博 士 論 文

大規模・複雑化するデータセンターにおける運用管理技術高度化に関する研究

主指導教員　敷田 幹文教授

北陸先端科学技術大学院大学
情報科学研究科情報科学専攻

坂下 幸徳

2015 年 3 月

要 旨

データセンターが大規模化し，世界中のデータ量が 2020 年には 2013 年の約 10 倍に到達する見込みである．また，データセンターのインフラストラクチャ機器に着目すると，サーバ，ネットワーク，ストレージと各レイヤで仮想化技術の導入が進み，構成が複雑化している．一方，このように大規模・複雑化しているデータセンターを運用する管理者の数は一定傾向にある．そのため，管理者一人当たりの負荷が増大しデータセンターの大規模・複雑化に追随できていない．さらに，管理者の体系も，管理作業の効率化を狙いサーバ，ネットワーク，ストレージとレイヤ毎に特化した管理者を揃え運用する水平分業管理が進んでいる．その結果，データセンター全体で一貫性のある運用が困難になっているだけでなく，経験の浅い管理者に知識の偏りが生じてしまい，障害発生時などでは，データセンター全体を把握している熟練管理者の経験と勘に頼らざるを得ない運用となっている．

本研究では，大規模・複雑化するデータセンターの運用に対し，データセンター全体の構成を一元管理する管理基盤の構築，特定管理者に依存しない運用の実現，未知な機器にも対応できる構成情報の推定方式，さらには管理者の知識向上支援を実現する．これに向け 3 つのステップにてアプローチする．まず，第一のステップでは，大規模・複雑化するデータセンターの全体構成を把握可能すべく管理ソフトウェアの管理基盤のスケールアップを行う．第二のステップでは，この管理基盤をベースに，各レイヤの管理者の専門的な知識をデータ化することで，専門知識を持たない他レイヤの管理者でも，各レイヤを縦断し一貫性をもった運用ができる運用管理技術を実現する．第三のステップでは，管理ソフトウェアが把握できない機器に対しても，これらの運用管理技術の適用させるために推論手法を使った構成情報の推定方式の確立と，管理者の知識向上の支援により進化するデータセンターに追随できる運用管理技術を実現する．本研究成果により，管理ソフトウェアと管理者の両面から運用管理技術を高度化させ，大規模・複雑化するデータセンターを熟練管理者に頼らず少人数の管理者でも破綻することなく運用できる目処をつける．

目 次

第 1章

序論

本章では，本研究の背景を述べた後，本論文の構成を述べる．

1.1 研究の背景

IT (Information Technology) システムの普及により，世界中で生み出されるデータ量が加速度的に増加し，2020 年には 2013 年の約 10 倍の 44 ZBytes に到達する見込みである．このような増加するデータを格納する先であるデータセンターが大規模化している [1]．特に，太平洋・アジア地区での大規模データセンターの成長率が高く年率 11.7%で増加している [2].

大規模なデータセンターが所有するサーバ，ネットワーク，ストレージなどのインフラストラクチャ機器に着目すると，仮想化技術の導入が進んでいる．仮装化技術はサーバ，ネットワークやストレージといった異なるレイヤのインフラストラクチャ機器に適用されている．このように各レイヤにて仮想化技術の適用が進んだことで，データセンターのインフラストラクチャ機器の構成が複雑化している．さらに，仮想化された機器は，物理的なハードウェアの変更に比べ，容易に構成が変更できるため，機器構成も頻繁に変更されるようになっている．そのため，これを運用する管理者は，インフラストラクチャの物理構成だけでなく仮想化された構成も理解し運用しなければならない．

一方，このように大規模・複雑化するデータセンターを運用する管理者の数は，一定傾向にある [3] ．そのため，管理者一人当たりの担当作業が増えたことで負荷

が増加し，管理者がデータセンターの大規模・複雑化に追随できていない．さらに，管理者の組織体系も変化している．小規模なデータセンターであれば，少数の管理者でサーバからストレージまで全てのインフラストラクチャ機器の構成を把握し運用している．しかし，インフラストラクチャ機器数の増加と仮想化技術によるインフラストラクチャ機器の複雑化により，各レイヤごとに特化した管理者チームで運用を行う水平分業管理化が進んでいる．そのため，各レイヤの管理者の連携は必要不可欠にも関わらず．各レイヤに管理者が分断され，データセンター全体で一貫性のある運用が難しくなっている．さらに，レイヤごとに管理者が分かれてしまったことで知識に偏りが生じ，全体を把握している管理者がごく一握りの経験豊富な熟練の管理者に偏ってきている．そのため，データセンター全体を通して判断する必要がある障害発生時などの場合，ごく一握りの熟練管理者の経験や勘に頼った運用となってしまっている．このため，熟練管理者が不在時や他の作業で対応できない場合，大規模データセンターの運用が破綻してしまうリスクを抱えている．データセンターの運用の破綻は，提供するサービスの質を低下させるだけでなく，最悪の事態ではサービス停止を引き起こし，これを利用する企業や大学などのビジネスに甚大な被害を与え，社会的な問題に発展するリスクを抱えている．本論文では，このように大規模・複雑化するデータセンターの運用を行う管理者の負荷を軽減し運用の破綻リスクを低減させるべく，運用管理技術の高度化を行う．

1.2 論文の構成

本論文では，大規模化・複雑化するデータセンターにおける運用管理技術高度化について，次に示す構成で述べる．

2 章 データセンターの実態と関連研究

本研究の対象とするデータセンターの実態と関連研究を，インフラストラクチャ機器の運用，管理者および管理者が利用する管理ソフトウェアのそれぞれの側面より述べる．

3 章 データセンターの運用の課題と研究アプローチ

データセンターの運用における課題を述べた後，本研究のアプローチについて述べる．

4章 管理ソフトウェアのスケールアップによる一元管理基盤

大規模・複雑化するデータセンターが備えるインフラストラクチャ機器に追随すべく，管理ソフトウェアが備え持つ機器の構成情報の収集処理のスケールアップ方式について述べる．

5章 知識データを使ったデータセンター全体視点の管理省力化

管理者が持つインフラストラクチャ機器の知識をデータ化し，これを活用することで，データセンター全体で一貫性を持った運用を実現するとともに，管理者の負荷を軽減する管理省力化方式について述べる．

6章 進化するデータセンターに追随可能な学習型管理技術

管理ソフトウェアが把握できないインフラストラクチャ機器の構成情報に対し，ログファイルを用いた学習する推論手法を用いることで，機器構成の推定を実現するとともに，管理者の学習支援を支援する．これにより，管理ソフトウェアと管理者の両面からの進化を目指す学習型管理方式について述べる．

7章 結論

全体のまとめと，今後の課題について述べる．

第 2章

データセンターの運用の実態と関連研究

本章では，本研究の対象であるデータセンターの運用の実態と関連研究について，インフラストラクチャ機器の運用，管理者および管理者が利用する管理ソフトウェアのそれぞれの側面より述べる.

2.1 データセンターが備えるインフラストラクチャ機器と運用

データセンターの運用の歴史は，1980 年代以前の数台のスーパコンピュータやメインフレームなどの大型計算機を運用する計算機室からスタートした．当時の大型計算機は，管理者の人件費に比べ機器のコストが高く，1 台の大型計算機を数人がかりで運用している状況であった．その後，1990 年代には PC(Personal Computer) の普及により，大学などの演習室に見られる計算機室が登場してきた．しかし，徐々に管理対象である機器の数が増えてきたこと，機器のコストより運用する管理者の人件費が逆転し始めたこと，さらにネットワークが普及したこと，などの要因により，1990 年代後半には複数の機器を一箇所にあつめ集中管理を行うデータセンターへと進化した．さらには，このデータセンターもクラウドコンピューティングの概念のもと，2006 年頃よりパブリッククラウド，プライベートクラウドの

形態に進化している [4][5][6] .

パブリッククラウドは，Amazon や Google に代表されるようなクラウドサービスプロバイダー (CSP: Cloud Service Provider) がデータセンターを所有し，利用者はインターネットを通じサーバやストレージなどのインフラストラクチャ機器からメールサービスや Web サーバなど各種サービスを提供する形態である [7][8] . パブリッククラウドの場合，データセンターの管理者は，CSP が抱える管理者によって運用されている．また，パブリッククラウドのデータセンターは，利用者数や CSP のビジネス成長にあわせデータセンターが備えるインフラストラクチャ機器を増強し大規模化している．さらには，利用者数の拡大にあわせ均一のインフラストラクチャ機器を大量に揃えることで，画一的な運用をすることで運用のコストを削減する試みも行われている．代表的な研究として均一のインフラストラクチャ機器による管理者を介せずにスケーラビリティを高める研究 [9][10][11] [12][13] や消費電力などコスト削減に向けた研究がある [14][15].

プライベートクラウドは，企業や大学などが所有するインフラストラクチャ機器で構成されたデータセンターであり，企業や大学の特色にあわせたサービスを提供する形態である．プライベートクラウドでは，管理者は各企業や大学が抱える職員である．また，プライベートクラウドでは，CSP のように利用者数の拡大にあわせて大規模化するだけでなく，特に大手企業などに見られる企業買収などにより，買収元の企業が所有していたデータセンターと買収先の企業が備えていたデータセンターを統合したり，大学では異なる部局がもっていたインフラストラクチャ機器を統合したりと，組織の統合によって大規模化するケースがある．そのため，プライベートクラウドでは，異なるベンダや様々な特色をもったインフラストラクチャ機器で構成されるヘテロジニアスな環境となっている．さらには，運用方法が異なる管理者が混在しているデータセンターもある．プライベートクラウドに関する研究としては，運用の効率化に着目した研究だけでなく，パブリッククラウドには格納できない企業や大学などの運営に関わる重要なデータが多いことからセキュリティの向上に向けた研究 [16][17][18][19][20][21][22] も行われている．

別の観点として，データセンターが所有するサーバ，ネットワーク，ストレージなどのインフラストラクチャ機器に着目すると，仮想化技術の導入が進んでいる [23][24][25][26]．仮想化技術は，1998 年に x86 系サーバ向けにサーバ仮想化技術

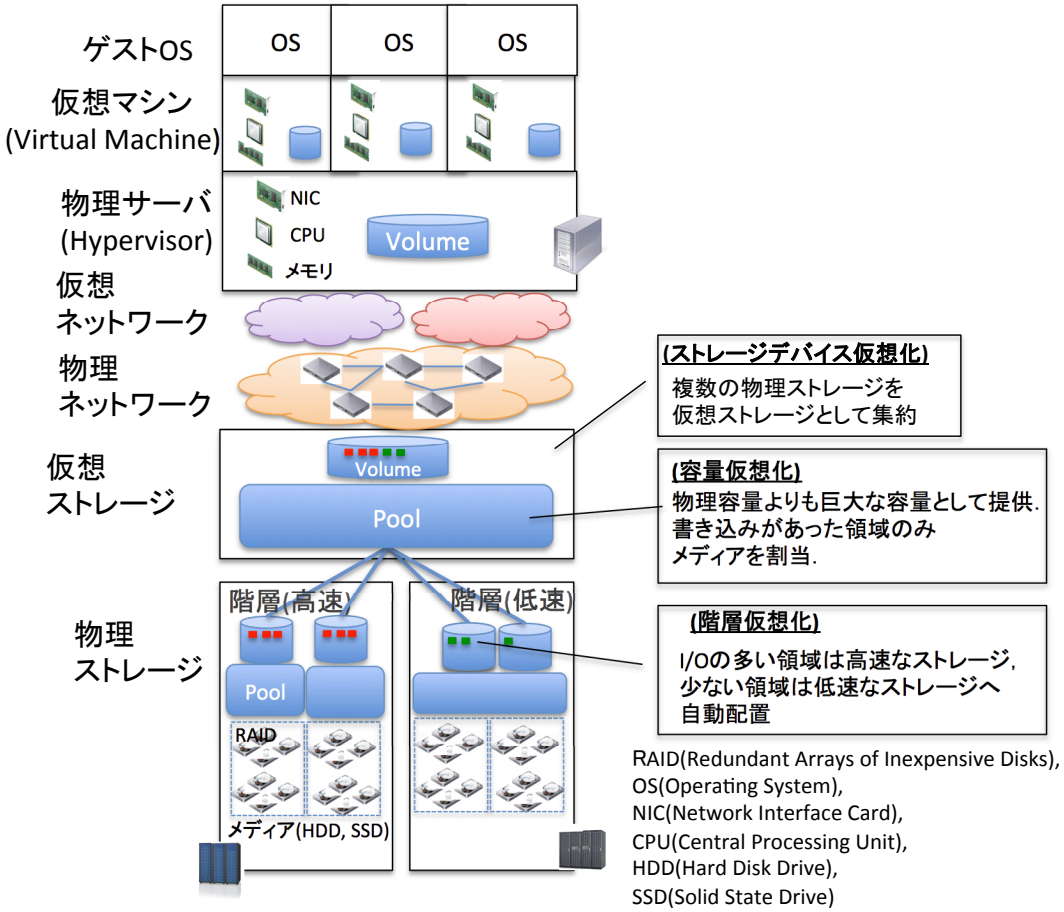

図 2.1: 仮想化技術による複雑化するインフラストラクチャ機器の例

が登場して以降，2012 年には物理サーバの導入台数を仮想サーバの導入台数が上回っており，今後も仮想サーバの導入が増加する見込みである [27] ．また，このサーバ仮想化にあわせ物理ネットワークを変更することなくネットワーク構成を自由に構成すべくネットワークの仮想化も進んでいる [28]．さらに，ストレージにおいては，複数台のストレージをあたかも 1 台のストレージに見せることで巨大なストレージを提供できるストレージデバイスの仮想化，サーバへ仮想的な巨大な容量を提供しつつも，実際にはサーバが書き込みを行った領域のみに HDD(Hard Disk Drive) などのメディアを割り当てることで，メディアのコストを削減するストレージ容量の仮想化，さらには，性能やメディアコストの異なる複数種類のメディアを階層化し，データの書き込み頻度に応じて階層化間を移動させ，性能とコストを最適化するストレージ階層の仮想化技術も登場している [29][30][31][32] ．

図 2.1 に仮想化技術により複雑化しているインフラストラクチャ機器の一例を示す．このようにサーバ，ネットワーク，ストレージの各レイヤで仮想化技術が登場しデータセンターへ導入が進んでいる [33][34][35].

