アクルーアル型故障検出器 ACCMOS の実装と評価

林 原 尚 浩[†1]

広帯域ネットワークの普及や情報家電の登場により各家庭でもホームネットワークを構築し，様々なサービスに依存しつつある．しかし，ホームネットワークでは企業で導入しているようなモニタリングシステムは，設置や監視設定などが複雑なため，一般家庭で導入することは困難である．

本研究では，ホームネットワークなどの比較的小規模なシステム管理のためのモニタリングシステムの開発を目的として，その中核となる故障検出器 ACCMOS の実装とその性能評価を行った．ACCMOS は，アクルーアル故障検出方式に基づき，ネットワークの状況に適応した適切なシステムの監視を行い，監視対象の状態に関する細粒度の情報を提供することができる．

Implementation and Performance Evaluation of the ACCMOS Failure Detector

Naohiro HAYASHIBARA[†1]

In this paper, I show an implementation of the accrual failure detector model, that I call the AtCCMOS failure detector as a core component of a monitoring system for home network. ACCMOS can dynamically adjust to the current network condition, and can adjust to diverse requirements of users simultaneously. It provides information of monitored nodes as the continuous scale, called suspicion level, instead of binary values. I have made several sets of experiments in IEEE802.11n wireless LAN with several parameters to evaluate this implementation.

†1 京都産業大学 コンピュータ理工学部
Faculty of Computer Science and Engineering, Kyoto Sangyo University

1. はじめに

企業などのサービスを提供しているシステムは，サーバ等の故障によるサービスの停止で多大な損害を被る可能性があるため，管理者を置いて常時，システムの監視を行っており，管理者支援のためのモニタリングシステムも多く開発されている．一方，FTTH などの広帯域ネットワークの普及により，家庭内でも小規模なネットワーク (ホームネットワーク) を構築し，コンピュータなどを接続しているが，ホームネットワークでは企業で導入しているようなモニタリングシステムは，設置や監視のための設定などが複雑なため導入することは困難である．また，このようなモニタリングシステムは機能が豊富である一方で，システム管理の経験がない人にとっては不必要な情報まで提供するため，逆にシステムの状況を理解することが難しくなる場合もある．

ホームネットワークにおいては，将来的に様々なネットワーク家電が導入され，日常生活がそれらのサービスに強く依存することも考えられる．このような場合，家電によるサービスは生活と直結するため，それらの監視，故障の発見は重要性を増すことが考えられる．例えば，防犯装置がホームネットワークに接続している場合，防犯装置の故障は，居住者の身の危険を招く可能性もあるため，このようなノードのモニタリングは重要である．

本研究では，ホームネットワーク向けのユーザビリティの高いモニタリングシステムの開発を目的としている．その第一段階として，監視対象ノードをモニタする故障検出器 ACCMOS の実装を行い，性能評価を行った．この故障検出器は，2004 年に林原らによって提案され，近年 Facebook [1] や OurGrid [2], APPIA [3] などの実用的なサービスやプラットフォームで採用されている，アクルーアル故障検出器 (Accrual Failure Detectors) [4, 5] に基づいて実装されている．

2. 故障検出器

故障検出器は，非同期分散システムにおいて分散合意アルゴリズムや全順序ブロードキャストなどの耐故障分散アルゴリズムを実装するために不可欠なコンポーネントとして用いられてきた．本章では，既存の故障検出

器，アクルーアル故障検出器とそれぞれの実装について述べる．

2.1 故障検出器の理論的モデル

2.1.1 非信頼性故障検出器

耐故障分散システムの分野で用いられてきた故障検出器は，監視対象のプロセスの中で，故障である疑いがあるプロセスのリストを出力する一種のオラクルであると定義されている．1996 年に Chandra と Toueg によって，非同期分散システムにおける分散合意問題を解決するために必要な故障検出器のクラスとそれらの定義が，非信頼性故障検出器 (Unreliable Failure Detector) として提案された [6]．

2.1.2 アクルーアル故障検出器

アクルーアル故障検出器は，故障検出器の理論的なモデルである [4]．従来の非信頼性故障検出器モデルが非同期システムでは実装不可能なモデルであるのに対し，このモデルは実装可能であり，さらに非信頼性故障検出器モデルへの変換アルゴリズムを持つ [5]．これは非信頼性故障検出器を前提とした全ての耐故障分散アルゴリズムに適用可能であることを示し，理論と実装の橋渡しをする役割を担っている．

