情報処理学会論文誌 Vol.56 No.3 767–776 (Mar. 2015)

障害原因解析における構成情報の統計的推論方式

坂下 幸徳[1,a)] 東条 敏[1,b)] 敷田 幹文[2,c)]

受付日 2014年6月25日, 採録日 2014年12月3日

概要：データセンタの管理者を支援する研究として障害発生時に障害原因を特定する障害原因解析技術が登場してきている．この障害原因解析技術を使うためにはサーバ，ストレージ，スイッチなどの構成情報が必要となるが，近年，大規模化，複雑化，さらにはクラウド化が進むデータセンタでは，構成情報の取得が困難になり，適用範囲が狭くなってきている．そこで，本論文では，サーバ，ストレージ，スイッチなどの機器が出力するログファイルを使い，統計的推論方式で構成情報を推定する方式を提案する．統計的推論方式としては，代表的な方式である隠れマルコフモデルとベイズ推定を用いる．これにより，障害原因解析技術の適用範囲を拡大を狙う．本提案方式の試作システムによる実験の結果，最大83%の正解率による構成情報の推定に成功した．

キーワード：障害管理，障害原因解析，データセンタ，大規模，隠れマルコフモデル，ベイズ推定

A Statistical Inference Method of Configuration Data for Root Cause Analysis for Fault Management

Yukinori Sakashita[1,a)] Satoshi Tojo[1,b)] Mikifumi Shikida[2,c)]

Received: June 25, 2014, Accepted: December 3, 2014

Abstract: Root Cause Analysis (RCA) has emerged as a valuable tool for helping administrators identify the sources of problems when failures occur. In order to use RCA, it must have access to configuration data that is configuration of servers, storage arrays and switches. But the configuration data has become problematic in recent years as data centers have been scaled-out to immense size, become more complex, and have migrated to the cloud. As a result, the application range of RCA has continued to shrink. In this paper, we propose a statistical inference method of the configuration data using log files in servers, storage arrays and switches. This proposal method use two typical statistical inference methods that are the hidden Markov model and Bayesian inference. So it is able to expand the application range of RCA. And we experimented using using a prototype system of this proposal method. As a result, we are able to successfully infer the configuration data up to accuracy of 83%.

Keywords: fault management, root cause analysis, data center, large-scale, Hidden Markov Model, Bayesian Inference

1. はじめに

近年，デジタルデータ量の増加や仮想化技術の普及により，データセンタが大規模，複雑化している．さらに，企業や大学などの自組織内の機器を集中管理しプライベートクラウドとして運用する形態 [1] や，外部のプロバイダが提供するパブリッククラウドを利用する形態などによりいっそう大規模，複雑化している．一方，多くのデータセンタでは，これを運用する管理者の数は一定もしくは減少し，

1 北陸先端科学技術大学院大学情報科学研究科
School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, Ishikawa 923–1292, Japan
2 北陸先端科学技術大学院大学情報社会基盤研究センター
Research Center for Advanced Computing Infrastructure, Japan Advanced Institute of Science and Technology, Nomi, Ishikawa 923–1292, Japan
a) sakasita@jaist.ac.jp
b) tojo@jaist.ac.jp
c) shikida@jaist.ac.jp

さらに知識の属人化が進んでいる．そのため，少人数で大規模で複雑なシステムを管理せざるをえず，管理作業が間に合わずサービスを停止してしまうような甚大な損害をビジネスに与えかねない状況である．また，特に障害発生時では，経験の浅い管理者では対応に時間がかかってしまうため，熟練管理者の経験や勘に頼らざるをえないケースも多い．

そこで，管理者の作業負荷の軽減を目指し，サーバ，ストレージ，スイッチなどの機器（以下，インフラ機器と略す）を効率良く維持管理するため，運用管理ソフトウェアの研究が進められている．特に，管理者の作業負荷が高い障害対応向けに，障害発生時の初期切り分けを支援する障害原因解析技術が注目されている．しかし，データセンタが大規模，複雑化，さらにはクラウド化したことで，すべてのシステムの構成を把握することが困難となり，障害原因解析技術の適用範囲が限られてしまっている．

そこで，本論文では，インフラ機器が出力するログファイルから統計的手法を用いて構成情報を推論する方式を提案する．これにより，運用管理ソフトウェアが把握できないインフラ機器の構成情報を補完し，適用できるシステムが限られていた障害原因解析技術の適用範囲を拡大させ，管理者の負荷軽減を狙う．

以下，2 章では関連研究を紹介し，3 章で提案方式である統計的推論方式を用いた構成情報の推定方式の詳細を述べる．次に，4 章で評価プログラムを用いた実験結果を述べ，最後に 5 章で考察を述べる．

2. 関連研究

障害発生時の管理者の負荷を軽減するための管理技術として，障害原因解析技術の研究 [2], [3], [4], [5], [6] が行われている．従来研究の障害原因解析技術について構成と処理の流れを**図 1**，構成情報 DB のスキーマを**図 2** に示す．

障害原因解析技術では，次のステップで障害原因を分析する．

(1) 障害イベントと予想される障害部位を IF-THEN ルール形式で汎用ルールとして事前定義
(2) サーバ，ストレージ，スイッチのインフラ機器の接続関係・設定情報や動作状態のステータスなどの情報を収集し構成情報 DB（Database）へ保存
(3) (1)，(2) よりデータセンタのインフラ機器の具体的なリソースを対象とした解析ルールを生成
(4) インフラ機器から SNMP（Simple Network Management Protocol）などで障害イベントを受信
(5) (4) の原因を (3) の解析ルールを使い障害原因となったリソースを特定

このように，事前に管理者のノウハウを IF-THEN ルールの形式で記述しておき，これを実際のインフラ機器の構成情報と合わせることで解析ルールを作成し，障害原因解析を実現している．しかし，構成情報が収集できないリソースに対しては解析ルールを作成できず，障害原因解析技術を使うことができない．

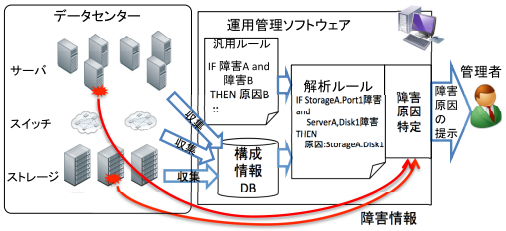

図 1 障害原因解析技術
Fig. 1 Root cause analysis.