2.2 管理者

次に，データセンターの運用を行う管理者について述べる．データセンターの運用を行う管理者の体系としては，図 2.2 (a) に示すサーバ・ネットワーク，ストレージの全レイヤのインフラストラクチャ機器を垂直的に運用する垂直統合管理と，図 2.2 (b) に示す各レイヤごとに管理者チームを作り運用している水平分業管理のいずれかを取っている．特に，100 ラック以上を備える多くの大規模データセンターでは，水平分業管理で運用している．この水平分業管理では各レイヤ内で同じ設定や構成を行う運用が多々あり，効率的に作業を行うためには各レイヤの機器に特化した専門的な知識が必要となっているためである．

各レイヤで行われている運用の一例として，ストレージのバックアップの設定がある．バックアップの設定では，データセンター内で定めたバックアップ間隔，時間，バックアップ方式に従いストレージの設定を行う．また，データ容量が大きくなればなるほど，これに比例しバックアップに時間がかかる．さらには，バックアップ先となるストレージは，サーバが直接アクセスするストレージに比べ数が少ないため，データ容量やストレージの性能などの情報からバックアップの完了時間を見積もった後，ストレージごとに時間をずらしバックアップの設定を実施する．このようにバックアップの運用においては，データセンターで備えるストレージを横断的に確認し設定する必要がある．バックアップの運用では，各ベンダの製品や構成により異なるストレージの性能やバックアップ方式など，これら複数ベンダのストレージに関する専門知識が管理者に必要となる．

特に，数多くのインフラストラクチャ機器を抱えるプライベートクラウドのデータセンターでは，単一の製品で構成されるケースは稀であり，複数ベンダの製品が混在するヘテロジニアスな環境となっている．そのため，異なるベンダの製品を大量に備えるデータセンターほど，管理者は製品の違いを意識し運用することに時間を費やしている．

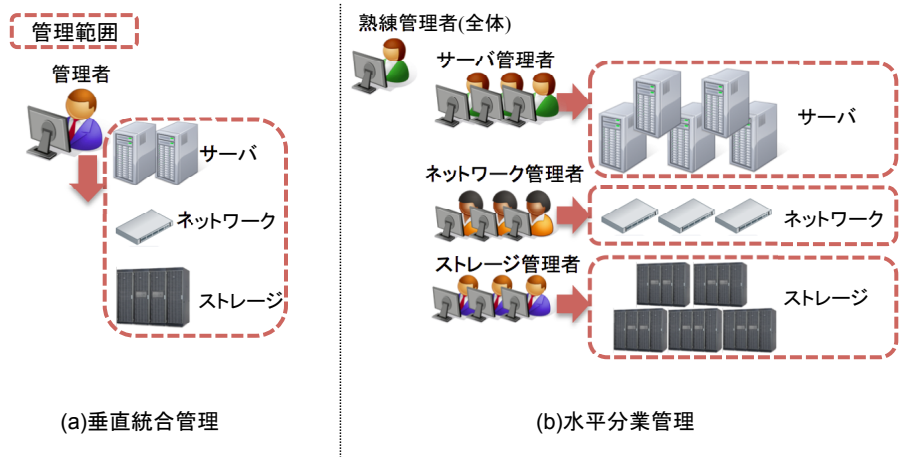

図 2.2: データセンターの管理者の担当範囲

別の観点として，管理者の知識レベルに着目する．データセンターは 2.1 節に示すように小規模なデータセンターから進化していることが多くあり，従来小規模なデータセンターを垂直統合管理の体系で運用していた管理者が，大規模データセンターの運用を担っているケースが多々ある．一例をあげると，データセンターの管理者として 1990 年代に就職し，ある組織の小規模なデータセンターの立ち上げから参画，その後，データセンターの大規模・複雑化とともに様々な機器を担当し，40 歳代後半となった今も第一線で運用を続けている．このような管理者は，長く運用しているため豊富な経験値をもっているだけでなく，過去にサーバ，ネットワーク，ストレージと様々なインフラストラクチャ機器を垂直統合管理していたことで，各装置の基礎的な知識を十分備えている．本論文では，このような管理者を熟練管理者として略す．このような熟練管理者は，元々小規模なデータセンターの運営であったことや，年齢を重ねたことで退職やマネージャー職となる事例が多いことから人数が減少傾向にある．一方，近年，採用された多くの管理者は，始めから大規模・複雑化しているデータセンターの運用を担当するため，自らが担当するインフラストラクチャ機器の専門知識に特化する傾向が強い．このように大規模・複雑化しているデータセンターは，経験豊富かつ様々なレイヤのインフラストラクチャ機器の知識を有するごく少数の熟練管理者と，特定レイヤの専門知識を有する多数の管理者によって運営されている．熟練管理者と各レイヤの管理者の役割と知識レベルについて文献 [36] を元に，図 2.3 にまとめる．

熟練管理者

[知識レベル]インフラストラクチャ機器全般の幅広い知識
[人数] 少ない(大規模データセンターでは1-10人規模)
[運用における主な役割(データセンター全体)]

通常運用時	・データセンター全体の運用状況の確認，対応 ・定期報告
障害発生時	・障害記録の内容（識別、分類等）の判断 ・緊急度・優先度の決定・障害対応の指揮 （対応方針の決定、関係部門との調整、エスカレーション） ・障害対応のクローズ

管理者(各レイヤー)

[知識レベル] 担当機器の深い専門知識
[人数] 多い (大規模データセンターでは各レイヤー 50-100人規模)
[運用における主な役割(担当機器のみ)]

通常運用時	・キャパシィ管理 ・イベント管理 ・モニタリング
障害発生時	・障害記録の入力 ・初期診断 ・調査と診断 ・解決と復旧 ・エスカレーション ・ワークアラウンド

図 2.3: 熟練管理者と管理者 (各レイヤ)

2.3 管理ソフトウェア

次に，管理ソフトウェアについて述べる．データセンターの管理者は，データセンターのインフラストラクチャ機器を運用する際，管理ソフトウェアを利用する．特に，仮想化環境においては，物理的な設置・変更を必要とせず仮装化機能により仮想化されたリソースを設定・変更する．そのため，仮装化環境が浸透しているデータセンターにおいては，管理ソフトウェアの良し悪しが管理者の負荷を大きく左右する．このような管理ソフトェアを対象とし，管理者の負荷を軽減する研究がある．

2003 年に Jefferey らにより，自律コンピューティングの概念が提唱され，そのなかでインフラストラクチャ機器の運用を省力化する管理モデルが提唱された [37][38][39][40]．図 2.4 に，自律コンピューティングの概念における管理モデルを図示する．この管理モデルでは，監視 (Monitor)，分析 (Analyze)，計画 (Plan)，実行 (Execute) の 4 つの管理フェーズを循環させることで管理者の作業を自律化し，自律コンピューティングを実現する管理モデルが提唱されている．この自律

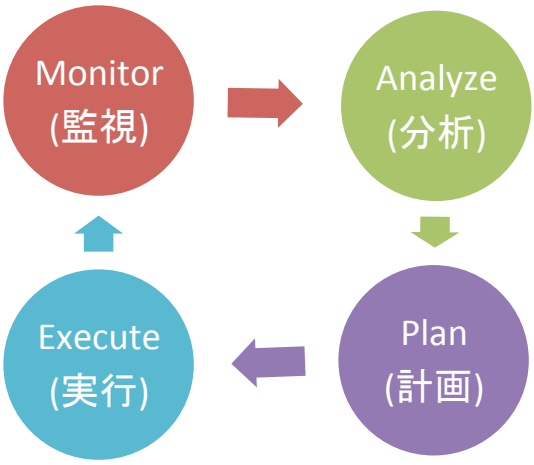

図 2.4: 自律コンピューティングの管理モデル

コンピューティングの概念の実現に向け，管理ソフトウェアを対象に研究が進められている．

監視フェーズの研究では，敷田らによるインフラストラクチャ機器の監視を支援する研究 [41][42][43][44][45][46] がある．これらの研究では，インフラストラクチャ機器が備える設定ファイルを読み取り，接続関係のあるサーバやストレージなどの機器の構成情報を抽出することで，障害やイベントが発生した際の影響範囲の把握を支援する [47][48]．たとえば，ファイルストレージに障害が発生した際，いずれの Web サーバに影響するかを管理ソフトウェアが特定することで，影響範囲を把握する作業を支援する．さらには，藤澤らにより障害の影響範囲から，障害通知先の管理者を特定する研究が行われている [49][50]．これらの研究は，インフラストラクチャ機器の接続の依存関係をベースに構成情報を監視することで，管理者の負荷を軽減している．このような研究を支える IT 業界の動向としては，複数の IT ベンダが協力しシステム管理の標準化を目指す団体として 1992 年に DMTF(Distributed Management Task Force) が設立された [51] ．DMTF では，インフラストラクチャ機器の構成を標準モデルとして規定する CIM(Common Information Model) や，管理ソフトウェアと各種インフラストラクチャ機器間の管理情報をやり取りするための通信プロトコルである WBEM(Web-Based Enterprise Management) などを標準仕様として仕様策定している [52][53]．このような取り組みにより，複数ベンダのインフラストラクチャ機器が混在するようなヘテロジニアスな環境においても，

データセンターの一元的な監視ができるようになりつつある．

分析フェーズの研究では，工藤らを始めとする障害通知を契機とした障害原因解析に関する研究 [54][55][56][57][58][59][60][61] がある．この障害原因解析の研究では，あらかじめ障害通知とこれに関連する障害原因を IF-THEN ルールの形式で予め管理ソフトウェアにて定義しておく．この IF-THEN ルールは，監視対象となるインフラストラクチャ機器の構成情報をベースに定義される．次に，障害発生時にインフラストラクチャ機器から送信される SNMP(Simple Network Management Protocol) に代表される障害通知を監視しておき，管理ソフトウェアが SNMP などを受信すると，予め定められた IF-THEN ルールに従い障害原因を分析する．この研究では，管理対象となるインフラストラクチャ機器の構成が把握できており，かつ IF-THEN ルールが記述可能な障害の場合に有効である．

次に，計画フェーズの研究では，野口らを始めとするデータ再配置による動的負荷分散機能の設計に関する研究 [62][63] がある．これらの研究ではクラスタ環境のサーバや HPC(High Performance Computing) の各計算ノードの稼働情報を基にデータの再配置を行うことで負荷分散を実現している．これに類する研究としては，仮装サーバを対象に VM(Virtual Machine) を移動させることで複数 VM で構成されるシステム性能を最適化する研究 [64][65][66] や，仮装ストレージを対象に，HDD メディアの性能差を考慮したデータ配置計画の研究 [67] がある．

実行フェーズの研究では，Kephart らを始めとするイベント駆動型の自動実行に関する研究 [68][69][70][71] がある．これらの研究では，インフラストラクチャ機器への複数設定操作をポリシーとして予め定義し，指定された時間や障害などのイベントの受信時に自動実行を実現している．たとえば，5 分間に 25 回以上ログインの失敗というイベントが発生した際，ユーザのアカウント情報を Disable に変更の制御を行う．これに類する研究として予め定めたポリシーの実行スケジューリングや実行タイミングをインフラストラクチャ機器の変更にあわせ変更する研究 [72][73][74] がある．

第 3章

データセンター運用の課題と研究アプローチ

本章では，大規模化・複雑化するデータセンターの運用における課題を述べた後，本研究のアプローチについて述べる．

3.1　データセンター運用の課題

IT システムが社会に深く浸透したことにより，データセンターが提供するサービスは，無停止かつ安定稼働されることが要求されるようになった．サービスの停止は，組織の活動が停止に結びつき社会的な問題に発展し，新聞やニュースなどのメディアで取り上げられることもあるためである [75][76]．2012 年に起こったデータセンターの障害の事例では，大手企業を含む 5698 件の Web サイト，メール，顧客情報，スケジュールなど様々なデータが失われる事態を引き起こし，ビジネスに大きな影響を与える結果となった [77][78] .

データセンターでは，無停止・安定稼働は当たり前のものとして考えられた上で，企業などが所有するプライベートクラウドでは，IT システムの取りまとめ責任者である CIO(Chief Information Officer) らから，パブリッククラウドと同じレベルもしくはそれ以下の低コストで且つ，パブリッククラウドにはない自らの組織の事業に貢献できる魅力あるサービスの提供が要望されている．しかし，CIO の期待に対し，データセンターを運用する現場が，この要望に即座に答えること

は容易ではない．それは，主にデータセンターのコストに関する問題と，これを運用する管理者に関する問題に起因する．

まずコストに関する問題について述べる．プライベートクラウドのデータセンターは，各地に分散していたインフラストラクチャ機器を，集中化し大規模なデータセンタを構築すると共に，インフラストラクチャ機器の設備コストの削減を狙い，仮装化技術を導入することで，物理的なサーバやネットワーク，ストレージなどのハードウェア機器の集約を行ってきた．これにより，全て物理的なハードウェアで構築するよりも，機器の設備コストを抑えつつ，より多くのサービスを迅速に提供することが可能となった．また，仮装化技術の導入により物理的な機器の数が減り，電力や設置スペースなどのコストも削減効果が見込める．しかし，運用コストにおいては，仮装化技術を導入したことで，逆に上がってしまうことがある．例えば，データセンターのインフラストラクチャ機器の構成を設計・設定する際には，従来の物理的な機器の設計・設定だけでなく，仮装化された機器も併せて設計・設定が必要となり，管理者の多くの時間と労力が必要となる．また，障害が発生した場合においては，サーバ，ネットワーク，ストレージの物理的な機器だけでなく，仮装化された機器，さらには両方の組み合わせまで，手を広げ障害原因を調査し対策しなければならない．しかも，サービス停止が許されないデータセンターの運用の現場では，これらが管理者に大きな負担となり，運用コストの削減が困難になっている．このように，データセンター全体でみると，インフラストラクチャ機器にかかる設備コストと運用コストを両立したコスト削減は難しい．

次に，管理者に関する問題について述べる．データセンターが提供するサービスに対し，無停止かつ安定稼働を求められているなか，運用上の課題として図 3.1 に示す報告がある [3] ．上位 4 位までが管理者に関する課題である．なお，図 3.1 の「スタッフ」「担当者」はデータセンターの運用を行う管理者と考え，本論文は以降,「スタッフ」「担当者」を「管理者」と読み替える．

データセンターでは 2.2 節で述べたように，インフラストラクチャ機器の複雑化に対応すべく，管理者が自身の担当する機器に特化した専門知識を持つようになった．これにより，特定の機器の特性を生かした設定が可能となった，しかし，反面，専門知識の習得と特定機器のみの運用に時間を取られ，データセンター全体

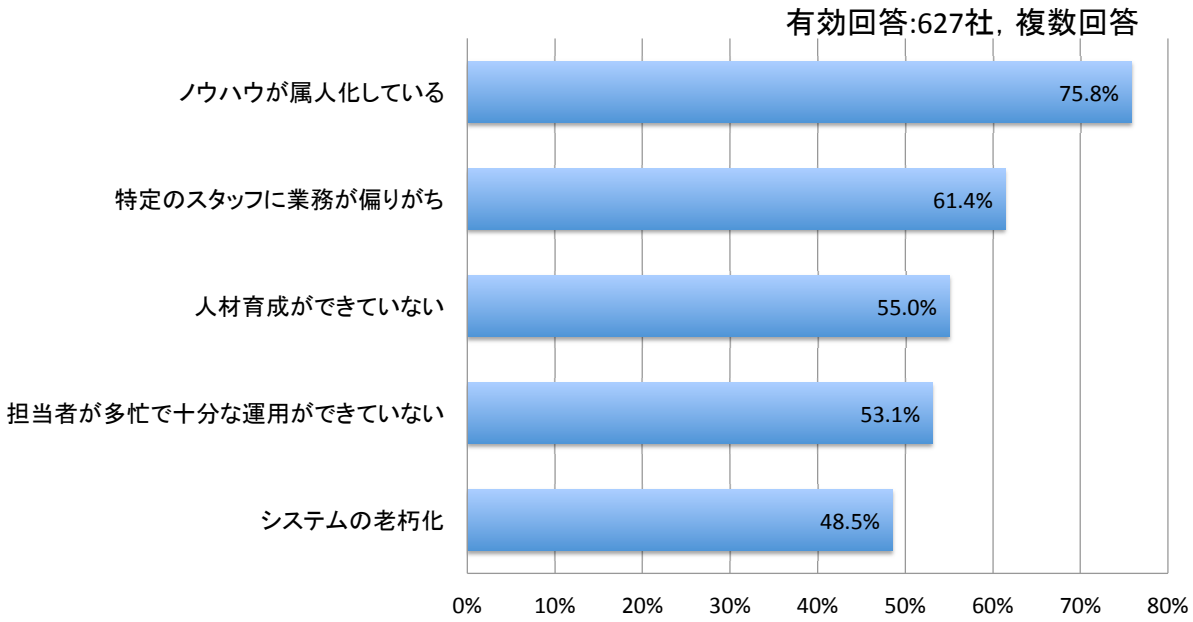

図 3.1: 運用上の課題 上位 5 項目 (文献 [3] より一部抜粋)