アクルーアル故障検出器は，監視対象ノード (もしくは，プロセス) の故障している度合いを示す *suspicion level* という値を出力する故障検出器として定義されている．今，監視元ノード q が監視対象ノード p を監視している場合を考える．アクルーアル故障検出器はノード q 上で動作している．p について時刻 t に問い合わせを行った場合，アクルーアル故障検出器は時刻 t における p の故障している度合いを示す連続的な値を出力する．従って，ノード p の suspicion level $susp\_level_p(t)$ は以下のように定義できる．ただし，$\mathbb{T}$ は時刻の集合，$\mathbb{R}^+$ は正の実数の集合を示す．

$$susp\_level_p(t) : \mathbb{T} \rightarrow \mathbb{R}^+ \tag{1}$$

また，上記で定義した $susp\_level_p(t)$ は以下の性質を満たす．

Property1 (漸近的完全性) ノード p が故障していれば, $susp\_level_p(t)$ は時間の増加と共に ∞ に近づく．

Property2 (将来的な単調増加性) ノード p が故障していれば, ある時刻 t_m が存在し，t_m 以降 $susp\_level_p(t)$ が単調に増加する．

Property3 (上限の存在) ノード p が正常であれば，またそのときに限り，$susp\_level_p(t)$ は上限値を持つ．

Property4 (リセット) ノード p が正常ならば, $\forall t_0 \in \mathbb{T}$ において，$susp\_level_p(t) = 0$ となる時刻 $t \geq t_0$ が存在する．

アクルーアル故障検出器に基づいた実装を行う場合，出力する suspicion level の値は上記の性質を満たす必要がある．しかし，監視対象ノードの監視方法や suspicion level の算出に関しては自由な実装が可能である．

2.2 故障検出器の実装

故障検出器の実装には大きく分けて二種類存在する．一つはタイムアウト型故障検出器と呼ばれるもので，パラメータとしてタイムアウト Δ_{to} を設定し，Δ_{to} を用いて監視対象ノードの故障の可否を判定し出力する．これは，故障検出器の出力値だけに注目すると， 2.1.1 節で述べた非信頼性故障検出器モデルに近い．もう一つは，2.1.2 節で述べたアクルーアル故障検出器モデルに基づいた故障検出器である．この故障検出器は監視対象の故障している度合いを示す suspicion level を出力するが，監視を行っている各ユーザやアプリケーションは，故障判定を行うために suspicion level に対するしきい値 Φ を設定する．例えば，あるユーザ A が監視対象ノード p を監視している場合，どれくらいの故障している度合いで p を故障と判定するかを Φ_p^A として持つ．

2.2.1 タイムアウト型故障検出器

一般的な故障検出器の多くは，このタイムアウト型故障検出器に分類される．このタイプの故障検出器は，まず監視対象ノード p の故障を判定するためのタイムアウト値 Δ_{to} をパラメータとしてユーザ，もしくはプロセスから与えられる．初期状態において，全ての監視対象ノードは *trust* という状態に設定されている．

その後，一定時間 Δ_i 毎に p と q の間でメッセージ通信を行い，p からのメッセージが q において Δ_{to} 以内に受信できない場合，p が故障の疑いがあるため，*suspect* という状態へ移行する．ユーザやプロセスが p の状態を問い合わせた場合，故障検出器はその時点での p の状態 $\{trust, suspect\}$ を返す．図 1 では，ハートビート型と Ping 型の故障検出器における故障判定を示している．