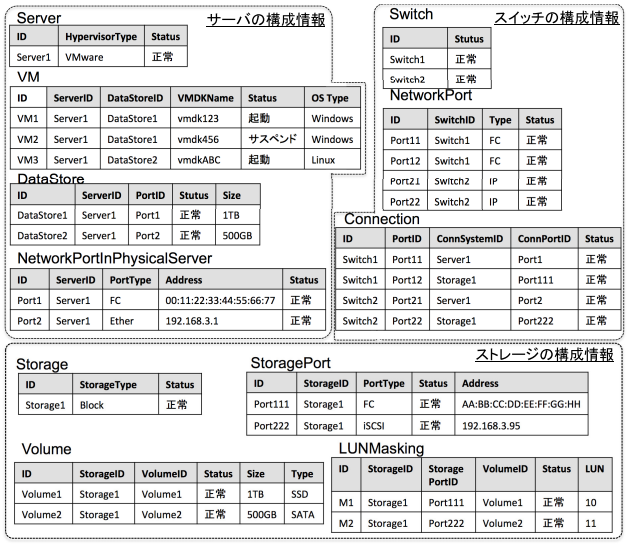

図 2 構成情報 DB のスキーマ
Fig. 2 A DB schema of configuration data.

また，インフラ機器からの構成情報の収集に関しては，SMASH（Systems Management Architecture for Server Hardware）[7] や SMI-S（Storage Management Initiative-Specification）[8] などの標準仕様のインタフェースを利用することにより，異なるベンダの機器であっても構成情報の収集が可能になってきている．しかし，標準仕様のインタフェースをサポートしていない機器や，パブリッククラウドのように，管理者からはインフラ機器の構成が隠蔽されてしまっている環境においては，構成情報の収集ができない．その結果，障害原因解析技術を適用できるシステムが限定されてしまっている．

3. 構成情報の統計的推論方式

3.1 方針

ログファイルは，管理者が行ったインフラ機器の設定内容や警告・障害メッセージなどインフラ機器の構成に関連する情報が出力されており，市販されているほとんどのインフラ機器が出力している．本提案方式では，このログファイルを活用することで，障害原因解析技術の基盤とな

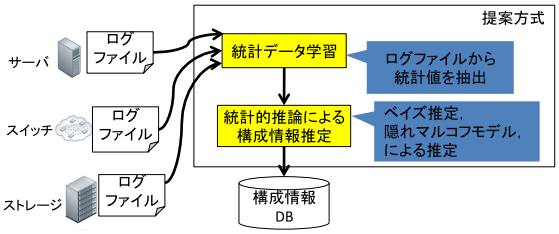

図 3 提案方式による構成情報の推論の流れ
Fig. 3 Flow of configuration information by proposal method.

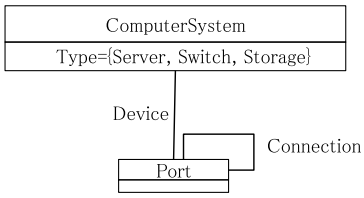

図 4 CIM の一例
Fig. 4 A part of CIM.

る構成情報を推論する．

推論方式としては，ある事象の発生確率を使い推論する隠れマルコフモデル，ベイズ推定，人間の脳をモデル化し推論するニューラルネットワーク，大量データのクラス分類を行うサポートベクタマシン（SVM）などが考えられる．ニューラルネットワークを使えば機器構成を，そのままネットワークになぞらえて出力層からネットワーク構成を推定する問題に帰着でき，SVM であれば，多数の機器間の切り分けをする問題に帰着できる可能性がある．一方，ログファイルの情報の特性としては，同一の機器の情報（サーバ名や IP アドレスなど）が特定のログファイルに限定されず複数のログファイルに出力される点，各機器に異なるフォーマットで出力されるため定まった方式で解析できない点がある．

そのため，ログファイルの特性より，発生確率を用い推論する隠れマルコフモデルとベイズ推定が有効であると考え，本提案方式では，隠れマルコフモデルとベイズ推定の両手法を用いる．

本提案方式の構成情報の推論の流れを**図 3** に示す．2 章の障害原因解析技術の処理ステップ (2) に対し，構成情報が収集できないインフラ機器について以下のステップを実施し，構成情報を補完する．なお，本提案方式の構成情報 DB では，2 章で述べた従来研究と扱う構成情報に変更点がないため同一のスキーマを利用する．

(2-1) インフラ機器からログファイルを収集
(2-2) ログファイルから構成情報に関するメッセージを抽出し，統計データを算出
(2-3) ベイズ推定を使って大まかな構成を把握
(2-4) 隠れマルコフモデルを使って曖昧さのある箇所を詳細化し，構成情報 DB へ格納

このように，構成情報が収集できなかったインフラ機器に対しても，ログファイルを用いた推論により構成情報を補完する．次節より，本提案の推論方式である統計的推論方式について詳細に述べる．

3.2 統計的推論方式

ベイズ推定は，ある事象が発生した際，原因となった事象の発生確率を推論する方式であり，スパムメールのフィルタリングなどで利用されている推論方式である [9]．隠れマルコフモデルは，ある事象の発生確率と，次の事象への状態遷移の確率から，時系列のデータをモデル化する統計的方式であり，自然言語処理の形態素解析などで利用されている推論方式である [10]．

この 2 つの推論方式は，両方式とも確率を使った統計的な推論方式であり，ある事象の発生確率を推定する点においては共通である．しかし，隠れマルコフモデルは，次の事象へ状態遷移の確率を求める前向き推論のため，次に遷移する事象のモデルが定まっていなくても推論できるが，ベイズ推定は，事象の発生の原因の確率を求める後ろ向き推論のため，原因となった事象の候補があらかじめ定まっている必要がある，といった違いがある．

本提案では，両方の方式とも，ある事象を統計的に推定する方式であることに着目し，推定対象とする「事象」を推定する「構成情報」と見なし適用する．また，両方式とも統計的推論である．そのため推論を行うために重要となるのが，観測可能な統計をとるためのデータである．この観測可能なデータとして，あらゆるインフラ機器が出力するログファイルを本提案では対象とする．