視点で設計し，インフラストラクチャ機器を設定できるスキルをもった熟練管理者への育成が難しくなっている．これが原因で図 3.1 の課題の 1,2 位の「ノウハウが属人化している」「特定の管理者に業務が偏りがち」となっていると分析する．さらには，ある機器に特化し設計・設定することは，部分的に最適化されるものの，全体としてはアンバランスな設計・設定となることがあり，障害発生のリスクを高める要因の一つとなっている．

「人材育成ができていない」「管理者が多忙で十分な運用ができていない」問題に関しては，日々の管理者間のコミュニケーションに起因すると考える．データセンターの運用にとって，管理者間のコミュニケーションは重要なスキルとなる．特に障害管理が発生した場合，運用の責任者 (管理者のリーダ) が障害管理に関わる経営層，情報システム企画部門，開発者，利用ユーザに対し，コミュニケーションのハブとなり，障害の影響，復旧までの見込み時間，対策手順などをまとめ迅速に伝えることで，ビジネスへの影響を軽減させる役割を担う．さらに，障害を未然に防ぐためには，日頃より管理者間の自発的かつ活発なコミュニケーションによりコミュニケーションにより，お互いが担当しているインフラストラクチャ機器の情報を共有することによって，事前に障害が発生しやすい問題点を明らかに

し，対策を検討・実施することが人材育成においても重要である．このようなコミュニケーションは，10 名以下の少人数で運営される小規模なデータセンターでは実施できるかもしれない．しかし，100 人を超えるような管理者を抱える大規模なデータセンターにおいては難しい．管理者が自身の担当する機器の設定変更が，誰の担当する機器に影響を及ぼすの関係性がわかりにくく，自発的なコミュニケーションが不足がちとなるだけでなく形式的な報告となってしまい，問題を早期に発見することが難しい．特に，大規模なデータセンターでは，管理者一人あたりが管理しなければならない機器も多いことから，日々の作業におわれ，自発的なコミュニケーションに時間を十分に割けていないこともコミュニケーションが不足している要因の一つである．また，大規模な障害などを経験したことのない管理者も数多く存在しており，管理者間の連携の必要性を身をもって感じることができず，疎なコミュニケーションとなっている運用現場もある．

このようにデータセンターを運用する管理者が抱える問題により，無停止かつ安定稼働したサービスを運用することは容易ではない．

3.2 研究アプローチ

2 章にて，大規模・複雑化するデータセンターと，これを運用する管理者，管理者が利用する管理ソフトウェアについて紹介した．また，2 章で紹介した関連研究は，パブリッククラウドのような均一のインフラストラクチャ機器で構成されたデータセンターや，ある特定のインフラストラクチャ機器には有効である．しかし，プライベートクラウドのようなヘテロジニアスの環境の大規模・複雑化するデータセンターには対応できない．さらに，インフラストラクチャ機器に関連する課題だけでなく，3.1 節で述べた運用上の課題の上位 4 つは管理者に関する課題である．

そこで本研究では，大規模・複雑化するプライベートクラウドのデータセンターを運用するために，分断されたレイヤを縦断しデータセンター全体で一貫性を持った運用と，少人数かつ熟練管理者に依存しない運用の両方を同時に可能とする運用管理技術の高度化を実現する．

これを実現するために，次の 3 つのステップにて取り組む．まず，初めのステッ

プでは，データセンター全体の一元管理に向け，管理ソフトウェアの管理基盤である構成情報収集のスケールアップを行う．次のステップでは，この一元管理された管理基盤を使い，管理者の専門的な知識をデータ化することで，専門知識を持たない他レイヤの運用を担当する管理者でも，インフラストラクチャ機器の各レイヤを縦断し，データセンター全体で一貫性を持った運用ができる運用管理技術を実現する．最後のステップでは，管理ソフトウェアが把握できないインフラストラクチャ機器に対しても，これらの運用管理技術の適用を可能とする構成情報の推論手法を確立し，さらに，経験の浅い管理者の学習支援を実現することで熟練管理者に依存しない運用を実現する．この 3 つの研究について説明する．

(1) 管理ソフトウェアのスケールアップによる一元管理基盤の構築

大規模化するデータセンターを一元管理するためには，管理ソフトウェアが備えるインフラストラクチャ機器の構成情報に対し大規模対応が必要である．従来までは，管理ソフトウェアがインフラストラクチャ機器から構成情報を収集する際，収集処理が長時間化し実運用に耐えられなかった．そこで，本研究では，収集処理を一括取得から分割取得に変更することでメモリ使用量を削減し，削減したメモリを使い収集処理を並列化することで，収集処理を高速化する．

(2) 管理者知識のデータ化によるレイヤを縦断する運用管理技術の実現

大規模データセンターでは，インフラストラクチャ機器のレイヤごとに専門の管理者が存在し運用を行なっている．このようななか，利用者からはサーバやストレージなどのレイヤごとに分かれたサービスの提供ではなく，サーバ・ストレージを一体で提供するような，システムとしてのサービス提供が要望されるように変化してきている．この利用者の要望に応えるためには，データセンター全体で一貫性を持ち運用することが必要である．そこで，本研究では各レイヤの管理者が持つ専門知識のデータ化と (1) の構成情報と組み合わせることで，専門知識を有さない管理者でも，レイヤを横断し一貫性をもってリソースの適正配置が行える運用管理技術を実現する．

(3) 推論手法による構成機器把握と管理者の学習支援による運用管理技術の確立

(1)(2) の研究にて大規模なデータセンターを少人数で管理する管理基盤の

確立と管理高度化を実現する．しかし，データセンターは仮想化技術の登場により物理的なインフラストラクチャ機器よりも容易に構成変更が行えるようになったことで，短期間での構成変更や未知の機器の導入が加速しただけでなく，構成の複雑化が進んでいる．そのため，管理ソフトウェアがサポートできず把握できない環境が登場してきている．特に，データセンター全体を見渡し原因調査が必要となる障害発生時などでは，サポートできない機器があると管理ソフトウェアでの支援が不十分となり，熟練管理者の経験に頼り運用せざるを得ないケースが多々ある．そこで，本研究では，このような管理ソフトウェアが把握できない環境でも構成情報を推定できる管理技術を実現するとともに，これを利用する管理者の学習を支援する管理技術を実現する．構成情報の推定に向けては，管理対象となるインフラストラクチャ機器が出力するログファイルを使い，統計的推論手法であるベイズ推定と隠れマルコフモデルにて，構成情報を取得できない環境の構成情報を推定できる方式を確立する．さらに，推定した構成情報を確信度の高い順に絞り込み表示することで，熟練管理者に頼らずとも運用できる管理者の知識の向上を支援する．

第 4章

管理ソフトウェアのスケールアップによる一元管理基盤

本章では，大規模・複雑化するデータセンターが備えるインフラストラクチャ機器の構成を一元管理すべく，管理ソフトウェアが備え持つ機器の構成情報の収集処理をスケー ルアップする方式について述べる．

4.1 緒言

2.1 節で示したように，IT システムの普及によりデータセンターの規模が大規模化している．これの要因は主に 2 つある．1 つ目は，利用されているサーバやストレージの仮想化技術の進歩である．仮想化技術により，CPU やメモリといったサーバリソースや，ストレージのボリューム などのストレージリソースが仮想化されるようになり，物理的リソースの消費を抑え低コストで大量リソースを提供できるようになったため，データセンター内のリソース数が年々増加している．2 つ目は，データセンターへのデータ集中である．従来，サーバ内に保存されていたデータが，ネットワークの普及により，データセンターのストレージに保存されることが一般的になった．そのため，データセンターに保存されるデータ量が増加している．これに伴い，データセンターが所有するストレージの台数が増加し，世界中での全出荷台数が年率約 4 倍のペースで増加し続け，約 5 年後には 100 台以上ものストレージを所有するデータセンターの登場が予測されている [79]．このよ

うな状況のなか，大規模化するデータセンターの運用において，リソースの利用効率向上や障害発生時の迅速な対応を実現するためには，仮想サーバからストレージまでの接続関係や設定内容を示す構成情報の一元管理が不可欠である．しかし，データセンターの運用に利用していた従来の管理ソフトウェアでは，管理対象機器から構成情報を収集する際のスケーラビリティが低く，大規模化するデータセンターを一元管理できなかった．そこで本章では，大規模データセンターに対応すべく管理ソフトウェアのスケーラビリティを向上させる構成情報の高速収集方式を提案する．以下，4.2 節では，従来の管理ソフトウェアについて述べ，4.3 節では本章で対象とする問題について述べる．次に，4.4 節で問題を解決する方式を提案し，4.5 節では提案方式の試作システムと，それを用いた測定結果を述べる．4.6 節では，測定結果に基づいた提案方式の有効性を議論する．

4.2 従来の管理ソフトウェア

サーバやストレージを一元管理すべく管理対象機器と管理ソフトウェア間の管理インターフェースに関する研究 [80][81] が行われている．また，サーバからストレージまでを一元管理する商用の管理ソフトウェア [82][83][84] も登場している．これらの管理ソフトウェアは，まず，管理対象のインフラストラクチャ機器が備える管理用インターフェースを通じ，機器と通信を行うことでインフラストラクチャ機器の構成情報を収集し運用を行っている．管理対象であるインフラストラクチャ機器と構成情報収集モジュール間の管理用インターフェースとして，従来は，SNMP(Simple Network Management Protocol) に代表されるように通信データ量の少ないシンプルな形式のものが利用されていた．しかし，SNMP は障害情報の通知には利用されているものの，構成情報の収集では，XML (Extensible Markup Language) 形式のデータフォーマットの利用が一般化している．これは，サーバやストレージが備え持つ各リソース間の関係が複雑化したためである．従来は，サーバやストレージの構成がシンプルであったため，リソース間の関連を SNMP のようなツリー形式のモデルで表現できていた．しかし，リソース間の関係が複雑化したことで，ツリー形式のモデルでは，表現できなくなった．そこで，複雑化したリソース間の関係を適切に表現すべく CIM(Common Information Model)[52] に

表 4.1: 管理インターフェースとデータフォーマット

管理対象	管理インターフェース	モデル	データフォーマット
仮想サーバ (VMware)	SOAP	独自モデル	XML 形式
仮想サーバ (VMware)	SMI-S	標準モデル (CIM)	XML 形式
仮想サーバ (Hyper-V)	WMI	標準モデル (CIM)	XML 形式
FC スイッチ	SMI-S	標準モデル (CIM)	XML 形式
ストレージ	SMI-S	標準モデル (CIM)	XML 形式

代表されるようなオブジェクト指向形式のモデルの利用が進んでいる．このオブジェクト指向形式のモデルを通信データ上で表現するのに，表現の自由度の高さが必要となり，XML 形式のデータフォーマットが利用されている．表 4.1 に代表的な管理対象の管理インターフェースとデータフォーマットを示す．

管理インターフェースとしては，大手ストレージベンダの 90%以上の製品に採用されている国際標準仕様 SMI-S(Storage Management Initiative-Specification)[85]，Web Service などで利用されている SOAP(Simple Object Access Protocol)，Windows が OS 標準で備えている管理インターフェースの WMI(Windows Management Instrumentation) が普及している．

4.3 大規模環境の一元管理における問題点

本節では，データセンターの大規模化に伴い発生する従来の構成情報収集方式の問題点について述べる．

4.3.1 情報取得時におけるメモリ使用量の増大

1 つ目の問題点として，管理対象のインフラストラクチャ機器から構成情報を取得する際のメモリ使用量の増大があげられる．その原因は，各種機器が備えるリソース数の増加である．具体例をあげ説明する．エンタープライズクラスのストレージでは，1 台のストレージが備えるボリューム数が増加している．最大構成で 128,000 ボリュームを備えるストレージも登場し，メインフレームなどの大型

計算機を利用する環境では，最大構成にて利用するケースもある．そのため，構成情報を収集する場合において，ボリュームなど数が多いリソースの構成情報を収集する場合にメモリ量が問題となる．構成情報を収集する際 SMI-S では，指定されたリソースの全情報を 1 個の XML データにエンコードする．そのため，1 回の通信で送信されるデータ量が巨大化するだけでなく，その XML データの生成処理，解析処理に膨大なメモリが必要となる．例えば，1 個のボリューム の構成情報を表現する XML のデータ量は約 11KBytes である．1 台のストレージに 10,000 ボリュームが存在する場合では，約 100MBytes ものデータ量となる．これを，32bit 環境の Java5 の標準の XML パーサを利用した場合のメモリ使用量を測定すると，XML データの読み込み時に，メモリ上で Java のオブジェクト形式に変換するため，XML データの読み込み処理だけで，約 500MBytes 必要となる．128,000 ボリュームを持つストレージの場合では，約 1GBytes ものデータ量となり，これのデータ処理には 5GBytes 以上ものメモリが必要となる．これは，異なる OS 上で実行することの多い管理ソフトウェアにて採用実績の多い 32bit 環境の Java VM のメモリのヒープサイズが理論値で最大 2GBytes であることを考慮すると，同環境では，このような大規模なエンタープライズクラスのストレージ 1 台すら運用できない．メモリ量に関しては，管理ソフトウェアを導入する環境を 64bit 環境で且つ大量のメモリを搭載した管理用 PC に導入するなど，ハードウェア側からのアプローチにより改善する方法もある．しかし，データセンターの大規模化が加速度的に進んでいる状況や，複数ストレージやサーバをあたかも 1 台のハードウェアに見せる仮想化技術の発展を考慮すると，使用するメモリ量も加速度的に進むと予測される．例えば，100 台のストレージを仮想化技術により 1 台の仮想的なストレージとして扱うケースを想定すると，500GByte 以上ものメモリが必要となり，エンタープライズクラスのサーバが必要になる．しかし，業務システムのサービスを提供するサーバとは異なり，データセンターの運用管理を支援する管理ソフトウェアは，運用管理に掛かるコストを削減することが目的のため，コストの観点から，オフィス業務などで利用している一般的な PC など安価なハードウェアにインストールされ利用されることが多い．そのため，ハードウェア側からのアプローチだけでは限界があり，ソフトウェア側での使用メモリ量の削減が必要である．

4.3.2 情報収集時間の増大

次に，2 つ目の問題点として管理対象のインフラストラクチャ機器から構成情報を収集する際の時間の増大があげられる．その原因は，データセンター内のサーバやストレージなどの台数増加である．管理ソフトウェアは，リソースの利用効率向上や障害発生時の迅速な対応を実現するために，複数ストレージの構成情報を一元管理しなければならない．また，単に一元管理するだけでなく，適切な運用を行うためには複数の機器間で構成情報の整合性を保つことが不可欠であり，各種機器間の構成情報の収集時の時刻の差が重要である．つまり，各種機器から構成情報を収集した時刻に差が大きいと，正しい構成情報とならず，適切な運用ができない．しかし，従来方式では，1 台あたりの収集時間が長時間化しており，更に，4.3.1 項で述べたメモリ使用量の問題のため，大規模なストレージの場合，1 台ずつ情報取得を行わなければならない．そのため，管理対象のリソース数や管理対象のインフラストラクチャ機器の台数が年々増加している状況のなか，大規模データセンターの構成情報の収集時の時刻の差が年々拡大し問題である．たとえば，100 台のストレージから構成されるような大規模環境において，ストレージの利用が少ないと想定される夜間に構成情報の一括収集を行う運用を想定した場合を考える．この場合，従来方式では，100 台のストレージから情報収集するのに 10 時間以上かかる．そのため，一晩 (7 時間) かかっても構成情報の収集が完了しないだけでなく，最初に収集したストレージと最後に収集したストレージでは 10 時間以上もの時刻の差が生じてしまう．そのため，情報の整合性を保つことができず実運用に耐えられない．