タイムアウト型故障検出器は，実際に大規模なシステム管理のためのモニタリングシステムに用いられている．それらは，故障検出以外にも，監

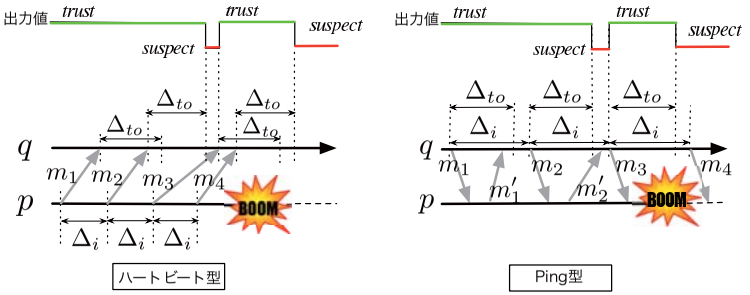


図 1 ハートビート型と Ping 型の故障検出器における故障判定とタイムアウト Δ_{to}

視対象ノードのサービスの監視や統計情報を可視化するなどの機能を持っているものもある．以下に，いくつかの代表的なモニタリングシステムを挙げる．

2.2.1.1 代表的なモニタリングシステムの実装

モニタリングシステムの実装は，Nagios [7], Hobbit [8], Hinemos [9] などを始めとしてとして数多く存在する．特に，オープンソースの Nagios は，多くの機能がプラグインとして実装されているため拡張性が高く．VRML を用いた 3D ネットワーク図の表示などシステムの構成を視覚的に把握しやすい．

2.2.1.2 適応型故障検出器

適応型故障検出器 (Adaptive Failure Detectors) は，ネットワークの状態に適応した故障検出を行うため，動的にタイムアウトを調整するアルゴリズムを用いたタイムアウト型故障検出器である．

Chen ら [10] は，独自に定義した QoS パラメータと確率的なネットワークのモデルを入力することで，ユーザの要求とネットワークの状況に適応するアルゴリズムを提案した．Bertier ら [11] は，TCP のフロー制御などに用いられている Jacobson のアルゴリズム [12] を用いてタイムアウトを動的に変化させ，ネットワーク適応性の高い故障検出器を実現した．

これらの適応型故障検出器は，共に監視対象ノードが送信するハートビートメッセージを用いて監視を行い，タイムアウトによって故障判定をしている．

2.2.1.3 タイムアウト型故障検出器の特徴

タイムアウトを用いたモニタリングシステムや故障検出器における故障検出の精度は，タイムアウトが適切に設定されているかどうかに依存する．しかし，監視対象ノードの所属するネットワークの特性を熟知した管理者の経験や勘がないと，ネットワークの状況に最適なタイムアウトの値を設定することは困難である．また，ネットワークの状況も絶えず変化することが多い．ネットワーク適応型の故障検出器も存在するが，多くのユーザの異なる故障検出に対する要求 [⋆1] を同時に実現するためには，それぞれの要求に対応したタイムアウトを管理する必要があり，スケーラビリティに難がある．

2.2.2 アクルーアル型 φ 故障検出器

φ 故障検出器 (φ Failure Detector) [4] は，アクルーアル故障検出器モデルに基づいた故障検出器のプロトタイプ実装である．この故障検出器は，故障検出器とユーザやアプリケーションを含めた故障検出に関する一連の処理を以下の 3 つのステップに分けて設計されている．

(1) **モニタリング**．監視対象ノードと通信を行うことによって，その状態を予測するための情報の収集，解析を行う．
(2) **インタープリテーション**．モニタリングによって得た情報により，監視対象ノードの故障判定を行う．
(3) **アクション**．ユーザやアプリケーションが，故障判定を受けて行う動作を指す．

タイムアウト型故障検出器モデルに基づいた多くの実装は，上記のモニタリングとインタープリテーションを故障検出器内部で行っている．特に，インタープリテーションはタイムアウトによって故障判定するため，ボトルネックになり監視対象ノードの情報を要求するユーザやアプリケーションが増えるとパフォーマンスが低下する．

φ 故障検出器はこの問題を解決するために，インタープリテーションを故障検出器から切り離し，ユーザやアプリケーションにおいて行うことによってスケーラビリティの向上を図っている．φ 故障検出器は，モニタリングにおいて監視対象ノード p から送られてくるハートビートの受信間

⋆1 「故障検出に時間がかかっても良いから誤検出を低減したい」,「精度よりも故障検出にかかる時間を短くしたい」などの要求が考えられる．

隔を蓄積する．蓄積したデータが正規分布に従っていると仮定し，正規分布の累積密度関数を用いて時刻 t_{now} における $susp\_level_p(t_{now})$ を表す値 φ_p をユーザやアプリケーションに提供する．

ユーザやアプリケーションは個別に φ_p に対するしきい値 Φ_p を設定し，独立して故障判定を行う（例えば，$\varphi_p > \Phi_p$ となったときに故障と判定）．また，φ_p は連続的な値であるため，一つのアプリケーションに複数のしきい値 $\Phi_p^1, \Phi_p^2, \ldots, \Phi_p^n$ を設定し，φ_p の推移に応じて段階的に，監視対象ノード p の故障に備えた処理を行うこともできる．

