19-24, 2006.

[10] D. Mahrenholz and S. Ivanov: “Real-Time Network Emulation with ns-2,” In 8th IEEE International Symposium on Distributed Simulation and Real-Time Applications, ser. 8. IEEE Computer Society, pp. 29-36, October 2004.