4.4 高速情報収集方式の提案

本節では，前章で述べた問題点の解決に向け，管理対象からの情報取得時における使用メモリ量を削減し，更に，削減したメモリを使い情報収集処理を並列化することで収集時間の短縮を行う高速情報収集方式を提案する．

4.4.1 メモリ使用量の削減

取得対象となるリソースの情報取得する際，従来の SMI-S に規定されているオペレーションでは，XML フォーマットの仕様上，全ての XML データを受信しなければ，XML データとしての最低限のフォーマット規約を満たせないため，Java の XML パーサで処理できず，全ての XML データを受信した後にパースしなければならなかった．たとえば，1 台のストレージに 10,000 ボリュームある環境にて，従来の管理ソフトウェアで多々利用されているリクエストである詳細情報取得 (EnumerateInstances) を使った場合，1 回のリクエストで全ボリュームの情報を 1 個の XML データとしてエンコードするため，約 100M のデータ量となってしまい，メモリ消費の主要因となっていた．これを解決するために，次の 2 つの方式を考えた．

(1) 複数リクエスト組み合わせ方式

(2) 取得情報分割方式

(1) は，メモリを大量消費するオペレーションを使わず図 4.1 に示すように，メモリ消費の少ない 2 個のオペレーションに分割することでメモリ量を減らす方式である．

まず，最初のオペレーションとして取得する対象リソースの識別子を取得するオペレーションを実行する．次に取得した識別子を使い，リソース毎の情報を取得するオペレーションを実行し，リソースの詳細情報を取得する．これを，最初のオペレーションで取得したリソースの個数分実行すれば，全リソースの詳細情報が取得できる．この方式は，既存の標準仕様のオペレーションを組み合わせた方式であるため，実現が容易である．

しかし，この方式には二つの問題点がある．1 つ目の問題点は，実行時間が増大してしまう点である．従来まで利用されていた全リソースの情報を一度に取得するオペレーションであれば 1 回の通信であったが，2 個のオペレーションに分割したことで，1+Ni 回 (Ni:リソース数) の HTTP のコネクション処理およびユーザ認証処理のオーバーヘッドが発生し，実行時間が増大してしまう．2 つ目の問題点は，1 回目のオペレーションである識別子一覧を取得するオペレーションのメモリ使用量である．このオペレーションは，識別子一覧のみ取得するため全リ

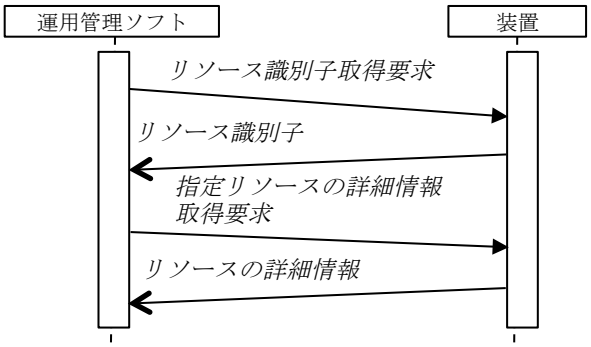

図 4.1: 複数リクエスト組み合わせ方式

ソースの情報を一度に取得するオペレーションに比べメモリ使用量は，約 1/10 に抑えられる．しかし，将来ストレージやサーバなどの管理対象が，大量にリソースを備えた場合に，いずれ対応できなくなる点である．次に，(2) の方式について述べる．(2) は，通信時のデータを分割し，リソースの情報を取得する方式である．まず，はじめに管理ソフトウェアより，1 回の通信で取得するリソース数を指定し，リクエストを送信する．リクエストを受信したストレージなどの管理対象は，リクエストを受け取ると，指定されたリソース数分の XML データに分割し，レスポンスを返す．これにより，1 回の通信で送信されるデータ量を削減でき，メモリ量の削減が見込める．この方式だと (1) の方式で発生していた二つの問題点を緩和できる．問題点の 1 つ目であった実行時間の増大に関しては，1 回の通信で取得するリソース数を，メモリ使用量を見つつ調整することで，リソース数に依存し実行時間が増大していた (1) より削減できる．問題点の 2 つ目については，識別子一覧を取得するオペレーションを利用しないため，発生しない．しかし，(2) の方式は，既存の標準仕様のオペレーションでは実現できないため，特定ベンダの限られた機器でしか実現できない．そこで，著者らは，SMI-S を策定している SNIA(Storage Networking Industry Association) 及び関連仕様を策定している DMTF(Distributed Management Task Force, Inc.) にて議論を行った [86]．その結果，(2) の方式が“ CIM Operations over HTTP 1.3.1 ”の Pulled Enumeration Operations として，国際標準化された [53]．(2) の方式が国際標準化されたことで様々なベンダが利用可能となり，(1) の方式と比べ実行時間，メモリ使用量の観点より有効なため，(2) の方式を提案方式に採用する．

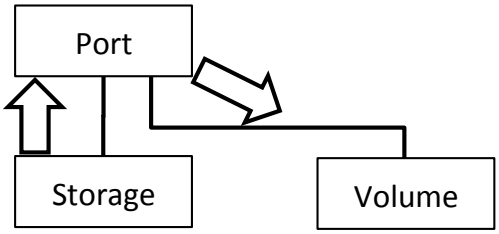

図 4.2: リソースの依存関係と強弱

4.4.2 情報収集処理の並列化

次に，情報収集時間の短縮に向け，構成情報収集処理の並列化を行う．近年，一般的に利用されているオブジェクト指向形式のモデルでは，管理対象のリソースを示すオブジェクト間の依存関係が強く，単純に並列化することができない．

一例をあげると，ストレージが Port とボリュームを有する場合，管理ソフトウェアは，まずストレージの情報を入力値として，そのストレージが所有する Port の情報取得リクエストを発行する．これにより，指定されたストレージ上に存在する Port の情報を取得する．続いて取得した Port の情報を入力値とし，ボリュームの情報の取得リクエストを順次発行することで構成情報を取得する．このようにオブジェクト指向形式のモデルを利用した管理インターフェースでは，リソース間の繋がりを意識したオペレーションとなっており，目的のリソースの情報を取得する際，関連元のリソースの情報が入力値として必要となるため，単純には情報収集処理の並列化はできない．そこで，これを解決するために本提案では，情報収集処理の並列化に向け，まずリソース間の依存関係の強弱に着目し，依存関係の強いリソース群をグループとしてまとめ，このグループ単位で情報収集処理の並列化を行う．

まず，リソースのグループ化について述べる．オブジェクト指向形式のモデルでは，各リソースの関係が表現されており，それぞれの関係には管理ソフトウェアにおける強弱がある．リソースの関係と強弱を表 4.2 にまとめる．

リソースの依存関係が最も強いものとして“ 他リソースの情報を所有 ”がある．例えば，ストレージを示す情報の詳細な値に，そのストレージが所有する Port の識別情報やポート数などの Port 情報の一部情報を持っている場合である．このようなケースではストレージと Port の情報に不整合が生じると，管理ソフトウェア

表 4.2: リソースの依存関係と強弱

強弱 (スコア)	関係
強 (4 点)	他リソースの情報を共有
↑ (3 点)	物理的な接続関係
↓ (2 点)	論理的な接続関係
弱 (1 点)	設定操作時に関係

にて，ストレージの情報として Port の一覧表示していた画面から，Port の詳細画面に遷移しようとした場合に，Port の詳細画面に遷移できない等，管理ソフトウェアにおいて致命的なエラーが発生する．

次に，リソースの依存関係の強いものとして,“ 物理的な接続関係 ”と“ 論理的な接続関係 ”がある．前者は，各リソースがネットワークや結線により物理に接続していることを示している接続関係，後者は，ストレージとホスト間のアクセス制御情報 (Logical Unit Number Security や Zoning) などの各種設定情報である．管理ソフトウェアにおいて，接続関係の情報に不整合が生じてしまうと，ハードウェアの故障による障害管理などにおいて，正常な部位があたかも障害が発生したかのように間違った結果を表示し，重大な問題を引き起こす原因となる．また，論理的な接続関係を利用するケースでは，物理的な接続関係が前提となるケースが多い．例えば，管理者によるオペレーションミスによる設定ミスの検知を実施する場合，物理的な接続関係と現在設定されている設定情報の突き合わせが必要となる．そのため，物理的な接続関係と論理的な接続関係は，物理的な接続関係の方が，依存関係が強い．

最後のリソースの依存関係としては“ 設定操作時に関係 ”がある．これは，ストレージへ接続可能なホストを登録設定など，管理者が行う設定操作において，必要なパラメータとして関係している場合である．例えば，管理ソフトウェアにおいて，設定操作時に管理者が指定するパラメータの一部だけが更新されておらず古い情報であった場合，設定操作がエラーとなる場合がある．このように，リソースの関係の意味により，管理ソフトウェアが運用するリソースの情報に不整合が発生した場合に引き起こされる問題の重要度が異なる．そのため，オブジェクト指向形式で示されたモデルのリソース間の関係の意味を考慮し，情報の不整合が

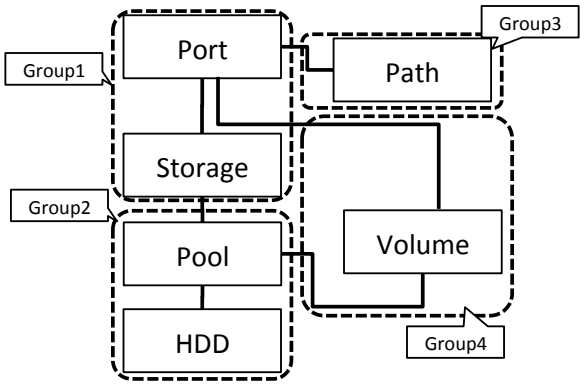

図 4.3: グループ化の一例

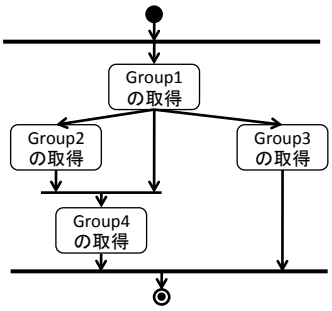

図 4.4: 情報取得の並列化

致命的な問題が発生しないように，リソースをグループ化する．

本章の評価に利用した試作システムでは，リソース間の依存関係の強弱のスコアが 5 点以上の関係を強い依存関係をもったリソースを同一グループとしてグループ化した．グループ化の対象としては，ストレージの構成管理を行うために必要な 51 種類のリソースを対象とし 16 グループに分類した．グループ化したリソースの一例を図 4.3 に示す．

次に，グループ単位での情報収集処理の並列化について述べる．図 4.3 の例を用い説明する．上記で分けたグループ間の依存関係に着目し，Group1 の情報を取得した後，直接接続関係がなく依存関係のない Group2 と Group3 を並列に取得し，Group1 と Group2 と依存関係のある Group4 については，Group1，Group2 の情報取得後取得するように制御する．図 4.4 に構成情報取得における並列化処理をアクティビティ図にて示す．

表 4.3: 試作システム

開発環境	Java5
使用ライブラリ	J WBEM Server 3.1.2
管理対象機器	ストレージ
対象リソース種別数 (CIM クラス数)	51 クラス

表 4.4: 測定環境

PC	CPU:Core2Due 2.00GHz,Memory:4GBytes
OS	Windows XP Professional SP3

このようにリソースを依存関係の強さにてリソースをグループ化し，さらにグループ間の依存関係を考慮することで，構成情報の収集処理の並列化を実現する．

4.5 実験

本節では，4.4 節で述べた提案方式の試作システムを用い，測定した使用メモリ量と構成情報収集時間の結果を述べる．

4.5.1 試作システム

4.4 節の提案方式を適用した試作システムの仕様を表 4.3 に示す．

本試作システムでは，メモリ使用量の削減に向けた取得情報分割方式の実装に，Pulled Enumeration Operations をサポートしているライブラリ J WBEM Server3.1.2 を利用した．

4.5.2 測定環境

次に，4.5.1 項で述べた試作システムを使い，大規模環境で問題となる使用メモリ量と構成情報収集時間を測定した．表 5.2 に測定環境を示す．なお，測定では，一度に取得するリソースの数を 1,000 個 (J WBEM Server での最大数) とした．

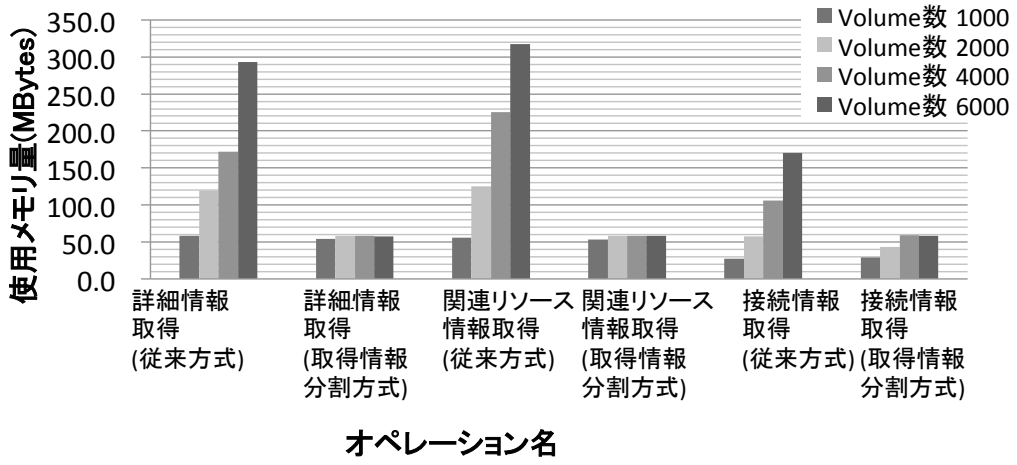

図 4.5: 分割情報取得方式の使用メモリ量

4.5.3 メモリ量に関する測定結果

まず始めに，提案方式の取得情報分割方式におけるメモリ削減効果を測定した．測定では，ストレージのボリューム数を 1,000， 2,000， 4,000， 6,000 と構成変更を行い，使用メモリ量を測定した．なお，測定では管理ソフトウェアで利用される頻度の高い 3 種類のオペレーションに対し測定を実施した．

測定の結果，従来のオペレーション (詳細情報取得 (EnumerateInstances)，関連リソース情報取得 (Associators)，接続情報取得 (References)) では，インスタンス数に比例し，使用メモリ量が増加しているのに対し，提案方式のオペレーション (詳細情報取得 (OpenEnumerateInstanes), 関連リソース情報取得 (OpenAssociatorInstances), 接続情報取得 (OpenReferenceInstances)) では，最大 56MBytes で，使用メモリ量が一定となっている．また，図 4.1 を用いて述べた複数リクエスト組み合わせ方式に関しては，最初のリクエストであるリソース識別子一覧取得の要求は，図 4.5 の詳細情報取得のサブセットの要求のため，使用メモリ量は，インスタンス数に比例し増加する．