φ 故障検出器は，タイムアウト型故障検出器が時間軸であるタイムアウトを固定して故障判定を行うのに対し，要求する故障検出精度を入力として与え，ネットワークの状況と入力された故障検出精度に合わせた故障検出を行う画期的な故障検出器である．

φ 故障検出器と前述の適応型故障検出器との比較実験の結果，ネットワーク適応性に関して同等の性能を持つことが確認された [4]．これによって，φ 故障検出器はアクルーアル故障検出器モデルが持つ，ユーザ要求への適応性やスケーラビリティなどの利点に加え，ネットワーク適応性においても高い性能を持つことを示した．

2.2.2.1 アクルーアル故障検出器モデル及び φ 故障検出器の実用例

世界中で広くサービスを提供するソーシャルネットワーキングサービス Facebook [1] の大規模分散データベース管理システム Cassandra [13] や通信フレームワーク APPIA [3]，グリッドミドルウェアである Our Grid [2] においては透過的なグリッドノードへのアクセスを提供する JIC(Java Internet Communication) [14] などでアクルーアル故障検出器モデルに基づいた故障検出器が実装されている．しかし，これらはそれぞれのプラットフォーム依存の実装であり，一般的なモニタリングシステムとして使用することはできない．

3. ACCMOS の設計と実装

本研究では，ホームネットワークなどの中小規模のシステム向けのモニタリングシステムを開発している．このモニタリングシステムは，初期設定の自動化機構，アクルーアル型故障検出器，GUI などのコンポーネントにより構成される．

本稿では，このモニタリングシステムの中核となる故障検出器 ACCMOS の実装と性能評価に焦点を当てる．

アクルーアル故障検出器モデルに基づいて実装された ACCMOS は，従来のタイムアウト型故障検出器が監視対象ノード p の状態を二値 (例えば，$\{trust, suspect\}$) で表現していたのに対し，φ 故障検出器同様，suspicion level φ_p という連続的な値で表現することによって，より詳細な監視対象ノードの状態を提供することができる．

本章では，一般的なモニタリングシステムの中核となる故障検出器 ACCMOS の実装の詳細について述べる．

3.1 監視対象のモニタリング

今，故障検出器 ACCMOS が動作しているノードを q とし，監視対象ノードを p とする．q が p を監視するためには，それらの間で通信を行う必要がある．φ 故障検出器ではハートビート型の実装を行っている [4]．この場合は，監視対象ノード p にハートビートメッセージを一定間隔 Δ_i に送信する実装を導入し，Δ_i は p と q の間で共有する必要がある．Ping 型の実装は，RFC792 で標準化されている ICMP (Internet Control Message Protocol) を用いて行うため，p に特別な実装を施す必要が無く，また多種のノードを監視することができる．そのうえ，Δ_i は q の故障検出器のみ知っていればよいため，メッセージの送信間隔は故障検出器が任意に設定することができる (図 1 参照)．

ACCMOS では，より多種のノードを監視することが可能な Ping 型の通信によって監視対象ノードのモニタリングを行う．ACCMOS の実装は，基本的に Java で行ったが，Ping 型の通信に関しては C 言語で ICMP パケットの送受信を実装し，JNI を用いて呼び出すようにした．

3.2 データの蓄積と suspicion level φ の計算

ACCMOS の実装の概略を図 2 に示す．

まず，ICMP Echo/Reply メッセージによってノード q, p 間のラウンドトリップ時間を得る．ACCMOS は，φ_p を計算するために，ラウンドトリップ時間を蓄積する．蓄積するデータ sum には次のように重み付けされる．