次節よりベイズ推定，隠れマルコフモデルを使った構成情報の推定方法について詳細に述べる．

3.3 ベイズ推定による構成情報の推定

ベイズ推定を使った構成情報の推定方式について述べる [11]．ベイズ推定では，とりうる可能性がある構成があらかじめ判明している必要がある，そこで，とりうる可能性がある構成を求める方法として，DMTF（Distributed Management Task Force, Inc.）が定めるシステムの標準モデルである CIM（Common Information Model）[12] を用いる．CIM は，サーバ，ストレージ，スイッチなどシステムの一般的な構成をモデル化しており，多くの運用管理ソフトウェアや OS で構成情報を表現するのに利用されているモデルである．

CIM のモデルの一例を，**図 4** に示す．図 4 のモデルでは，“ComputerSystem” がデバイスとして “Port” を所有し，“Port” は別の “Port” と接続関係にあることを示している．

この CIM を使い，とりうる可能性がある構成を導出する．具体的な例を**図 5** を用いて説明する．図 5 左図のように，Server1 が Switch1 を経由し Storage1 に接続されて

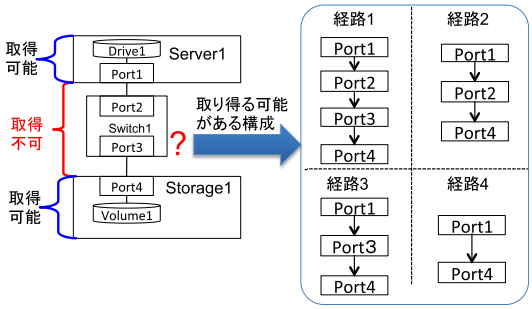

図 5 IT システムの構成例
Fig. 5 Example of assumption system.

```
2012-10-01T07:59:56Z iscsid: connection failed
2012-10-01T07:59:56Z iscsid: connection to 192.168.3.1 failed
2012-10-01T07:59:56Z iscsid: connection to 192.168.3.95 failed
2012-10-01T07:59:56Z iscsid: Login Target: iqn.2004-08.jp.Storage1-
HDD-001D732783AC:sakashita-vm if=default
addr=192.168.3.95:3260 (TPGT:1 ISID:0x1)
::
```

図 6 ログファイル例
Fig. 6 Example of log file.

表 1 ログファイルから抽出した構成情報と統計情報の例
Table 1 Example of extracting configuration information from log file and statistics data.

構成情報	出現回数	発生確率
192.168.3.1 ⇒ 192.168.3.95	433	42%
192.168.3.1 ⇒ Storage1	10	1%
192.168.3.95 ⇒ Storage1	205	20%
Storage1 ⇒ 192.168.3.95	337	33%
Storage1 ⇒ 192.168.3.1	0	0%

おり，Switch1 の構成情報（接続関係）が取得できないケースを想定する．このケースを，図 4 で示されたモデルにあてはめると，図 5 右図に示す 4 つのパターンの経路がとりうる可能性がある経路であることが導出できる．また，導出されたパターンには，CIM より論理的に導出するため，実際には接続することができない構成である経路 2 や経路 3 も導出される．これをベイズ推定にあてはめると，たとえば，Server1 の Port1 が Switch1 の Port2 が接続関係を持っている確率は $P(Port2|Port1)$ となる．

このように，CIM を使いとりうる可能生のある構成情報の導出を行った後，各インフラ機器のログファイルに出力されたリソース情報の発生確率を求め推定する．ログファイルの解析による発生確率の算出例を，**図 6** のログファイルをもとに抽出した構成情報とその発生確率の例を**表 1** に示す．本例では，Server1 のログファイルを対象とし，Port1 に関連したレコードを抽出した後，出現回数をカウントし発生確率を算出している．また，本例では 1 つのログファイルを例に説明しているが，実際の算出では複数のログファイルを対象とし算出する．なお，Server1 の Port1 に付与されている IP アドレスは 192.168.3.1，Storage1 の

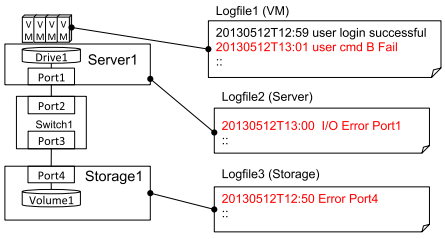

図 7 ログファイルのメッセージ例
Fig. 7 Example of messages in log files.

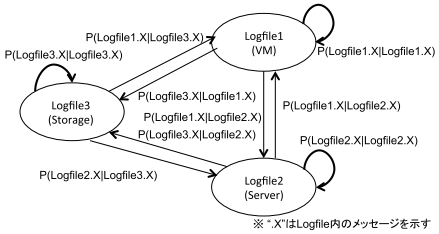

図 8 隠れマルコフモデルによる構成情報の推定方式
Fig. 8 Inference method of configuration information by Hidden Markov Model.

Port4 に付与されている IP アドレスは 192.168.3.95 として例を示す．表 1 のリソース名に記載の '⇒' は，左辺のリソースに関するメッセージが出力された後，右辺のリソースに関するメッセージが出力されたことを示す．

このようにして算出した CIM とログファイルから求めた統計データを使い，図 5 右図に示すすべての経路に関し，ベイズ推定で発生確率を算出し，比較を行うことで構成情報を推定する．

3.4 隠れマルコフモデルによる構成情報の推定

次に，隠れマルコフモデルを使った構成情報の推定方式について述べる．隠れマルコフモデルは，時系列のデータをモデル化することができる方式である．そこで，複数のログファイルに出現するメッセージの関連性に着目する．一例として**図 7** にログファイルのメッセージ例を示す．図 7 の例では，ストレージのログファイル Logfile3 にエラーメッセージが出力された後，そのストレージを利用しているサーバのログファイル Logfile2 にエラーメッセージが出力されている．さらに，そのストレージとサーバを利用している VM のログファイル Logfile1 にもエラーメッセージが出力される．このように，接続関係にあるインフラ機器の間では，出力されるログメッセージには関連がある場合がある．この関連を隠れマルコフモデルで示すと，**図 8** のようになる．