4.5.4 構成情報収集時間に関する測定結果

構成情報の収集時間に関する測定では，取得情報分割方式におけるオーバヘッドの影響及び情報取得処理の並列化における最適並列数を調査すべく実施した．

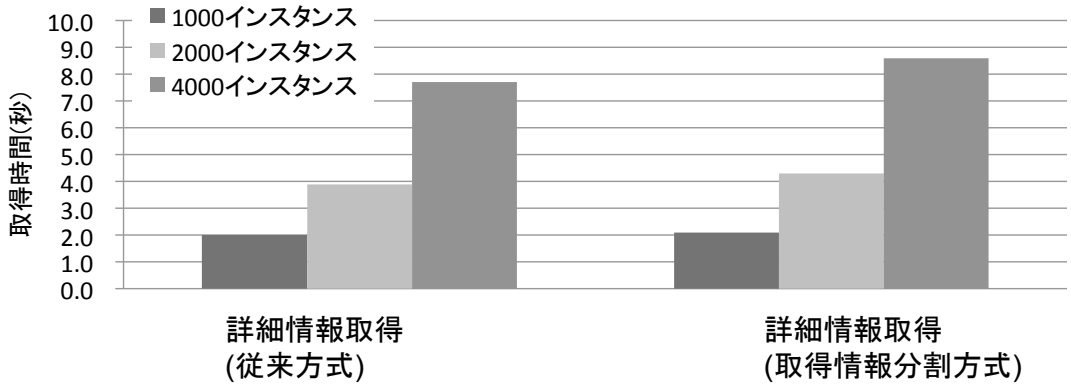

図 4.6: 取得情報分割方式のオーバヘッド

(1) 取得情報分割方式におけるオーバヘッドの影響測定

取得情報分割方式では，XML データを分割し情報の取得を行うため，従来方式ではなかった XML データの分割処理が必要となる．そのため，従来方式では 1 回のリクエストで全リソースの情報を一度に取得していたのに比べ，全リソースの情報を取得するのに複数回リクエストを発行する必要がある．例えば，4,000 個のリソースを取得する場合，取得情報分割では，1 回のリクエストで取得するリソース数を 1,000 個とした場合，全リソースの情報を取得するのに，従来方式に比べ 3 回のリクエストが増える．そのため，本測定では，分割処理におけるオーバヘッド及びリクエスト回数が増加することに伴うオーバヘッドについて測定した．結果を図 4.6 に示す.

測定の結果，提案方式で新たに発生するリクエスト回数が増加することに伴うオーバヘッドに関しては，リソース数 2,000，リソース数 4,000 の場合の測定結果を比較すると，平均 0.3 秒/リクエストであり，影響は少ない.

(2) 情報取得処理の並列化における最適並列数

次に，試作システムを使い，最も効果的な並列数 (スレッド数) を調査すべく，主要ストレージベンダ 3 社のストレージに対し構成情報の収集時間について測定を実施した．測定環境としては，A 社ストレージ 3,200 ボリューム，B 社ストレージ 3,100 ボリューム，C 社ストレージ 3,050 ボリュームのストレージを用いた．測定の結果を図 4.7 に示す.

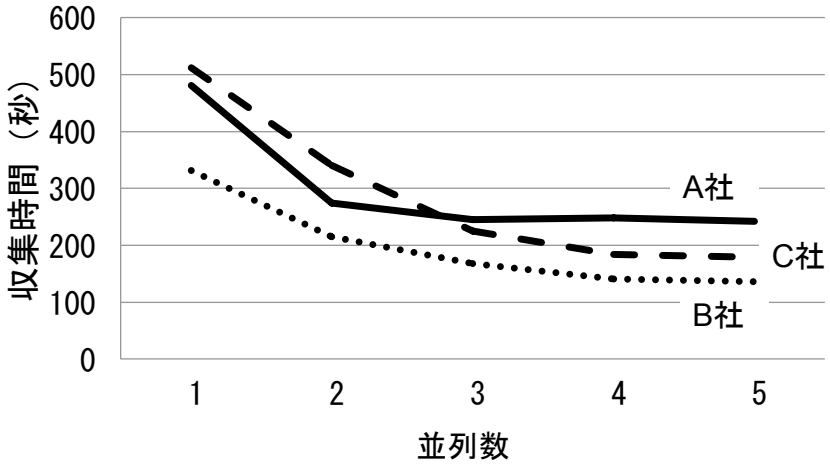

図 4.7: 構成情報取得時の収集時間とスレッド数

主要ストレージベンダ 3 社のストレージに対し測定した結果によると，スレッド数 1 から 2 にかけて，並列処理による効果が高く各社ともスレッド数 4 で並列処理の効果が収束傾向となった．また，試作システムでの CPU の処理速度及び使用メモリ量の限界による影響を無くすため，3 社のストレージにおいて，100 ボリューム程度の構成にて測定を行ったが，図 4.7 と同様にスレッド数 4 で収束傾向であった．このスレッド数の上限は，各社のストレージのログファイルを確認したところ，ストレージ側で行われているオブジェクト指向形式のモデルの情報生成処理の処理能力がスレッド数 4 で収束しているためと考える．

4.6 考察

本節では，メモリ量削減に向け適用した取得情報分割方式と，情報収集時間短縮に向け適用した情報収集処理の並列化について，それぞれ有効性を述べた後，これらを組み合わせた本提案の高速情報収集方式の有効性について議論を行う．

4.6.1 取得情報分割方式の有効性

4.3.1 項に述べたように，従来方式では，取得するリソース数に比例し使用メモリ量が増加していた．これに対し，取得情報分割方式では，取得するリソース数が増

加しても，使用メモリ量は一定で抑えられており，最大約 56MBytes のメモリ量となることが 4.5.3 の測定結果より実証された．また，取得情報分割方式では，XML データを分割取得するため，従来方式に比べ情報取得時間について XML データを分割する処理時間及びリクエスト回数が増加することに対する処理時間のオーバヘッドが生じるが，4.5.4 項 (1) の測定の結果，構成情報の収集処理に与える影響は少ないことが判明した．

これにより，従来方式でボリュームの情報を収集する場合，32bit 環境の Java の環境下では，1 台のストレージあたり約 20,000 ボリュームのリソースがあると，メモリ不足となり処理の限界であったが，取得情報分割方式では，リソース数が増加しても，メモリ使用量を一定に抑えることができると推察する．これにより，1 台のストレージで 20,000 ボリューム以上ものリソースを標準で備えるエンタープライズクラスのストレージを管理する場合であっても，取得情報分割方式を用いれば，メモリ不足を発生させることなく構成情報を収集できると考える．これらより，取得情報分割方式はメモリ削減に有効な方式であると言える．

4.6.2 情報収集の並列化方式の有効性

次に，情報収集の並列化について述べる．従来は，4.4.2 項で述べたように，オブジェクト指向形式のモデルで表現された管理インターフェースを利用する場合，各リソース間の依存関係を考慮する必要があるため，構成情報の収集処理をシーケンシャルに実行せざるを得なかった．そこで，提案の並列化方式では，各リソースの依存関係の強度を考慮したグループ化を行うことで，並列処理を可能とした．この並列化方式は，4.5.4 項 (2) の測定結果より，ボリューム 数が約 3,000 ボリュームの規模のストレージから構成情報を収集するのに，従来方式だと 3 社の平均で約 7 分 (441 秒) かかっていたものが，並列化方式では，約 3 分 (191 秒) になり，約 57%もの時間短縮に成功した．これは，複数台のストレージを運用するケースにおいて，N 台のストレージから構成情報を収集するのに，441N 秒必要だったものが 191N 秒に短縮できる．

次に，管理できるデータセンターの規模の観点から述べる．ストレージからの構成情報収集を一晩 (7 時間と想定) で完了しようと計画した場合，従来方式だと

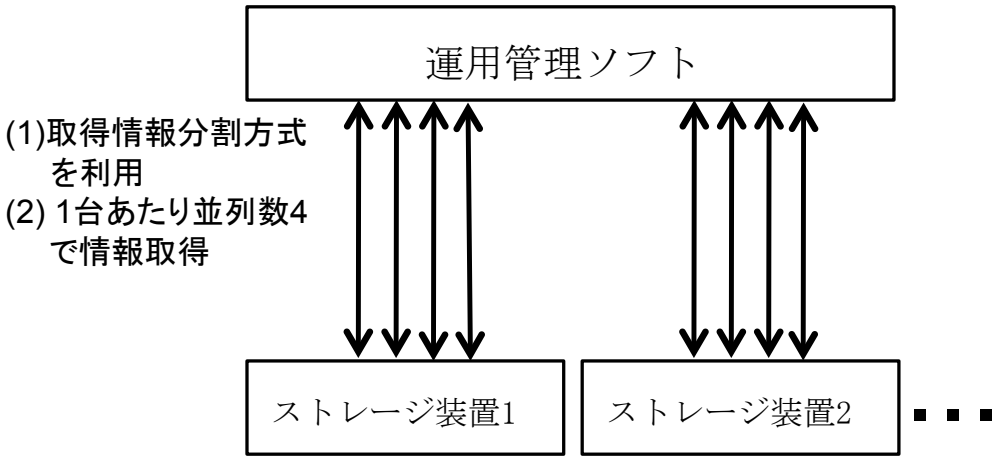

図 4.8: 複数台のストレージからの情報収集

441N 秒必要であったため，57 台まで扱えなかった．これに対し，並列化方式では 191N 秒に短縮したことで，131 台まで管理できるようになる．つまり，構成情報の収集処理を並列化したことで，収集時間を短縮でき，従来方式の 2 倍以上の台数のインフラストラクチャ機器を管理できるようになった．これらより，提案の並列化方式は構成情報の収集時間の短縮に有効であると言える．

4.6.3 高速情報収集方式の有効性

4.6.1 節，4.6.2 節より，取得情報分割方式と情報収集の並列化方式がそれぞれ使用メモリ量削減，構成情報の収集時間の短縮，に有効であることが判明した．そこで，これら 2 つを組み合わせた提案方式である高速情報収集方式について有効性を述べる．まず，1 台のストレージからの情報収集に関しては 4.6.2 節で述べたように，並列数 4 で処理した場合，各社の平均を取ると 191 秒であった．この場合の使用メモリ量は，取得情報分割方式を利用することで，1 スレッドあたり最大 56MBytes となることから，並列数 4 で最大 224MBytes となる．これは，4.3.1 節で述べたように 32bit 環境の Java VM のヒープサイズが理論値で最大 2Gbytes であることから，問題ないメモリ量であると考える．次に，さらに情報収集の処理時間の短縮を狙い図 4.8 に示すように複数台のストレージから同時に情報収集することを考える．

これにより，例えば，ストレージ 1，ストレージ 2 と 2 台のストレージから情報を収集する場合，合計 8 スレッドの並列数で情報収集する．この複数台のストレージから情報収集する場合の使用メモリ量と情報収集時間の関係を式 4.1, 4.2 にて示す．

$$Um = 224T \tag{4.1}$$

$$Rt = 191N/T \tag{4.2}$$

Um: 使用メモリ量 (MBytes)

T: 同時に情報収集するストレージ台数

Rt: 情報収集時間 (Sec)

N: 管理対象とするストレージの全台数

例えば，ストレージからの構成情報収集処理に，32bit 環境の Java VM の最大ヒープサイズ 2Gbytes の半分である 1GBytes のメモリを割り当てたと仮定する．この場合，100 台のストレージから構成情報を収集するのに必要な時間を式 (2) にあてはめ算出すると，同時に構成情報収集可能なストレージの台数は 4 台となり，構成情報取得時間は，約 1 時間 20 分 (4,775 秒) ですべてのストレージからの構成情報収集が可能となる．これにより，従来方式では，約 12 時間以上 (44,133 秒) 必要であった情報収集の処理時間を約 90%短縮できる．

次に，管理できるデータセンターの規模の観点から述べる．ストレージからの構成情報収集を一晩で完了しようと計画した場合，構成情報が収集能なストレージの台数も従来方式では 57 台であったものが，提案方式では 527 台と 9 倍以上もの台数を管理できるようになる．これは，2010 年時点で 60 台以上のストレージを有するデータセンターが登場しており，2015 年には 100 台，2020 年には 500 台を超えるストレージを有する大規模データセンターが登場すると予測されている状況において，本提案の高速情報収集方式は，大規模データセンターの構成情報を一元管理するために十分なスケーラビリティを有した方式であると考える．

4.7 結言

本章では，インフラストラクチャ機器の構成情報の収集処理における使用メモリ量と処理時間を削減する高速情報収集方式を提案した．従来の管理ソフトウェアで用いられていた方式では，使用メモリ量と構成情報の収集時間の増大が原因で，年々大規模化し 2015 年には 100 台以上ものストレージが存在すると予測されているデータセンターを一元管理することができなかった．そこで，本提案方式は，従来方式と比べ 90%の情報収集時間を短縮し，さらに管理台数も 9 倍以上のスケールアップを実現した．これにより，2015 年の環境だけでなく，現在と同じペースで増加すると仮定した場合に 500 台以上のストレージが存在すると予測される 2020 年環境も一元管理できるようになった．

更に，ストレージだけでなく仮想サーバにおいても，年々利用台数が増加しており，近い将来，ストレージと同様に情報収集時の限界問題が発生する．これに対しても，本提案方式は，仮想サーバが備える管理インターフェースにも適用できるため，ストレージと同様に解決ができる．また，仮想サーバのスケーラビリティに関しては，本章で述べた 4.5.3 節，4.5.4 節の測定結果と近似の値と考えられる．これは，代表的な仮想サーバである VMware で利用されている CIM のクラスは 48 クラスであり，本章の試作プログラムで対象としたストレージの CIM のクラス 51 クラスと近似のクラス数であり，更にそのモデルも類似しているためである．

この高速情報収集方式により，大規模データセンターを一元管理可能とする管理基盤を構築した．

第 5 章

知識データを使ったデータセンター全体視点の管理省力化

本章では，管理者が持つインフラストラクチャ機器に関する知識をデータ化し，これを活用することで，データセンター全体で一貫性を持った運用を実現するとともに，管理者の負荷を軽減する管理省力化方式について述べる．以降，データ化した管理者の知識を知識データと略す.

5.1 緒言

1.1 節で示したように，世界中で生み出されるデータ量が加速度的に増加しており，2020 年には 2013 年の約 10 倍の 44 ZB に到達する見込みである．さらに，サーバ，ネットワーク，ストレージなどのインフラストラクチャ機器のデータセンターへの集約が進んでおり，データセンターが大規模化している．このようななか，大規模データセンターでは，設備コストを削減するために，仮想化技術を使ったサーバやストレージのコンソリデーションが進められている．しかし，一方でサーバやデータセンターの大規模化と仮想環境によるシステムの複雑化により，運用コストは増加傾向にある．さらに，これを運用する管理者の数は一定傾向にあり，大規模化，複雑化するデータセンターを少人数の管理者で運用しなければならない [3].