$$sum = sum \times \omega + RTT_{new} \times (1 - \omega) \tag{2}$$

ω $(0 < \omega < 1)$ は既に蓄積されたラウンドトリップ時間の重みを示し，新

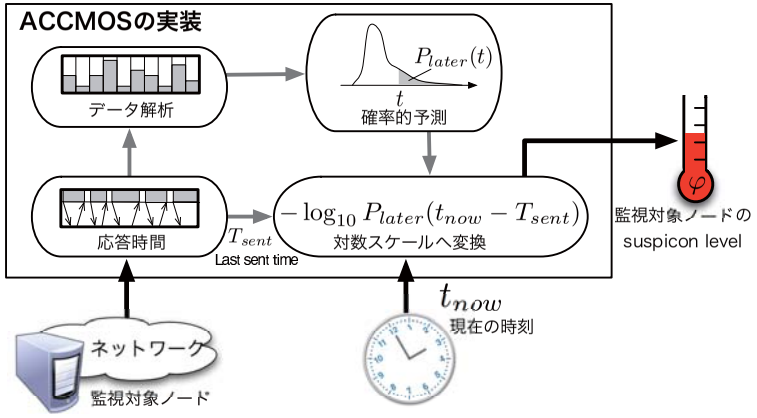


図 2　ACCMOS の実装

たに得たラウンドトリップ時間 RTT_{new} の重みは $1-\omega$ となる．今回の実装では，$\omega \in \{0.8, 0.6\}$ の 2 種類とし，RTT_{new} が sum に与える影響を変化させることができる．ICMP Echo パケットを最後に送信した時刻を T_{sent}，現在の時刻を t_{now} としたとき，t_{now} における p が正常である確率[⋆1]は $P_{later}(t_{now} - T_{sent})$ となる (図 2 参照)．時刻 T_{sent} に q から p へ送信した ICMP Echo パケットに対応する Reply パケットが，q において既に受信されている場合，$P_{later} = 1$ となる．この値は，蓄積されたラウンドトリップ時間から確率分布関数を用いることによって計算される．ある ICMP Echo/Reply パケットによって計測されるラウンドトリップ時間は他の ICMP パケットと独立でかつ ICMP パケットはランダムに遅延すると仮定する．そのため確率分布関数として指数分布関数を用いて実装を行った．

suspicion level φ_p は監視対象ノード p が故障している度合いを示す．従って，P_{later} が減少するにつれて φ_p は増加しなければならない．ただし，ネットワーク越しに監視している故障検出器は p の故障を 100%検出することはできないので，P_{later} は 0 に限りなく近づくが，0 にはならない．φ_p はこの様な性質を満たすために，以下に示すとおり，P_{later} を対数スケールに変換して算出される．

$$\varphi_p(t_{now}) \stackrel{\text{def}}{=} -\log_{10} P_{later}(t_{now} - T_{sent}) \tag{3}$$

監視対象ノード p が故障している場合，ノード q から送信された ICMP Echo パケットに対応する ICMP Reply パケットは戻ってこないので，φ_p は時間の経過と共に ∞ に近づく (漸近的完全性, 将来的な単調増加性)．一方，p が正常であれば，q において p からの ICMP Reply パケットが受信されるため，$\varphi_p = 0$ となる (上限の存在，リセット)．これにより，φ_p は 2.1.2 節で定義した suspicion level の性質を満たすことが分かる．

3.2.1　監視対象ノードの故障判定

ACCMOS は監視対象ノード p の故障判定をユーザやアプリケーション側で個別に行うことによって，ユーザの要求に適応した故障検出サービスの提供とスケーラビリティを両立させている．監視対象ノード p の suspicion level φ_p は，ACCMOS に問い合わせを行うことで得ることができる．ユーザやアプリケーションはそれぞれの要求を反映したしきい値 Φ_p を持ち，$\varphi_p > \Phi_p$ となったときに故障と判定する．しきい値 Φ_p は，監視対象ノード p の故障している度合いがどのくらい高まれば故障と判定するかを示す．例えば，$\Phi_p = 1.0$ ならば，故障している確率が 90% 以上 ($P_{later} < 0.1$) になれば故障と判定することを意味する．Φ_p の値を上げれば，故障検出に時間はかかるが，精度の高い故障検出を行うことができる．一方，Φ_p の値を下げれば，精度は下がるが，故障検出にかかる時間は短くなる．

Φ_p はそれぞれのユーザやアプリケーションが独自に設定するため，ACCMOS は φ_p を提供するだけで，個々の要求した故障検出精度に応じた故障検出サービスを同時に提供することができる．