この関連を用いた構成情報の推定方法の一例を示す．ストレージのログファイルに Port4 のエラーメッセージが出力された場合，次にサーバのログファイルにメッセージが出現する確率は $P(Logfile2.\text{“}Port1\text{”}|Logfile3.\text{“}Port4\text{”})$ となる．つまり，ストレージのログファイルに Port4 が出現した直

後に，サーバのログファイルに Port1 が出現した回数をカウントすることで，$P(Logfile2.\text{“}Port1\text{”}|Logfile3.\text{“}Port4\text{”})$ が分かり，Port4 と Port1 が接続関係を持っている確率を求めることができる．

このように，あるリソースに関するメッセージがログファイルに出現した直後に，別のリソースに関するメッセージが出力される回数をカウントし確率を求めることで，リソース間で接続関係を持っている確率を求めることができる．

4. 実験

4.1 概要

本実験では，3 章で示した提案方式の有効性を検証すべく，各推論方式の特徴を明らかにするために，以下の実験を行った．

(1) ログファイルの種類（サーバ，ストレージ，スイッチ）の違いによる正解率の傾向を測定
(2) ログファイルの収集期間による正解率の傾向を測定
(3) 統計的推論方式の適用対象となるシステムの大規模化にともなう正解率の傾向を測定

はじめに，本実験での推論結果における正解率の算出方法について説明する．本実験では，**図 9** に示すようにスイッチの構成情報が取得できないケースを想定し，提案方式でサーバとストレージの接続経路の構成を推定する．推論結果として，図 9 の例ではとりうる接続経路の可能性として 4 パターンある．この 4 パターンには，ポートの論理的な組合せパターンのため，実際にはありえない経路 2，経路 3 のような同一のポートに 2 つの機器が接続されるケースも含まれている．この 4 パターンのうち，実構成の経路である経路 1 を正解として正解率を算出する．一例として，経路 1 が 5 回，経路 2 が 4 回，経路 3 が 3 回，経路 4 が 2 回，推論結果として算出されたと仮定する．この場合，経路 n の出現回数を Rn とすると，正解率は $100 * R1/\sum_{n=1}^{4} Rn$ となり，36%と算出される．なお，本実験では，経路の方向性に関しては同一のものとして扱うものとする．つまり，Port1 → Port2 → Port3 → Port4 と Port4 → Port3 → Port2 → Port1 は同一の経路と判断する．これは，構成情報においては前者と後者の経路は同一の構成であるためである．本実験では，**図 10** の左図のシステム構成を正解の構成とする．たとえば，正解の構成の一例としては Server1 の Port1 → Switch1 の PortA → Switch1 の PortE → Storage1 の Portα がある．提案方式により推定された構成の総数のうち何パターンが正解の構成であったかをカウントし正解率を算出する．

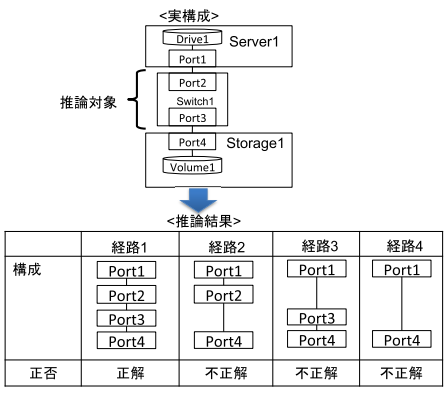

図 9 推論結果の例
Fig. 9 Example of inference result.

4.2 評価プログラムと実験環境

図 10 と**表 2** に本提案方式の評価プログラムおよび評価環境を示す．

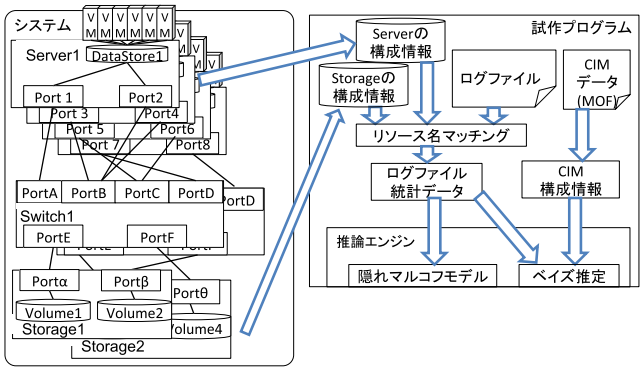

図 10 評価プログラムと評価環境
Fig. 10 The system and program for evaluation.

表 2 開発・実験環境
Table 2 The environment for evaluation.

物理サーバ	PC × 4 台（CPU: Intel Corei7 3.4 GHz, Memory16 GB，Network Port 数 2）
仮想サーバ	VMware ESXi5[*1]
VM	24 VM
ストレージ	Hitachi AMS 2100[*2]（Network Port 数 4, うち 2Port のみ結線し利用）× 2 台
スイッチ	Brocade SilkWorm 3200[*3]（Network Port 数 8, うち 6Port のみ結線し利用）× 2 台
開発環境	MacBook Air[*4]（Intel Corei5，Memory 4 GB） Mac OS 10.8.4, Java6[*5]
構成情報 DB	MySQL5.5.2[*5]
学習用 CIM データ	CIM Schema 2.35.0 の MOF ファイル
学習用ログデータ	16 週間分のログファイル（Size1.4 GB，リソース情報数：228,704 レコード）【対象】仮想サーバの/var 配下のログ一式, スイッチの保守用ログ一式, ストレージの保守用ログ一式
学習エンジンの実行マシン	SGI Altix UV1000[*6]（Intel Xeon 2.66 GHz（1536Core 中 32Core 使用）, Memory12 TB（うち 256 GB 使用））
構成情報推定エンジンの実行マシン	開発環境と同一

[*1] VMware は VMware, Inc. の米国およびその他の国における登録商標です．
[*2] Hitachi AMS は株式会社日立製作所の登録商標です．
[*3] Blocade，SilkWorm は Brocade Communications Systems, Inc. の米国およびその他の国における登録商標です．
[*4] MacBook Air は Apple, Inc. の米国およびその他の国における登録商標です．
[*5] Java，MySQL は Oracle Corporation およびその子会社，関連会社の米国およびその他の国における登録商標です．
[*6] Altix は Silicon Graphics International の商標です．

<フォーマット>
対象機器名1:ログファイル名1:“抽出したリソース名”，対象機器名2:ログファイル名2:“抽出したリソース名”，出現回数

<例>
server1:system.log:“192.168.3.1”,storage1:message.log:“192.168.3.95”,433
server1:system.log:“192.168.3.1”,server1:system.log:“storage1”,10
::

図 11 ログファイル統計データのフォーマット
Fig. 11 A format of statical data of log file.