このようななか，仮想環境を柔軟に構成変更し性能やメディアコストを適正化

するために，仮想サーバ (Hypervisor) 上の VM(Virtual Machine) へ割り当てられたデータを，VM を停止することなくデータを自動移動するデータ移動技術が登場している．このデータ移動技術は仮想サーバやストレージなど異なるレイヤの実装がある．しかし，これらのデータ移動技術は，各々のレイヤの視点でデータ移動を行うため，性能やメディアコストが個別最適となり，データセンター全体視点での性能やメディアコストを考慮したデータ移動は困難である．そのため，データセンター全体視点での性能やメディアコストを考慮しようとした場合，高度な知識をもった管理者の経験に頼らざるを得ず，管理者の負担となっていた．データセンターの大規模化，複雑化が進んでいる状況を考慮すると，管理者の負担が益々増加し，その結果，設備コストは下がったものの，逆に運用コストが増加してしまい，トータルのコストは下がらないことが多々ある．

そこで，本章では，管理者が有していたデータセンター全体の構成情報や各種ストレージリソースの性能，メディアコストの情報を使い，データセンター全体での性能とメディアコストを適正に配置する自動データ適正配置方式を提案する．これにより，データセンター全体視点で管理者の要求する性能を保ったままメディアコストの削減と，管理者の知識を使ったデータの自動配置を行う事で管理者の負担を削減する．

以下，5.2 節では従来のデータ移動技術と問題点について述べ，5.3 節では提案方式について述べる．次に 5.4 節では，提案方式の試作システムを使った測定結果を述べ，5.5 節で測定結果に基づいた提案方式の有効性を考察する．

5.2 従来のリソース配置技術と問題点

本節では，従来のデータ移動技術と特徴を紹介した後，問題点について述べる．

5.2.1 従来技術

近年，サーバやストレージの仮想化が進み，CPU やメモリ，ストレージ (Volume) など様々なリソースが仮想化されている．さらに，仮想化されたリソースは，利用していない VM のリソースを性能不足の VM に割り当てることで，コストを抑

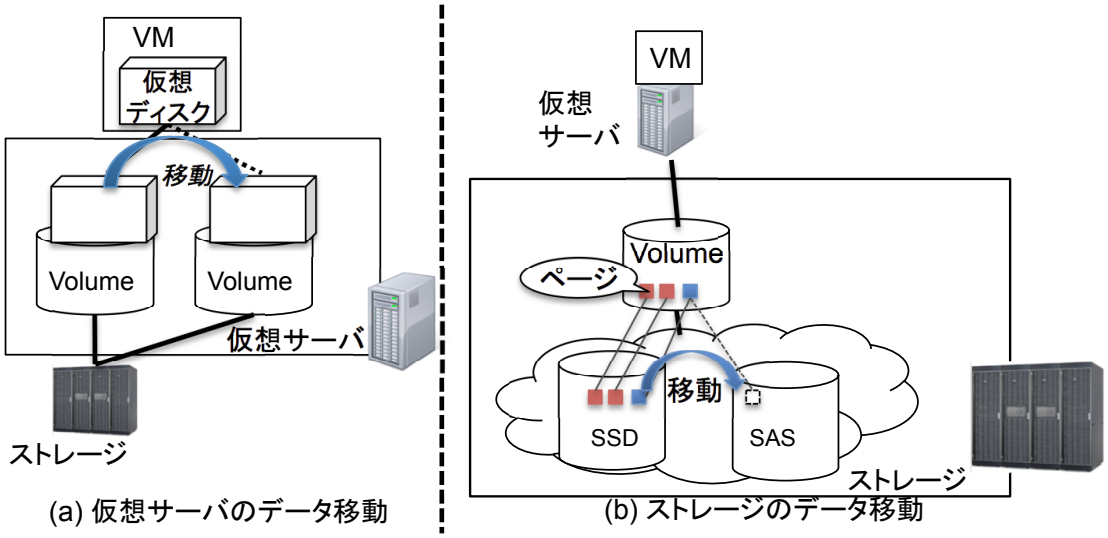

図 5.1: データ移動技術

えつつ要求性能を満たすことが求められている．そのため，性能の観点によるネットワークや CPU, メモリの割り当てだけでなく，性能とコストの両観点によるデータを格納するメディアの割り当てが重要視されている．このようななか，VM に割り当てられた Volume 上の仮想ディスク (Virtual Machine Disk や Virtual Hard Disk) を無停止で移動する技術が仮想サーバ，ストレージと異なるレイヤの装置向けに研究 [64][67] や製品化が進められている．図 5.1(a) に仮想サーバ，(b) にストレージのデータ移動技術の詳細を示し，その特徴を表 5.1 にまとめる．

図 5.1(a) に示す仮想サーバのデータ移動技術は，仮想サーバが受信した I/O の統計情報をもとに，VM に割り当てられた Volume 上の仮想ディスクを，別 Volume へ移動させる技術である．この技術により，仮想サーバが備え持つ複数の内蔵 Volume 間やストレージ間のデータ移動ができる．代表的な製品として，VMware の Storage Distributed Resource Scheduler [87] がある．

ストレージのデータ移動技術は，ストレージ仮想化の一部として提供されている．まず，ストレージ仮想化技術の代表的なものに Thin Provisioning [30] と呼ばれる容量の仮想化技術がある．容量の仮想化技術は，ストレージへの書き込みがあった際に，書き込みデータ分の領域 (以降，ページと略す) をメディア上に確保することで，事前に大量のメディアを用意する必要をなくす技術である．さらに，ストレージ仮想化は，性能やメディアコストの異なる複数メディアを仮想的に階層

表 5.1: データ移動技術の特徴

提供装置 / 特徴点	仮想サーバ (Hypervisor)	ストレージ
移動範囲	装置内のメディア間の移動, 装置外のメディア間の移動	装置内のメディア間の移動
VM への負荷	あり	なし
移動判断材料	仮想サーバの I/O 性能	ストレージの I/O 性能
移動データの単位	仮想ディスク単位	ページ
自動移動契機	I/O 性能劣化時	一定時間毎

化している．階層の仮想化では，SSD(Solid State Drive) や SAS(Serial Attached SCSI)，SATA(Serial Advanced Technology Attachment) のハードディスクなど異なる種別のメディアを 1 つの Volume にまとめサーバへ割り当て，ストレージが受信した I/O の統計情報をもとに，性能やメディアコストの異なる複数メディア (SSD， SAS など) 間をページ単位でデータ移動させ，階層の仮想化を実現している．このような，階層の仮想化のデータ移動技術により，仮想サーバや VM へ負荷をかけずデータ移動ができる．代表的な製品として， Hitachi Virtual Storage Platform の Dynamic Tiering [31] がある．

このように，仮想サーバ，ストレージのデータ移動技術には，それぞれ特徴がある．

5.2.2 問題点

5.2.1 節の技術を使い，管理者は性能要求を満たす目的や，メディアコストの削減目的でデータ移動を行っている．しかし，従来技術では，異なるレイヤのデータ移動技術の矛盾動作，性能とメディアコストの個別最適，管理者への負担，の 3 つの点で問題がある．

(1) 異なるレイヤのデータ移動技術の矛盾

仮想サーバとストレージのデータ移動技術を併用すると，異なるレイヤで相反するデータ移動を行うことがある．相反するデータ移動の例を図 5.2 にて紹介す

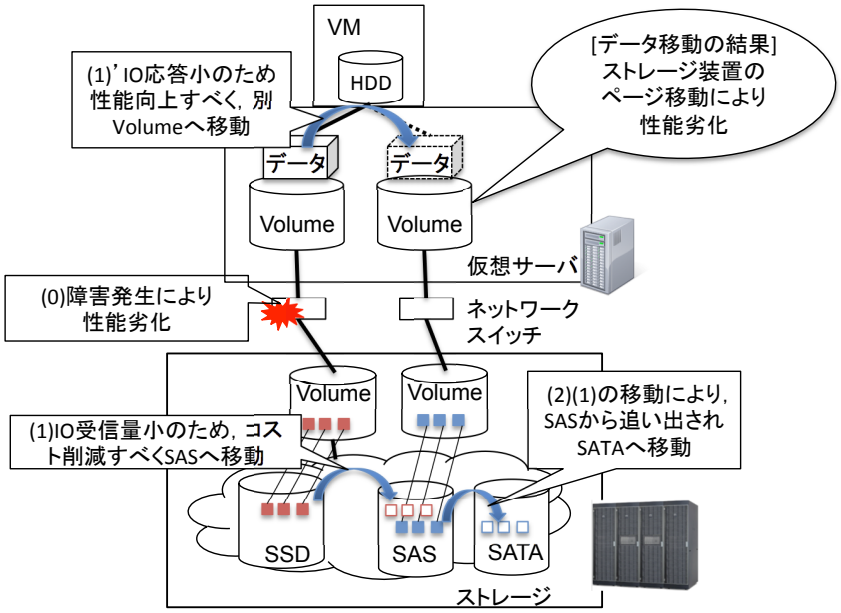

図 5.2: データ移動技術の併用時に発生する矛盾の一例

る．本例では，障害発生でネットワークの性能劣化が発生すると，仮想サーバは性能向上をさせるために別 Volume へデータ移動しようとするが，ストレージはメディアコスト削減のために低速なメディアへデータ移動してしまい，より一層性能が低下する．理想の動作としては，仮想サーバにて性能向上をさせるために別 Volume へデータ移動をした際，ストレージ側は移動は行わないか，もしくは性能向上させるよう移動させることで，システム全体として両方のデータ移動技術が協調し性能向上を図ることである．しかし，2014 年時点では，このような矛盾したデータ移動を行わないように，ベンダー自ら何れか一方のデータ移動技術のみ利用することを推奨している [88].

(2) 性能とメディアコストの個別最適

仮想サーバのデータ移動技術は I/O 性能によってデータの移動先を決定し，ストレージのデータ移動技術は，I/O 性能とメディアコストを意識したメディア種別の情報を使い決定する．仮想サーバのデータ移動では，ストレージのメディアの種別が取得できないため，メディアコストを意識した移動ができず，性能が最適になるようにデータ移動する．一方，ストレージは，仮想サーバが備えるメディアコストやネットワークの性能を考慮せず，ストレージ内のみ性能とメディアコ

ストが最適になるようデータ移動する．そのため，仮想サーバ，ストレージがそれぞれの視点での個別最適となり，データセンター視点での全体最適な性能とメディアコストになっていない．

(3) 管理者への高負荷

仮想サーバ，ストレージのデータ移動技術は登場してきたものの，データセンター全体を考慮した適切なデータ配置を実現するためには，(1)(2) の問題により，自動でのデータ移動技術は利用できない．そのため，データセンター全体のストレージの性能・メディアコスト，仮想サーバやストレージの構成情報を把握している高度な管理者の知識や経験に頼り，データの配置を検討しデータ移動を行っていた．これは，管理者の負荷となっていた．さらに，2012 年において，銀行や通信キャリアなど大規模なデータセンターを所有するエンタープライズ企業の中には，既に 80 台をこえるストレージを備えるデータセンターが登場し始めており，調査会社の報告によると 2015 年には，ストレージが 100 台以上，VM は 50,000 台以上となる大規模なデータセンターの登場が予測されている [79]．このような大規模データセンターでは，管理対象が増加することで管理者の負担も増加している．

このデータ配置に関する管理者の負担の量を数値化するために定式化する．管理者の一人当たりの 1ヶ月あたりの作業時間を Wt，1 サービスの初期構築にかかる時間を It，1ヶ月あたりのデータ配置の見直し回数を N，1 サービスあたりのデータ配置計画時間を Pt，提供するサービス数を S，管理者数を A，とすると管理者の負荷はサービスを開始する始めの月の作業時間 (W0) を式 5.1，その翌月以降の月々の作業時間を式 5.2 のように表すことができる．

$$W0 = It \times S/A + Wt \tag{5.1}$$

$$Wt = NPt \times S/A \tag{5.2}$$

式 5.1，式 5.2 を使い，2015 年の大規模環境における管理者の負担を試算する．VM10 台で 1 つのサービスを提供，1 サービスの初期構築にかかる時間を 60 分，1ヶ月あたりのデータの見直し回数を 4 回，1 サービスあたりのデータ配置の計画時間を 10 分，高度な管理者 30 人が分担して運用していると仮定する．この場合，式 5.1，式 5.2 に当てはめ管理者の作業時間を算出すると，サービスを開始する始め

の月は 16,667 分かかり，その翌月からは 6,667 分かかることになる．また，管理者が 1 日あたり平均 8 時間の労働，月 20 日間勤務したとすると，1ヶ月あたりの作業時間は 9,600 分である．そのため，サービスを開始する始めの月の作業負荷は 174%，その翌月からは 70%となる．サービスを開始する月に関しては，規定の業務時間だけでは対応できず，翌月からもデータ配置の作業だけで，管理者が行う作業の 70%がデータ配置の作業になる．これは，他の管理業務も考えると，現実的ではない．

また，仮想環境の進展によりシステムが複雑化し各リソースの依存関係も考慮しなければならないことを考慮すると，1 サービスあたりのデータ配置計画時間が長くなるだけでなく，複数管理者による分業が難しくなり，さらに負荷は増大する．

5.3 知識データ化とデータ適正配置方式の提案

本節では，5.2 節の従来技術の問題点の解決を狙い自動データ適正配置方式を提案する．まず始めに，方針，システム構成を述べた後，本提案方式の特徴であるデータ適正配置アルゴリズムについて述べる．

5.3.1 方針

従来，管理者がデータセンター全体の性能とメディアコストを考慮しデータ移動を行う際，以下のステップにて運用するのが一般的であった．

(1) 監視/分析
仮想サーバ，ストレージ各々の管理ソフトウェアを使い性能，構成情報を監視し，性能比較を行うことで，性能ボトルネックの部位を特定

(2) 計画
性能ボトルネックの部位情報を元に，移動するデータ及び移動先を管理者の知識や経験により決定

(3) 実行
計画に従い，仮想サーバもしくはストレージのデータ移動技術によりデータ移動を実行

そこで，本提案では，この運用ステップをベースとし，5.2.2 節の問題点を以下の方針で解決する．

[方針１] 管理ソフトウェアによる仮想サーバ，ストレージの性能・構成情報を一元管理

[方針２] 管理者の知識や経験に依存していた計画ステップをアルゴリズム化

[方針３] 方針２と方針３により，監視/分析，計画，実行の各ステップを連携させ自動化

次節より，本方針に従い設計したシステム構成及びデータ適正配置アルゴリズムについて説明する．

5.3.2 システム構成

提案方式のシステム構成を図 5.3 に示す．本システムでは，まず始めに，仮想サーバ，ストレージのインフラストラクチャ機器より性能・構成情報を収集する．次に，収集した情報から性能ボトルネック分析部，データ配置計画部にてデータ配置の計画をたて，最後に仮想サーバまたはストレージに対しデータ移動を指示する．これを管理者の性能要件を満たすまで繰り返し，データの適正配置を行う．

なお，本システムでは，データ配置計画を行うにあたり，管理者からの性能要件を事前設定し，データ配置計画を立てる．この性能要件については，管理者とユーザの間で取り交わされるサービスレベルの保証契約である SLA(Service Level Agreement) を想定している．

5.3.3 データ適正配置アルゴリズム

5.3.2 節で示したシステムにて動作するデータ配置を行うためのアルゴリズムであるデータ適正配置アルゴリズムについて述べる．データ適正配置アルゴリズム

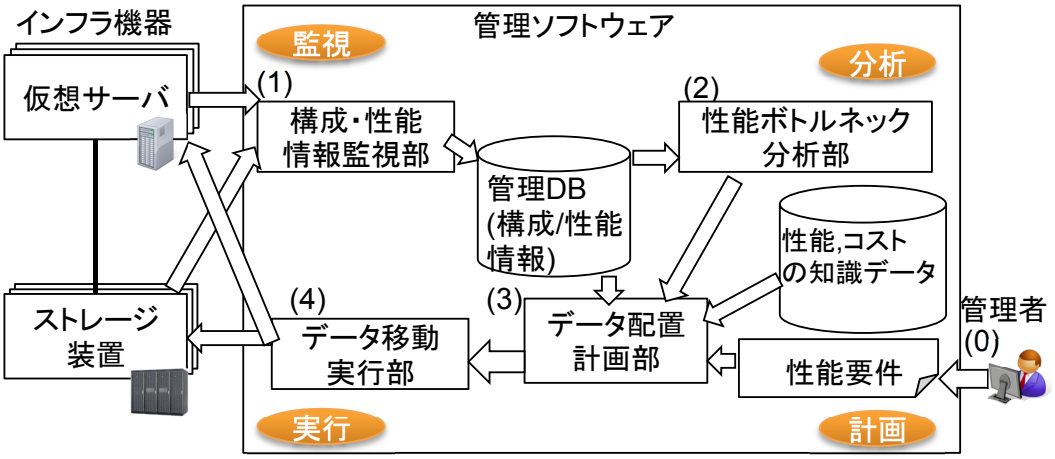

図 5.3: システム構成

を Algorithm1 に示す.