3.2.2　ACCMOS の利点

ACCMOS は監視対象ノード p の情報を φ_p という連続的な値として提供するため，この値を温度計のように表示することで監視対象ノードの状態を分かりやすくすることができる (図 2 参照)．また，ACCMOS は故障判定のために要求する故障検出精度 Φ_p をパラメータとして与えるため，トラフィックの傾向が変化しても与えられた精度で適切に監視し，ユーザ

⋆1 つまり，q において p からの ICMP Reply パケットが t_{now} 以降に到着する確率

による設定変更を必要としない．

4. 評 価 実 験

4.1 故障検出器の評価指標

評価基準は Chen らによって提案された故障検出器の QoS 指標を用いる [10]．本実験では以下の評価指標を用いる．

Definition1 (平均誤検出レート λ_M) 故障検出器が単位時間あたりに故障していない監視対象ノードを誤検出する平均回数

Definition2 (正検出確率 P_A) 故障検出器が監視ノード p の状態に関する正しい情報を提供する確率

λ_M は単位時間に誤検出を行う平均回数であるが，誤検出回数が同一でも誤検出している時間 $^{\star 1}$が異なる場合は，λ_M が同一でも P_A が異なる．これらの指標によって ACCMOS における故障判定の偽陽性 (false positive) を評価する．故障判定の偽陽性は分散合意アルゴリズムの性能低下を招くなどの弊害があり [15]，λ_M や P_A は故障検出器の性能評価指標として広く用いられている．

4.2 実験環境と ACCMOS の設定

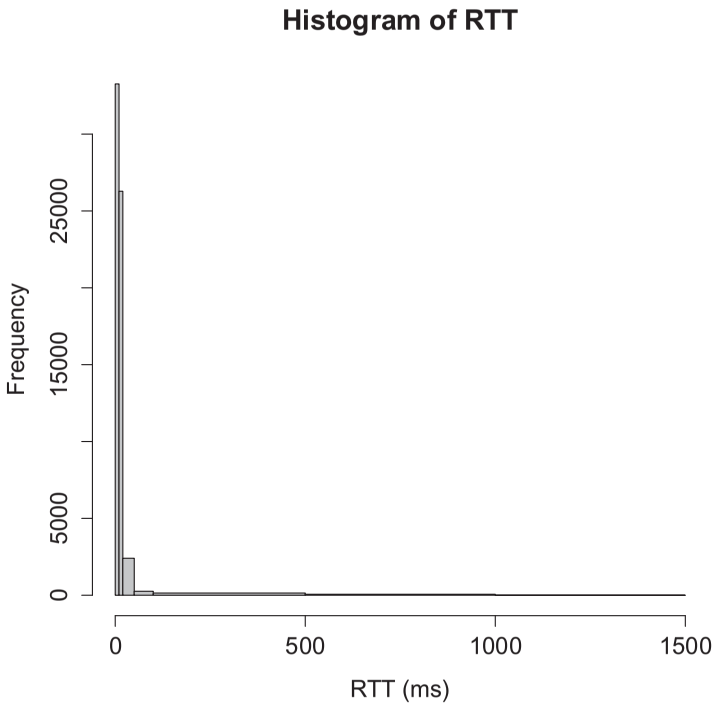
実験は各家庭でも導入が進んでいる IEEE802.11n 無線ネットワークの同一サブネット内で行った．実験を行った環境におけるラウンドトリップ時間の分布は図 3 の通りである．

監視対象ノード p と ACCMOS が動作しているノード q は共に常に正常に動作している．実験の結果としては，q において動作している ACCMOS の平均誤検出レート λ_M と正検出確率 P_A を得ることができる．

ACCMOS は，重みを $\omega \in \{0.6, 0.8\}$ から選択して，蓄積されたデータと最新のラウンドトリップ時間が φ_p の計算に影響を与える比率を変化させることができる．ユーザが与える故障判定のしきい値は，$\Phi_p \in \{0.7, 1.0, 2.0, 3.0\}$ (それぞれ，$P_{later} = \{0.2, 0.1, 0.01, 0.001\}$ に対応) とする．この 2 種類のパラメータの組み合わせによる 8 通りの設定について，それぞれ 3 時間実験を行った．

4.3 実 験 結 果

前述のネットワーク環境において実験を行い，実験結果として図 4 が得られた．平均誤検出レート λ_M は，ACCMOS が 1 秒間に起こす誤検出の平均回数を示している．今回の実験では，2 種類の重み $\omega \in \{0.6, 0.8\}$ で φ_p を計算したが，ほぼ同様の結果となった．λ_M，P_A に関して，$\Phi_p = 0.7$ のとき $\omega = 0.8$ が $\omega = 0.6$ より良好な結果となっているのは，ラウンド