評価に用いた学習用ログデータは，企業内のプログラム開発向けに実運用しているシステムのログファイルを用いた．なお，本システムでは，プログラム開発で使われるシステムのため，ログファイルのメッセージが最も多く出力されるデバックレベルの設定で運用している．本実験では，図 10 左図に示すシステムで，スイッチの構成が取得できないケースを想定して実験を行った．

評価プログラムでは，構成情報を推定する推論エンジンを実行する前に，ログファイルと CIM を使った学習としてリソースの情報を抽出し，3.2 節に示すように各リソースの統計データを算出しログファイル統計データを出力する．ログファイルからリソースの情報抽出方法として，図 10 のリソース名マッチングモジュールで，Server の構成情報と Storage の構成情報に格納された各リソースの ID や Address の情報および IP アドレスや FC アドレスのフォーマットと，ログファイル中のメッセージ文字列が一致しているものを抽出する．**図 11** にログファイル統計データのフォーマットと例を示す．

CIM データからの構成情報の抽出は，図 10 の CIM 統計モジュールで，テキスト形式で CIM 定義を記述している MOF（Managed Object Format）ファイルを使い，Element クラス名と，Association クラスの定義情報より各 Element クラスの接続情報を抽出する．その後，推論エンジンで，ログファイル統計データを用いて，ベイズ推定と隠れマルコフモデルを使い構成情報の推定を行う．

この学習処理と推定処理を分離した実装は，学習にかかる処理時間と推論にかかる処理時間の差を考慮したためである．学習にかかる時間としては，表 2 の仮想サーバ，ストレージ，スイッチの 16 週間のすべてのログファイルを対象とした場合，学習エンジンの実行マシンに示す高性能な並列計算機で約 1 時間 30 分，開発環境のマシンで約 2 時間 40 分であった．推論エンジンでの推定処理の時間としては，開発環境のマシンで 5 回の平均をとったところ，ベイズ推定は 0.8 秒，隠れマルコフモデルは 2.1 秒であった．このように，学習と推論には大きな時間差がある．また，実行頻度の観点では，学習は週や月単位の定期実行，構成変更時に行えばよく，推論時に毎回実施する必要はない．さらに，隠れマルコフモデルとベイズ推定ともにログファイルの統計データを使うため，ログファイル統計データは同一のものが利用可能である．これらの理由により，評価プログラムでは，学習と推論の処理を分離した構成とした．

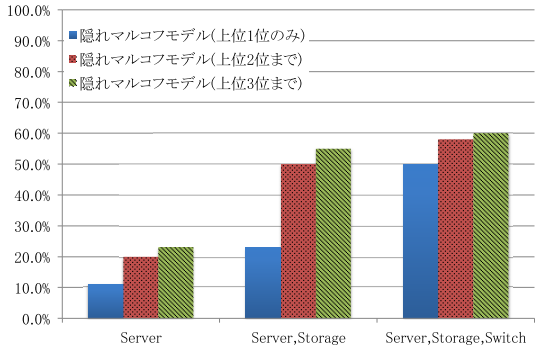

図 12 ログファイル種類による正解率（隠れマルコフモデル）
Fig. 12 Correct answer rate by type of log file (Hidden Markov Model).

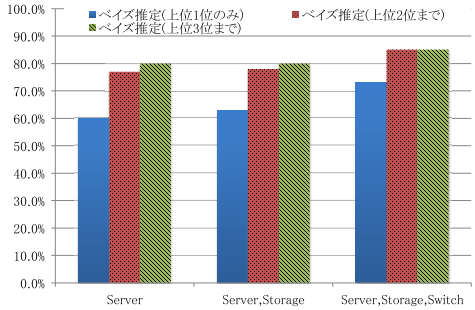

図 13 ログファイル種類による正解率（ベイズ推定）
Fig. 13 Correct answer rate by type of log file (Bayesian Inference).

4.3 ログファイルの種類変化の実験

実験 1 は，サーバのログファイルのみ，サーバ＋ストレージのログファイル，サーバ＋ストレージ＋スイッチのログファイルを使った提案方式による構成情報の推定結果の正解率を測定した．実験に使ったログファイルは，2 週間のログファイルを対象とした．さらに，提案方式の推定結果のうち，確率が高かった上位 1 位から上位 3 位までに正しい構成が含まれているか否かも測定した．実験結果を**図 12**，**図 13** に示す．

実験の結果，隠れマルコフモデルは，サーバのみのログファイルを利用した場合，上位 1 位の正解率は 11%であったのにあったのに対し，サーバ＋ストレージ＋スイッチのログファイルを利用した場合，50%まで向上することが判明した．ベイズ推定は，サーバのみのログファイルを利用した場合の上位 1 位の正解率は 60%と隠れマルコフモデルの約 6 倍の正解率となった．さらに，ベイズ推定は，ログファイルの種類を増やした場合の正解率は 73%となり 13%向上する結果となった．さらに，上位 1 位だけでなく，上位 3 位まで推定結果に含めた場合の正解率は，隠れマルコフモデルは，サーバ＋ストレージ＋スイッチを使った場合 60%に向上，ベイズ推定は 85%と両方式とも 10%以上正解率が向上した．また，誤って推定された構成の上位としては，Server1 の Port1 と Storage1 の Portα を

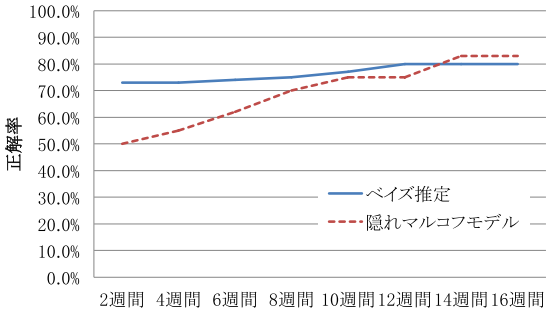

図 14 ログファイルの収集期間による正解率
Fig. 14 Correct answer rate by collection period of log file.