データ適正配置アルゴリズムは，5.3.1 節に示した従来管理者が行っていた監視/分析，計画，実行の 3 つのフェーズにて実現する．監視/分析フェーズは Algorithm1(1 行目-8 行目) で示されるフェーズ，計画フェーズは Algorithm1(9 行目-29 行目) のフェーズ，実行フェーズは Algorithm1(30 行目-31 行目) のフェーズである．

監視/分析フェーズでは，仮想サーバとストレージから取得した構成情報から VM に割り当てられた仮想ディスクが格納されている Volume を特定する．その後，仮想サーバとストレージから取得した Volume の性能を比較し，ボトルネック箇所を特定する．

計画フェーズでは，まず始めにボトルネックの箇所に応じ移動対象となるデータ (仮想ディスク) の決定を行う．次に，管理者からの性能要求から移動目的が性能向上なのか，コスト削減なのかを判断し，目的に応じて移動先のメディアを決定する．この移動対象データの候補決定ステップ (Algorithm1(9 行目-13 行目)) と，移動先の決定ステップ (Algorithm1(14 行目-29 行目)) では，従来管理者が有しているメディアの性能やコストの情報を知識データ化し算出する．

最後の実行フェーズでは，計画フェーズにて決定した移動対象データと移動先から，ストレージのデータ移動を使いデータ移動を行うか，または仮想サーバのデータ移動技術を使いデータ移動を行うかを決定し，データ移動を実行する．

次節より，このデータ適正配置アルゴリズムの計画フェーズの移動対象データ

Algorithm 1 データ適正配置アルゴリズム

1: $resources \leftarrow$ 改善対象のリソースの格納先リソースを管理 DB より特定
2: $perf1 \leftarrow$ 仮想サーバから$resource$ の応答性能の情報を取得
3: $perf2 \leftarrow$ ストレージ装置から$resource$ の応答性能の情報を取得
4: **if** $perf1 = perf2$ **then**
5: $bottleneck \leftarrow STORAGE$
6: **else**
7: $bottleneck \leftarrow OTHER$
8: **end if**
9: **if** $bottleneck = STORAGE$ **then**
10: $targets \leftarrow resource$ 上のデータを選出
11: **else**
12: $targets \leftarrow$ 合計サイズが最小となるデータの組み合わせを選出
13: **end if**
14: $dests \leftarrow targets$
15: **while** 全メディア数 **do**
16: $perf3 \leftarrow$ 現在の性能改善対象リソースの応答性能の情報を取得
17: **if** $sla > perf3$ **then**
18: {性能向上が目的の場合}
19: $dests \leftarrow dests$ のリソースより性能が高いリソースを選択
20: **else**
21: {コスト削減が目的の場合}
22: $dests \leftarrow dests$ のリソースよりコストが低いリソースを選択
23: **end if**
24: $predictperf \leftarrow$ 性能・メディアコスト比と $perf3$ から移動先メディアの予測性能を算出
25: $predictperf \leftarrow predictperf *$ データ移動中の計算処理への影響度
26: **if** 管理者の性能要件を満たしている **then**
27: break
28: **end if**
29: **end while**
30: $machines \leftarrow targets, dests$ を所有する機器を特定
31: $machines$ のデータ移動機能を呼び出し $targets$ を $dests$ へ移動

の候補決定と移動先の決定について詳細に述べる．

移動対象データの候補決定

Algorithm1(9 行目-13 行目) に示す移動対象データの候補決定では，単純に性能ボトルネックとなっているデータを移動すれば良いとは限らない．性能ボトルネックとなっているデータを移動させることで移動中にさらなる性能低下が発生する場合には，他のデータを移動させ，性能改善を行うケースもある．そこで，性能ボトルネックの部位により整理する．

性能ボトルネックがストレージにある場合，性能ボトルネックになっているデータを，I/O 性能の高いメディアに移動させる必要がある．この場合，表 5.1 に示したデータ移動技術の特徴より，VM への負荷がないストレージでのデータ移動を行うのが良い．そのため，本ケースの場合，データ適正配置アルゴリズムでは，性能ボトルネックになっているデータを移動対象データの候補とする．

しかし，性能ボトルネックが仮想サーバもしくはネットワークにある場合，仮想サーバのデータ移動技術により移動を行うことになる．このデータ移動は，表 5.1 の特徴に示すように仮想サーバへ負荷が発生し，仮想サーバが提供する VM 上で実行している計算処理が低下する．そのため，仮想サーバへの負荷が少なくなるように VM への影響が小さいストレージリソースを選出することが重要である．VM への影響が小さくなるために考慮すべき重要なポイントは移動時間の短さである．そこで，データ適正配置アルゴリズムの移動対象データの候補決定では，移動時間を短くするために，移動するデータのサイズが最小となるデータの集合を求める．

移動対象データの候補は，Algorithm1(1 行目) の情報から性能ボトルネックの VM が格納されている Volume 上に存在する VM 群の中から選出する．各々の VM に割り当てられた仮想ディスクのサイズ S とすると，式 5.3 に示すように移動対象の候補となるデータの合計サイズ D が算出できる．この D が最小となる移動対象データの組み合わせを求め，データ適正配置アルゴリズムにおける Algorithm1(12 行目) に示す移動対象データの候補とする．

$$D = \sum_{k=1}^{n} S_k \tag{5.3}$$

移動先の決定

次に，Algorithm1(14 行目-29 行目) の移動先の決定について述べる．

移動先の決定では，移動後の性能・メディアコストのバランスと，移動中にかかる VM 上の計算処理への影響について考慮する必要がある．移動後の性能・メディアコストのバランスについては，管理者の性能要件を満たしつつ最もコストの低いメディアの選別が重要である．

そこで，仮想サーバ，ストレージが備える各種メディアにデータを配置したときの VM 上の計算処理性能とメディアコストの関係を示した性能・メディアコスト比，データ移動中の VM 上の計算処理への影響度を知識データ化し利用することで，データの移動先を決定する．

データの移動先決定のステップである Algorithm1(14 行目-29 行目) を以下に説明する．

(14 行目-23 行目) 1 段階性能が高いメディアもしくはコストが安いメディアを移動先として選出する．

(24 行目) 現時点の VM の仮想ディスクの性能 (例えば ResponseTime) に，性能・メディアコスト比 (後述の表 5.3) の性能値を乗算し，移動後の性能予測値を算出する．

(25 行目) データの移動中にかかる VM 上の計算処理への影響 (後述の表 5.4) を加味し 24 行目の性能予測値を補正する．

(26 行目-28 行目) 25 行目の性能予測値が，管理者の性能要件を満たさない場合，再度データ移動候補の選出 (15 行目) に戻り，次候補を選出しデータ移動先決定ステップを繰り返す．

これにより，管理者の性能要件を満たしつつ，低価格なストレージを移動先として選出する．

表 5.2: 測定環境

物理サーバ	Dell OptiPlex (CPU: Intel Corei7 3.4GHz, Memory16G)x2
仮想サーバ	VMware ESXi5
ストレージ	Hitachi Virtual Storage Platform
接続形態	物理サーバ/ストレージ間 (FC 直接続，8Gb/秒)
ミドルウェア	分散処理環境 Hadoop 1.0.2
VM	CentOS 6 x 10 台 (1VM あたり仮想ディスク 1 個)
測定プログラム	Hadoop1.0.2 付属の terasort (2G のデータをソート)
試作プログラム	Windows7, Java6 にて開発

5.4 実験

本実験では，5.3.3 節で述べた性能・メディアコスト比とデータ移動中の VM 上の計算処理への影響を知識データ化すべく事前測定を行う．その後，知識データ化した情報を適用した試作システムを用い，管理者が入力した性能要件との一致度について測定する．

5.4.1 測定環境

測定環境を表 5.2 と図 5.4 に示す．測定では，複数 VM が協調して処理を実施する分散処理環境を用い測定を行った．

5.4.2 データ適正配置アルゴリズム向け事前測定

データ適正配置アルゴリズムで利用する管理者が従来有していたサーバ/ストレージ横断での各種メディアの性能・メディアコストの情報とデータ移動による VM 上の計算処理へ与える影響を知識データ化すべく測定を実施した．

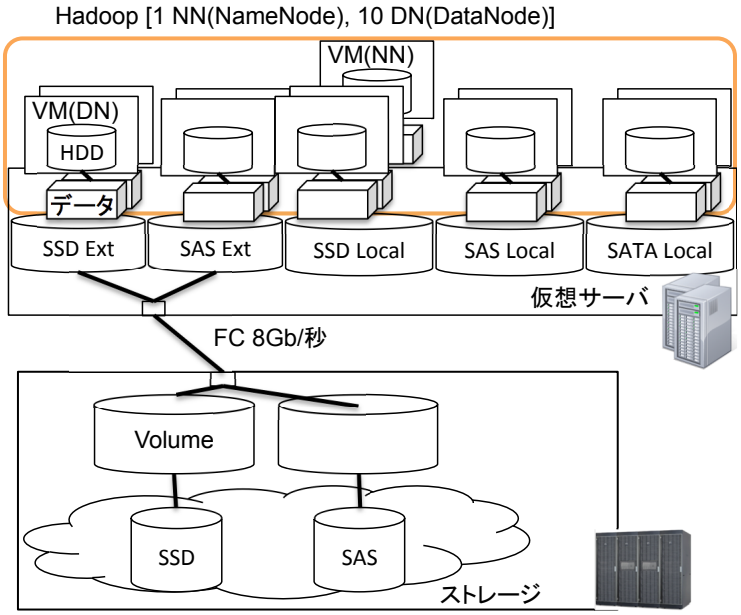

図 5.4: 測定システム

表 5.3: 性能・メディアコスト比

	SSD(Local)	SAS(Local)	SATA(Local)	SSD(Ext)	SAS(Ext)
性能比	1/1	10/29	10/52	1/1	10/29
コスト比	49.5	17.3	1.0	99.0	34.6

性能・メディアコスト比の測定

本測定では，5.3.3 節の移動先の決定にて利用する性能・メディアコスト比を知識データ化すべく測定を行った．本測定では，5.4.1 節の環境にて，単一種類メディアで構成された Volume 上に全 10 個の計算処理ノード (DataNode) の VM のデータを配置した後，測定プログラムを実施し，各種メディアにデータを配置した場合における VM 上の計算処理性能を測定した．さらに，各種メディアのコストに関しては，2011 年 SNIA Data Protection and Capacity Optimization(DPCO) Committee [89][90] にて公開されている GByte あたりのコストを元に比率を算出した．

測定結果を表 5.3 に示す．メディア種別の Local は，仮想サーバが備える内蔵のメディアを示し，Ext は，ストレージのメディアをそれぞれ示す．性能は，実行時

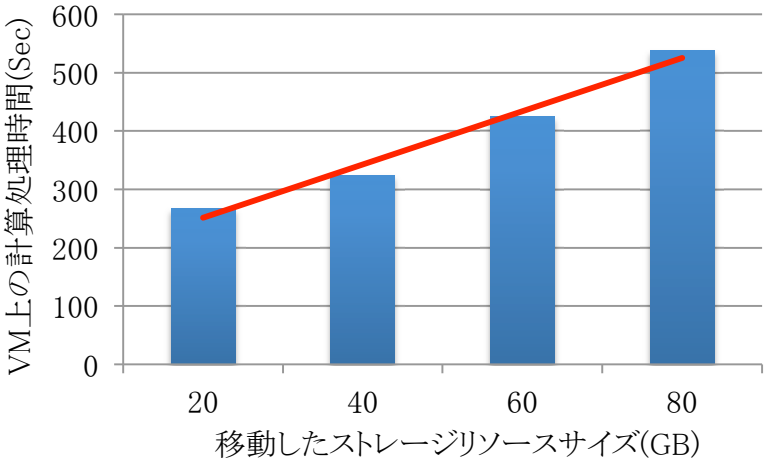

図 5.5: 移動データのサイズが VM に与える影響

間の最も短かった SSD(Local) を基準 (1.0) とし，メディア毎での実行時間を比率で示し，コストは，$/GBytes の最も安価であった SAS(Local) を基準 (1.0) とし，それぞれ比率で示す．表 5.3 を Algorithm1(24 行目) で利用するデータとして提案方式の試作システムに適用する．

データ移動中の VM 上の計算処理への影響度

本測定では，移動データのサイズ，異種メディア間のデータ移動がそれぞれ VM 上の計算処理に与える影響について測定した．

(1) 移動データのサイズが与える影響

最初の測定では，移動するデータのサイズが VM 上の計算処理に与える影響の傾向を測定すべく 20，40，60，80Gbytes のデータを用意し測定を行った．測定では，VM 上で測定プログラムを実行中に，20-80Gbytes のデータ 1 個を SSD(Local) から SAS(Local) へ移動させ，測定プログラムの実行時間について測定した．測定結果を図 5.5 に示す．本結果より，移動するデータのサイズと VM 上の計算処理の時間の関係は，単調増加であることが判明した．

(2) 異種メディア間のデータ移動が与える影響

表 5.4: データ移動にかかる計算処理への影響度

移動元＼移動先	SSD(Local)	SAS(Local)	SATA(Local)	SSD(Ext)	SAS(Ext)
SSD(Local)	-	1/1	10/30	10/8	10/11
SAS(Local)	10/11	-	10/48	10/11	10/16
SATA(Local)	10/32	10/47	-	10/33	10/48
SSD(Ext)	10/7	10/11	10/34	-	1/1
SAS(Ext)	10/11	10/16	10/53	1/1	-

次の測定では，20GBytes のデータを異種メディア間で 1 個移動させ，VM 上での実行時間を計測した．測定結果を表 5.4 に示す．20Gbytes のデータを SSD(Local) から SAS(Local) への移動した場合の VM 上の実行時間を基準 (1.0) とし，各メディア間の移動にかかる計算処理への影響を示す．

その結果，VM 上の実行時間には，各種メディアの性能が大きく影響し，最も性能の低い SATA(Local) から SAS(Ext) への移動が，実行時間に最も影響があることが判明した．この表 5.4 をデータ適正配置アルゴリズム Algorithm1(25 行目) で利用する管理者知識として提案方式の試作システムに適用する．