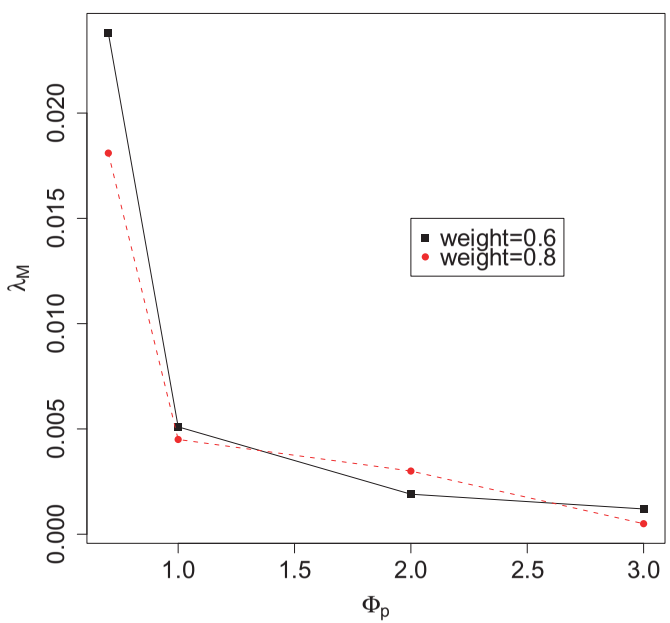


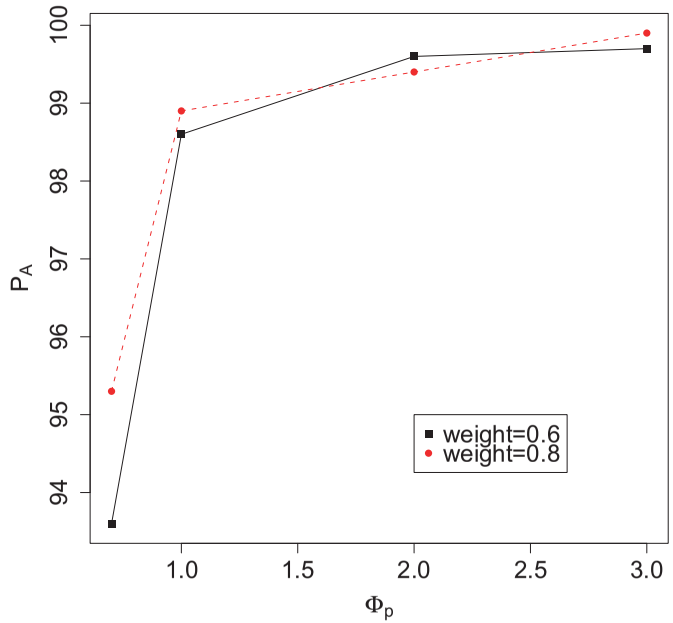
階級 (ms)	0 - 10	11 - 20	21 - 50	51 - 100
度数	33,264	26,282	2,405	257

階級 (ms)	101 - 500	501 - 1000	1001 - 1500
度数	142	63	1

図 3 実験環境におけるラウンドトリップ時間 (RTT) の分布

$^{\star 1}$ 図 1 において p は正常であるが，q の出力値が *suspect* となっている時間





指標	ω	$\Phi_p = 0.7$	$\Phi_p = 1.0$	$\Phi_p = 2.0$	$\Phi_p = 3.0$
λ_M	0.6	0.0238	0.0051	0.0019	0.0012
	0.8	0.0181	0.0045	0.0030	0.0005
P_A (%)	0.6	93.6	98.6	99.6	99.7
	0.8	95.3	98.9	99.4	99.9

図 4 実験結果: 平均誤検出レート λ_M と正検出確率 P_A

トリップ時間の標準偏差が $\omega = 0.6$ のとき 43.142，$\omega = 0.8$ のとき 5.438 となっており，$\omega = 0.6$ の時にネットワークの状態が著しく不安定だったことが原因であると考えられる．