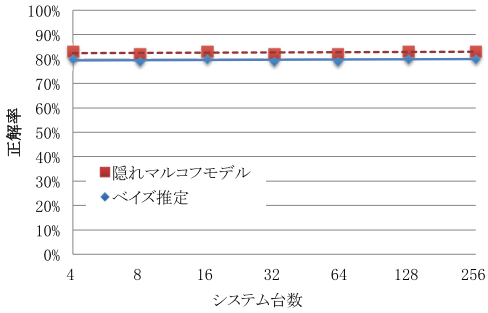

図 15 スケールアウトによる正解率
Fig. 15 Correct answer rate by scale out systems.

直接接続した構成，Server1 の Port1，Switch1 の PortA と Storage1 の Portα を接続した構成であった．これは，単一のログファイル内で連続して出力されているメッセージより推定されている構成であった．

4.4 ログファイルの収集期間変化の実験

実験 2 は，対象のログファイルとして，サーバ+ストレージ+スイッチとし，期間は 2 週間から 16 週間分のログファイルを使い実験を行った．結果を**図 14** に示す．

実験の結果，隠れマルコフモデルは，収集期間が長くなれば長くなるほど，正解率は向上し，2 週間では正解率が 50%であったが 16 週間では 83%まで向上している．2 週目から 8 週目までの間は，正解率の向上が著しいが，10 週目から収束し始める傾向であった．一方，ベイズ推定の正解率は，初期より高かったため隠れマルコフモデルより向上は少ないものの，16 週間で 7%の向上がみられた．また，13 週目にストレージの Port 故障が発生しており，ログファイルへこれまで出現していなかった Port に関するエラーメッセージが出現した．これにより，隠れマルコフモデルの正解率が向上する結果となった．

4.5 システムの規模数変化の実験

実験 3 は，システムの規模数の変化として，2 つのケースについて正解率の変化を測定した．

(1) スケールアウトによる大規模化
(2) スケールアップによる大規模化

スケールアウトによる大規模化は，主にクラウドデータセンタに代表されるケースである．このケースでは，サーバ，ストレージ，スイッチのシステム構成をあらかじめ固定的に定めておき，同じ構成のシステムが，クラウドデータセンタを利用するユーザの数とともに増加させる．このようにスケールアウトするシステムの，代表的なものに CloudStack などテンプレートを用いて構築される IaaS (Infrastructure as a Service) のシステムがある．

スケールアップによる大規模化は，主に性能向上を要求されるようなエンタプライズのデータセンタに代表されるケースである．このケースでは，CPU/メモリの向上が必要な場合はサーバを，Disk I/O の向上が必要な場合はストレージをそれぞれ追加し性能向上を図る．このようにスケールアップを行うシステムの代表的な例として，並列計算処理のシステムがある．

本実験では，まずスケールアウトによる大規模化の正解率の変化を測定した．測定では，4.2 節に示す環境での 16 週間分のログファイル使い，サーバ 4 台，ストレージ 2 台，スイッチ 2 台を 1 システムとし，構成を変えずシステム数を増加させ測定を行った．本測定では，4.2 節のログファイルの日付を変更し，256 システム分まで複製した後，ログファイル中に出力されているリソース名がユニークな値となるよう文字列置換を行うことで，スケールアウト環境のログファイルを擬似的に生成した．なお，学習の処理時間は，表 2 の高性能な並列計算機で約 8 時間であった．測定結果を，**図 15** に示す．

実験の結果，隠れマルコフモデル，ベイズ推定ともにシステムがスケールアウトしても，正解率は低下しないという結果となった．これは，システム数が増加しても，1 システム内の構成は一定であり，それぞれのシステム間の依存関係がないためであると推察する．

次に，スケールアップによる大規模化について測定を行った．測定では，4.2 節の環境での 16 週間分のログファイルを使い，ストレージ 2 台，スイッチ 2 台は固定とし，サーバ台数のみ増加させ測定を行った．本測定では，4.2 節のサーバのログファイルを 256 サーバ分まで複製した後，ログファイル中に出力されているサーバ名がユニークなサーバ名となるよう文字列置換を行った．さらに，仮想ネットワークを用いたサーバのスケールアップを想定し，スイッチのログファイルの置換前のサーバ名の出現箇所の直後に，置換後のサーバ名を変更した同様のメッセージを追加することで，スケールアップ環境のログファイルを擬似的に生成した．一例をあげると，複製したログファイル内の Server1 の Port1 を Server10 の Port10 に変更し，さらにスイッチのログファイル内の Server1 の Port1 の出現箇所の直後に Server10 の Port10 のメッセージを追加した．スイッチのログファイルの例を**図 16** に示す．これによりスイッチとログファイルの複製で作成したサーバが接続

```
2012-10-02T09:53:56Z Server1: port1 connection failed
2012-10-02T09:53:56Z Server10: port10 connection failed
```

図 16 スイッチのログファイル例
Fig. 16 Example of switch log file.

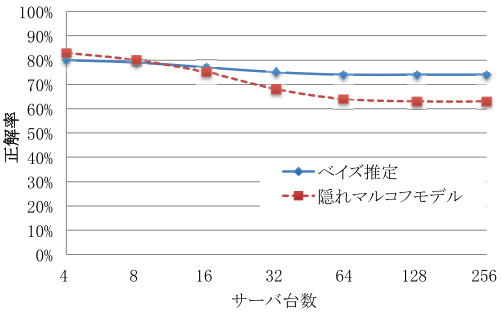

図 17 スケールアップよる正解率
Fig. 17 Correct answer rate by scale up systems.