また，いずれの設定においても，要求された検出精度 [⋆1] を上回るか，ほぼ同様の故障検出精度を示している．これは，実験環境のラウンドトリップ時間の分布に対して，概ね指数分布でうまくモデル化できていると言える．

LAN における φ 故障検出器の実験結果は，$\Phi_p = 1$ のとき $\lambda_M = 0.0897$，$\Phi_p = 2$ のとき $\lambda_M = 0.0249$，$\Phi_p = 3$ のとき $\lambda_M = 0.0063$ であった [16]．パケットロス率に関して言えば，φ 故障検出器の実験環境は 0.05%であり，今回の実験環境は 0.22%であったので，ACCMOS は，パケットロス率がより高い環境で実験を行ったことになる．従って，ACCMOS は偽陽性に関して φ 故障検出器より性能が大幅に向上していると言える．

5. まとめと展望

本稿では，アクルーアル型故障検出器 ACCMOS の設計と実装について述べ，無線ネットワーク環境における実験により性能評価を行った．

評価実験の結果，ACCMOS は無線 LAN 上で高精度の故障検出を実現することが確認された．また，同様の環境において，φ 故障検出器よりも監視対象ノードの誤検出を大幅に低減することも確認された．

今後，ACCMOS に加えて，手軽にホームネットワーク環境内のノードを監視できるようにするために，近傍ノードを探索し，それらの監視を開始する初期設定自動化機構と ACCMOS の出力を分かりやすく表示するための GUI を実装する．

謝辞　本研究は科研費 (No. 20700072)，及び京都産業大学 総合研究支援制度 (No. 530) の助成を受けたものである．

参 考 文 献

1) “Facebook.” http://www.facebook.com.
2) “Our grid.” http://www.ourgrid.org.
3) “Appia.” http://appia.di.fc.ul.pt.
4) N.Hayashibara, X.Défago, R.Yared, and T.Katayama, “The φ accrual failure detector,” in *Proc. 23nd IEEE Int’l Symp. on Reliable Distributed Systems (SRDS’04)*, (Florianópolis, Brazil), pp.66–78, October 2004.
5) X. Défago, P. Urbán, N. Hayashibara, and T. Katayama, “Definition and specification of accrual failure detectors,” in *Proc. Int’l Conf. on Dependable Systems and Networks (DSN’05)*, (Yokohama, Japan), pp.206–215, June 2005.
6) T.D. Chandra and S.Toueg, “Unreliable failure detectors for reliable distributed systems,” *Journal of the ACM*, vol. 43, no. 2, pp.225–267, 1996.
7) “Nagios.” http://www.nagios.org.
8) “Hobbit.” http://hobbitmon.sourceforge.net.
9) “Hinemos.” http://www.hinemos.info.
10) W.Chen, S.Toueg, and M.K. Aguilera, “On the quality of service of failure detectors,” *IEEE Transactions on Computers*, vol. 51, pp.561–580, May 2002.
11) M. Bertier, O. Marin, and P. Sens, “Implementation and performance evaluation of an adaptable failure detector,” in *Proc. of the 15th Int’l Conf. on Dependable Systems and Networks (DSN’02)*, (Washington, D.C., USA), pp.354–363, June 2002.
12) V. Jacobson, “Congestion avoidance and control,” in *Proc. of ACM SIGCOMM’88*, (Stanford, CA, USA), August 1988.
13) “Cassandra.” http://wiki.apache.org/cassandra/.
14) R.Lima, W.Cirne, F.Brasileiro, D.Fireman, and L.D.S. Distribudos, “A case for event-driven distributed objects,” in *In Proc.of DOA’06, LNCS 4276*, pp.1705–1721, Springer-Verlag, 2006.
15) L. M. R. Sampaio, F. V. Brasileiro, W. Cirne, and J. C. A. Figueiredo, “How bad are wrong suspicions? towards adaptive distributed protocols,” in *In Proc. IEEE Intl. Conf. on Dependable Systems and Networks (DSN’03)*, pp.551–560, 2003.
16) N.Hayashibara and M.Takizawa, “Performance evaluation of the φ failure detector,” in *In Proc. of the 8th IEEE ICDCS Workshops (MNSA’06)*, July 2006.

⋆1 しきい値 $\Phi_p = 0.7, 1.0, 2.0, 3.0$ は，要求する故障検出精度が 80%, 90%, 99%, 99.9%に対応する．