されているかのように扱えるログファイルを生成した．なお，学習の処理時間は，表 2 の高性能な並列計算機で約 5 時間であった．測定結果を**図 17** に示す．

実験の結果，隠れマルコフモデル，ベイズ推定ともに，サーバ数の増加とともに正解率の低下がみられるが，それぞれ 63%，74%で収束傾向となった．本実験では，サーバの台数増加による構成の複雑性は増すものの，ストレージ・スイッチ間の構成は変わらない．構成が変化しないストレージ・スイッチ間に関しては，スケールアウトの実験結果より正解率の低下の影響は少ないと考えられる．そのため，正解率の低下は，サーバ・スイッチ間の構成が複雑化したことによる影響と推察する．

5. 考察

5.1 統計的推論の有効性

4 章により明らかになった隠れマルコフモデルとベイズ推定の特徴より，提案方式の有効性について述べる．各推論方式の特徴を**表 3** に示す．

隠れマルコフモデルは，統計元となるログファイルの種類，期間が十分に収集可能なインフラ機器において，ベイズ推定よりも高い正解率で推定できることが実験より判明した．また，隠れマルコフモデルは，ログファイルのみで推定することができるため，新製品などの未知のインフラ機器の構成に対しても，ログファイルが十分に収集できれば構成情報を推定できる特徴を持つ推論手法であるといえる．一方，ベイズ推定は，隠れマルコフモデルに比べ，ログファイルだけでなく CIM を使い推定することで，ログファイルが十分に収集できないインフラの構成においても，高い正解率で推定できることが判明した．さらに，CIM でモデル化されている構成であれば，インフラ構成がスケールアップした場合においても正解率の低下は少なく構成情報を推定できる特徴を持つ推論手法であるといえる．

表 3 各推論手法での特徴
Table 3 Characteristic of inference methods.

アプローチ / 特徴	隠れマルコフモデル	ベイズ推定
統計元のデータ	ログファイル	ログファイル，CIM データ
ログファイルの種類による正解率への影響	（1 種類）23%→（3 種類）60%	（1 種類）80%→（3 種類）85%
短期間（2 週間）のログファイルでの正解率	50%	73%
長期間（16 週間）のログファイルでの正解率	83%	80%
スケールアウトによる正解率の影響	0%	0%
スケールアップによる正解率の影響	83%→ 63%	80%→ 74%

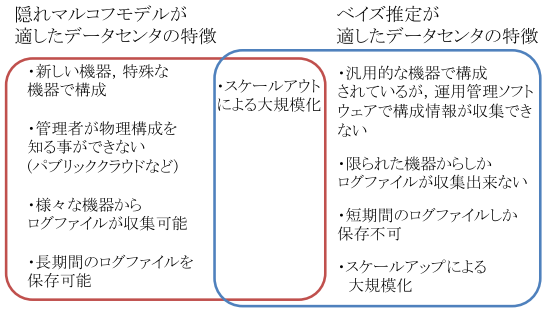

図 18 データセンタの特徴に合わせた推論方式
Fig. 18 Characteristic at data center and inference methods.

次に，データセンタの特徴の視点から各推論の特徴を**図 18** に示す．汎用的なインフラ機器で構成されているが，運用管理ソフトウェアがサポートするインタフェースの違いなどによりインフラ機器から構成情報を収集できない場合，さらには限られた機器・期間のログファイルしか収集できない場合には，ベイズ推定により構成情報が把握できる．一方，ベイズ推定で把握できない新しい機器や特殊な機器，パブリッククラウドなどのように管理者ですら詳細な構成情報を知りうる一般的な手段が提供されていない未知のインフラ構成に対し，多くのログファイルを収集さえできれば，隠れマルコフモデルにより構成を明らかにすることができるという特徴がある．

この両推論の特徴を活かした本提案方式は，ログファイルが十分に収集できていない状況でも一定の正解率であったベイズ推定で構成情報を推定した後，未知のインフラ機器などによって曖昧さのある構成を隠れマルコフモデルによって詳細化することで，より高い確率の推定結果を得ることができるといえる．

5.2 推論結果の正解率

4 章の結果から本提案方式は，最大 83%の確率で構成情報を推定できることが判明した．

大規模データセンタにおいて管理者が障害原因を解析するケースを考えた場合，障害原因と思われる候補を正解率の高い順に列挙し障害解析をナビゲートすることで，管理者の作業時間は大幅に短縮される．また，文献 [14] によ

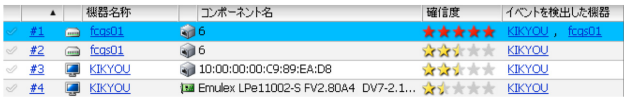

図 19 障害原因解析の表示画面
Fig. 19 GUI of root cause analysis.

ると，第一線で障害管理に取り組む熟練管理者は，過去に何度か実際に障害対応に取り組んだことで，障害対応スキルが向上した側面があるが，現在実際の障害を経験していない管理者が増えている．そのため，このような管理者のスキル向上のため，熟練管理者とともに行動させ経験させることが重要であると報告されている．しかし，熟練管理者が行動をともにできない場合においても，障害原因の候補を提示することによって，障害を経験しておらず独力で様々な要因を推察できない管理者に，複数の障害原因候補から検討を行うきっかけを与えることにつながる．この検討が経験の一部となり，管理者のスキルを向上させる可能性がある．

具体例を，従来研究の障害原因解析の結果表示画面（**図 19**）[13] を用いて説明する．この結果表示画面では，障害原因解析の実行で，あらかじめ IF-THEN ルールで用意している障害原因解析のルールと一致した数の累計を「確信度」として表示し，解析結果により導きだされた障害原因と推定されるコンポーネントの候補とあわせて表示している．この確信度は，値が高いほど事前に管理者が想定していた障害原因に近く，少ないほど遠いことを示している．つまり，実際の障害発生時では，ある 1 つの障害が引き金となり，複数の機器より障害通知が報告される．そのため，障害原因解析では，起点とする障害通知や受信した順序などによりあらかじめ用意しておいた IF-THEN ルールと一致する数が異なる．そこで，IF-THEN ルールとの一致する数が多いほど，管理者の想定に近い障害であると考え，確信度として表示している．

この障害原因解析の確信度と同様に，構成情報の確信度もあわせて表示することで，管理者の障害原因解析を支援することができる．構成情報の確信度は，4 章の正解率を確信度として考える．

本提案方式の構成情報を使った障害原因解析の結果表示では，注意しなければならないことがある．それは，一度に表示する障害原因と推定されるコンポーネントの候補数である．あまりにも大量の候補を一度に表示しては，管理者が候補の中から絞り込むのに時間を要してしまう．

そのため，正解率が収束する構成情報の候補数を考慮し表示するのが好ましい．4.3 節の実験結果より，隠れマルコフモデル，ベイズ推定ともに上位 3 位で正解率の収束傾向が見えると判断できる．つまり，構成情報の確信度は，正解率が収束する上位 3 位までを表示するのが効果的である．このように，統計的推論方式により推定した構成情報を正解率の高いものから表示することで，大規模，複雑化するデータセンタにおいて，管理者の障害原因解析を支援する．

6. おわりに

本論文では，インフラ機器が出力するログファイルを使った統計的推論方式で構成情報を推定する方式を提案した．近年，多くのデータセンタでは大規模・複雑化，さらには管理者数の減少と知識の属人化が進み，管理作業が間に合わずサービスを停止させてしまうリスクをかかえている．特に，障害発生時では，熟練管理者に頼った運用となってしまっていることが多くある．これに対し，本提案では，ベイズ推定と隠れマルコフモデルを使い，構成情報が収集できなかったインフラ機器の構成情報を推定することで，これまで適用範囲が限定されていた障害原因解析技術を広く適用できる見通しを得た．これにより，障害発生時における管理者の負荷を軽減できるだけでなく，短時間での解決が期待できる．

今後の課題は，障害原因解析技術で構成情報とならび重要な情報である解析ルールの自動生成を実現し，障害原因解析技術をさらに発展させ管理者を支援することである．これの実現に向け，生成規則を確率的に生成できる推論方式である PCFG（Probabilistic Context-Free Grammar）[15] などを用いて，管理者が予見し生成することができない解析ルールの自動生成を試みる所存である．

参考文献

[1] 宮下夏苗，上埜元嗣，宇多 仁，敷田幹文：大学におけるプライベートクラウド環境の構築と利用，第 3 回インターネットと運用技術シンポジウム，pp.17–24 (2010).
[2] 永井崇之，名倉正剛：迅速な危機回復を目的とする大規模環境向け障害原因解析システム，情報処理学会論文誌，Vol.54, No.3, pp.1109–1119 (2013).
[3] 工藤 裕，森村知弘，菅内公徳，増石哲也，鷹田憲久：障害原因解析のためのルール構築方法と解析実行方式，電気学会論文誌 C，Vol.132, No.10, pp.1689–1697 (2012).
[4] 敷田幹文：大規模サーバ間の部品依存関係に基づく障害通知方式の提案，情報処理学会論文誌，Vol.49. No.3, pp.1185–1193 (2008).
[5] Gupta, M. and Subramanian, M.: Preprocessor Algorithm for Network Management Codebook, *Proc. Workshop on Intrusion Detection and Network Monitoring*, pp.93–102 (1999).
[6] Wang, M., Holub, V., Parsons, T., et al.: Scalable Run-Time Correlation Engine for Monitoring in a Cloud Computing Environment, *Proc. 17th IEEE International Conference and Workshops on Engineering Component-Based Systems* (*ECBS 2010*), pp.29–38 (2010).
[7] DMTF: Systems Management Architecture for Server Hardware, available from ⟨http://www.dmtf.org/standards/smash/⟩.
[8] SNIA: Storage Management Initiative-Specification, available from ⟨http://www.snia.org/forums/smi/tech_programs/smi_home/⟩

[9] 本村陽一，岩崎弘利：ベイジアンネットワーク技術 ユーザ・顧客のモデル化と不確実性推論，東京電機大学出版局 (2006).
[10] 長尾 真，佐藤理史，黒橋貞夫，角田達彦：自然言語処理，岩波書店 (1996).
[11] 坂下幸徳, 東条 敏, 敷田幹文：ベイズ推論を用いたITシステム管理向け構成情報推定方式の提案，情報処理学会研究報告インターネットと運用技術，2013-IOT-20, No.32, pp.1–6 (2013).
[12] DMTF: Common Information Model, available from ⟨http://www.dmtf.org/standards/cim/⟩.
[13] 株式会社日立製作所：Hitachi IT Operations Analyzer, available from ⟨http://www.hitachi.co.jp/Prod/comp/soft1/itoperations/analyzer/⟩.
[14] 技術本部 ソフトウェア・エンジニアリング・センター：「障害管理の取組みに関する調査」報告書，独立行政法人情報処理推進機構 (2012).
[15] Jelinek, F.J.D. and Lafferty, R.L.M.: *Basic methods of probabilistic context free grammars*, Springer Berlin Heidelberg (1992).

坂下 幸徳 （正会員）

2003年北陸先端科学技術大学院大学情報科学研究科修士課程修了．同年株式会社日立製作所システム開発研究所（現，横浜研究所）入所．2012年北陸先端科学技術大学院大学情報科学研究科後期課程へ進学．2014年Hitachi America Ltd, R&D San Jose Innovative Solutions Laboratory．Lab Manager and Senior Researcher．ITシステム運用管理の研究開発に従事．SNIA Technical Council, SNIA日本支部技術委員会副委員長．

東条 敏 （正会員）

1981年東京大学工学部計数工学科卒業, 1983年同大学院工学系研究科修了．1995年同大学院博士（工学），1983〜1995年三菱総合研究所，1995年北陸先端科学技術大学院大学情報科学研究科助教授，2000年同教授．自然言語の形式意味論および人工知能の論理の研究に従事．人工知能学会，ソフトウェア科学会，言語処理学会，認知科学会各会員．

敷田 幹文 （正会員）

1965年生．1995年東京工業大学大学院理工学研究科情報工学専攻博士後期課程修了．博士（工学）．同年北陸先端科学技術大学院大学情報科学センター助手．2012年同大学情報社会基盤研究センター教授．大規模情報システム，グループウェアに関する研究に従事．ACM，電子情報通信学会，日本ソフトウェア科学会各会員．