さらに，この表 5.4 と図 5.5 の測定結果を使い，管理者の性能要件を満たすか否かの判定を行う．図 5.5 の測定結果より，データのサイズが VM 上の計算処理に与える影響は単調増加であり，1Gbyte あたり 4.6 秒の傾きで増加であることを加味し，表 5.4 の異種メディア間のデータ移動にかかる影響を M，式 5.3 により求めた移動対象データの候補のデータサイズ D より移動中の ResponseTime を求める．さらに，現時点の VM の仮想ディスクの性能 (ResponseTime) に表 5.3 の性能値を乗算し算出した移動後の ResponseTime の予測値 T0 を加算することで，式 5.4 の右辺にて移動中の負荷も加味した ResponseTime の予測値を求める．さらに，管理者からの性能要件の ResponseTime を U とすると，式 5.4 のようになる．

$$U \geq 4.6DM + T0 \tag{5.4}$$

この式 5.4 の条件を満たしつつ，右辺と左辺の差分が最も小さくなる移動データの集合が，最終決定する移動対象のデータとなる．また，右辺の値が同一となる移動対象のデータの集合が複数存在する場合には，いずれの移動対象のデータでも効果は同じと考え，データ適正配置アルゴリズムでは任意の集合 1 つを選択する．

提案方式の試作システムでは，式 5.4 の判定をデータ適正配置アルゴリズム Algorithm1(26 行目) の管理者の性能要件を満たすか否かの判定に適用した．

5.4.3 提案方式の試作システムによる測定

次の測定では，5.3.2 節に示す試作システムに，5.4.2 節の測定結果のデータを管理者知識として用いたデータ適正配置アルゴリズムを適用した試作システムを使い測定を行った．

測定方法は，各 20Gbytes の 10 個の計算処理ノード (DataNode) の VM が各メディアに 2 個ずつ均等に配置されている場合を初期構成とし，管理者の性能要件を初期構成の性能より 2/3 倍，4/3 倍，5/3 倍，2 倍の ResponseTime を指定し，VM 上での測定プログラムの実行時間を計測した．

管理者が入力した性能要件との一致度について測定した結果を図 5.6 に示し，それぞれの測定ケースのデータ配置結果を表 5.5 を示す．表 5.5 の数値は各メディアに搭載されている VM の数を示す．例えば，管理者の性能要求が初期構成より 2/3 倍の ResponseTime の向上を目指す測定 1 の場合，表 5.5 の初期構成と比較し，SAS(Ext) のデータ 2 個が SSD(Ext) へ移動したことが分かる．また，図 5.6 のメディアコストに関しては，表 5.5 の結果から表 5.3 のコスト比を用い算出した結果である．

この図 5.6 と表 5.5 の結果より，本提案方式により，管理者の性能要件を満たしつつメディアコストの低減に成功していることがわかる．管理者が指定した性能要件 (図 5.6 の目標処理時間) と提案方式によるデータ移動後の性能 (図 5.6 の処理時間) は，測定 1 の場合は一致しており，測定 4 の場合で 6.2%の差となった．また，データの移動傾向としては，VM 上の計算処理に影響のないストレージによるデータ移動が優先的に実施されている．そのため，ストレージのメディアの価格差と，仮想サーバのメディアの価格差より，目標処理時間を早くするとメディア

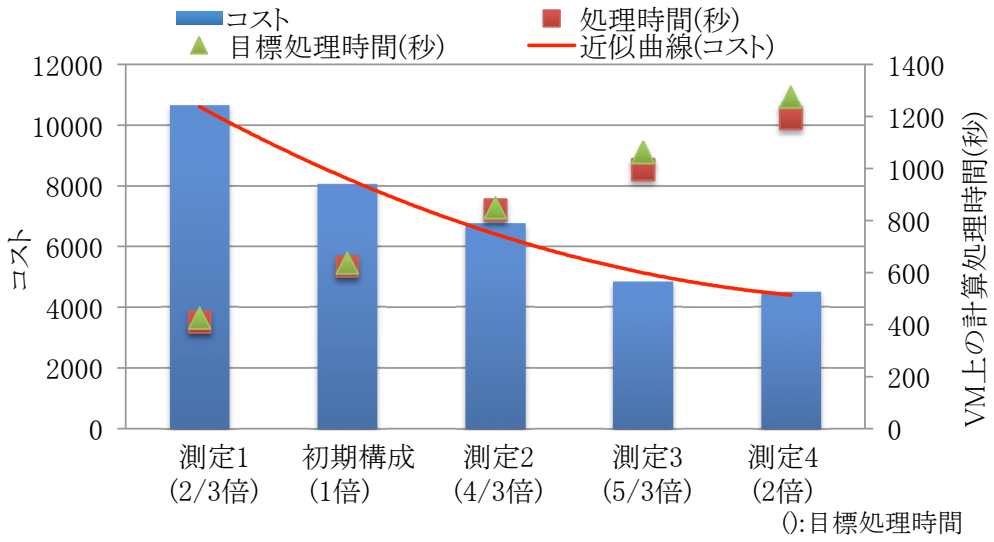

図 5.6: 試作システムによる効果

表 5.5: 提案方式による移動後のデータ配置

測定ケース / 配置先	測定 1	初期構成	測定 2	測定 3	測定 4
SSD(Local)	40G	40G	40G	20G	20G
SAS(Local)	40G	40G	40G	60G	80G
SATA(Local)	40G	40G	40G	40G	40G
SSD(Ext)	80G	40G	20G	0	0
SAS(Ext)	0	40G	60G	80G	60G

コストは高くなり，目標処理時間を遅くするとメディアコストは 2 次関数の削減傾向となった．

5.5 考察

本節では，5.2.2 節で述べた 3 つの問題点，すなわち異なるレイヤのデータ移動技術の矛盾，性能とメディアコストの個別最適および管理者への高負荷について提案方式の有効性を議論する．

5.5.1 データ移動技術の使い分け

本提案方式では，5.3.1 節と 5.3.2 節に示すように，まず仮想サーバとストレージより，性能・構成情報を収集し一元管理行い，本情報をもとに性能ボトルネックの部位を特定し，ボトルネック部位に応じデータ移動技術を使い分けるデータ適正配置アルゴリズムを実現した．

これにより，表 5.5 の測定 3(5/3 倍)，測定 4(2 倍) のようにメディアコストの削減を目指す場合には，仮想サーバ，ストレージともメディアコストを削減する同一のポリシーでのデータ配置を実現できた．これにより，従来技術では，異なるレイヤで異なる判断材料とアルゴリズムでデータ配置を行っていたため発生していたデータ移動技術の矛盾の問題を解決した．

次に，仮想サーバとストレージのデータ移動技術の使い分けに関する有用性を議論する．

プライベートクラウドの大規模データセンターでは，パブリッククラウドや新規のインフラストラクチャ機器を購入せずに，所有する仮想サーバやストレージに複数のサービスを集約することで，設備コストを削減する動きが活発化している．さらには，仮想化技術により，物理機器を新規調達するよりも早くインフラストラクチャ機器を調達できることから，ユーザから，物理機器の調達時に計画していた数よりも多くの新たなサービスの提供が要望されることもある．特に，プライベートクラウドのように利用用途や要件の異なるサービスが複数存在する環境において，インフラストラクチャ機器の特徴を考慮せずに複数のサービスを構

築・集約してしまうと，ユーザが求めるサービスレベルを保つことが難しい．例として，あるサービスはデータ分析用途のようにコスト削減よりも性能を重要視し，I/O 性能を劣化させずより高速なメディアへのデータ移動が求められたり，別のサービスでは VDI(Virtual Desktop Infrastructure) 用途のようにユーザの利用頻度の少ない時間帯であれば，性能が多少劣化してもコスト削減を優先させるデータ移動が求められたり，と様々な要件が混在していることがある．このような場合，表 5.1 に示すようにデータ移動技術の特徴から，前者の要件であればストレージが有するデータ移動技術の利用が良く，後者であれば仮想サーバが有するデータ移動技術を利用するのが良い．しかし，従来のデータ移動技術では，5.2.2 節 (1) に示すように異なるレイヤのデータ移動技術を混在することができなかった．そのため，これらの要件を満たすべく，それぞれの用途別に仮想サーバやストレージを用意し，仮想サーバもしくはストレージのどちらか一方のデータ移動のみ有効化することで対応せざるを得なかった．

これに対し，本提案方式では，5.3.3 節に示すデータ適正配置アルゴリズムにより，異なるレイヤのデータ移動技術の使い分けを実現した．これにより，仮想サーバとストレージの両方のデータ移動技術を有効化しても矛盾したデータ移動を行うことなく一貫したポリシーでデータ移動が可能となった．本提案方式を大規模データセンターに適用する場合，要件の異なるサービスが複数存在する場合であっても，それらサービスがデータを共有していない構成であれば，同一の仮想サーバやストレージで異なる要件を満たすことが可能となる．その結果，より一層のサービスの集約を促進させることができ，設備コスト削減が見込めるようになった．

このように，提案方式は，異なるレイヤのデータ移動技術を使い分け矛盾無くデータ移動を実現したことで，大規模データセンターにおいても設備コストの削減において有用であると言える．

5.5.2 性能とメディアコストのバランス

本提案方式では，5.4.3 節に示すように管理者の性能要件を満たしつつメディアコストの削減を実現した．本節では，メディアコスト，性能面から有用性を議論する．

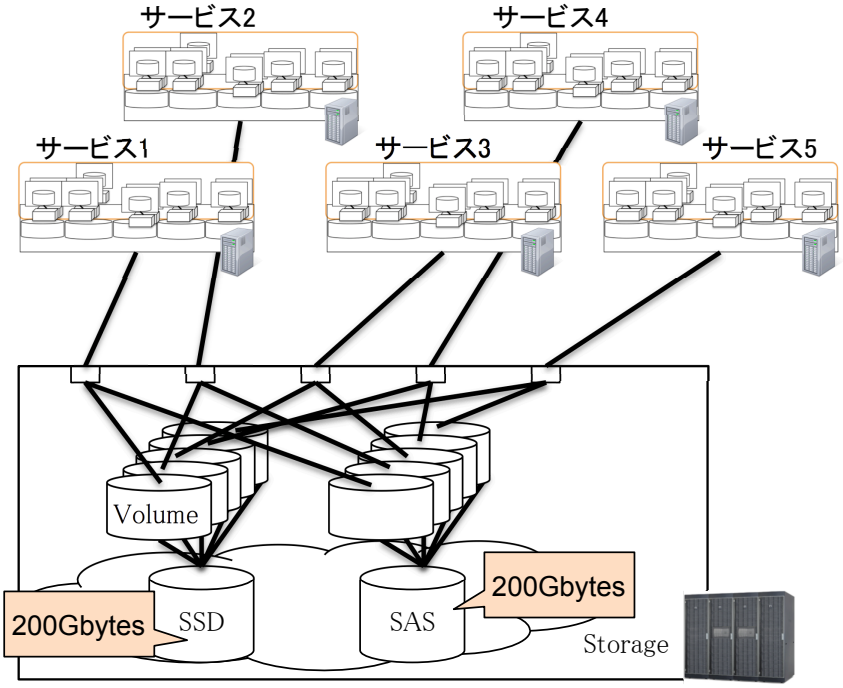

図 5.7: 5 つのサービスの並列実行環境

まず，メディアコスト面について議論する．本測定環境において，本提案方式を使わず性能のみを考慮し，最短時間で計算処理が完了するように全てのデータを SSD(Ext) に置くという単純なデータ配置と仮定した場合，メディアコストは表 5.3 より，19,800 となる．これに対し，管理者の性能要件が 5.4.3 節の測定 1(2/3 倍) のケースでは，全てのデータを SSD(Ext) に置いた場合と比較すると，要件を満たしつつ 47%のメディアコスト削減することができ，測定 4 のケースでは 77%ものメディアコスト削減することができる．

これらの結果より，提案方式がメディアコスト削減に有効であると言える．

次に，性能面について議論する．データセンターでは，ストレージのリソースには容量制限があり，さらに複数の計算処理が同時に実行されるのが一般的である．そこで，メディア毎のストレージ容量を 200Gbytes，5.4.1 節の Hadoop 環境を 1 サービスとし 5 つのサービスが並列実行される環境を想定する．この想定する環境を図 5.7 に示す．

この環境にて，全てのデータを SSD(Ext) に置いて処理する単純なデータ配置ポリシーにて運用した場合，最大ストレージ容量から，1 度に 1 つのサービス (10VM)

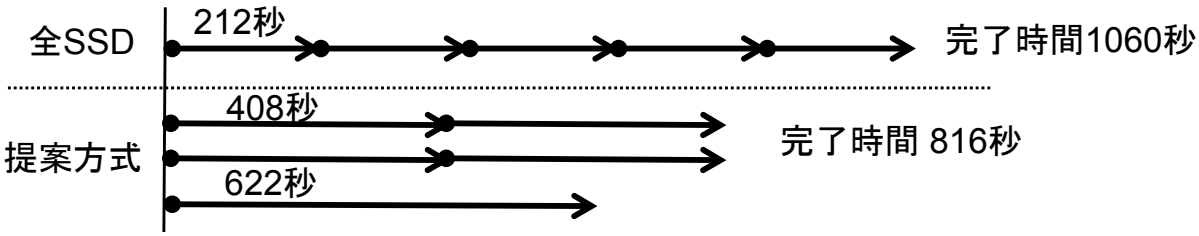

図 5.8: 5 つのサービスを並列実行したケースの実行時間

分のデータしか SSD に配置できず，SSD(Ext) が空くまで他のサービスが待ち状態となり，5 つのサービスをシーケンシャルに実行することになる．これに対し，提案方式では，表 5.5 から，測定 1 と同一構成のプロセスを 2 つ，初期構成と同一構成のプロセスを 1 つ並列に実行できる．その結果，図 5.8 に示すように全てのデータを SSD(Ext) に置いた場合の実行時間 212 秒を，シーケンシャル実行する場合の完了時間 1060 秒にくらべ，提案方式では，測定 1 と同一構成の実行時間 408 秒，測定 2 と同一構成の実行時間 622 秒が並列実行されるため，完了時間は 816 秒となる．全てのデータを SSD(Ext) に置いた場合より提案方式の方がトータルでの性能が向上し，23%の時間が短縮できる．なお，この性能向上は，並列実行される計算処理が増えれば増えるだけ性能改善効果は大きくなる．複数のサービスを提供する大規模データセンターにおいては，より一層の効果が見込める．

このように，本提案方式は，仮想サーバやストレージの個別最適の場合と比べ，仮想サーバやストレージといったシステム全体で判断しデータセンター全体最適となるようにデータ配置を決定することから，メディアコスト面・性能面のいずれの観点においても有効である．

5.5.3 管理者への負荷

本提案方式では，これまで管理者が有していた各種メディアの性能・コスト，仮想サーバやストレージの構成情報といった知識を使い実施していたデータの配置計画に関し，5.3.3 節のデータ適正配置アルゴリズムにて実現した．また，本アルゴリズムの実現にあたり，従来の管理者が製品仕様書や運用実績により知り得ていた知識については，性能・メディアコスト比 (表 5.3)，移動にかかる計算処理へ

の影響度 (表 5.4)，管理者の性能要件を満たすか否かの判定条件 (式 5.4) を測定により求めた．これにより，管理者がユーザの性能要件をサービスの初期構築時に設定するだけで，データセンター全体で一貫性をもった性能・メディアコストを考慮したデータの配置計画を行うデータ適正配置の自動化を実現した．

本提案方式の効果について，5.2.2 節で示した式 5.1, 式 5.2 に当てはめ算出する．本提案方式では，データ配置の計画をアルゴリズム化し自動化を実現したことで，管理者による 1 サービスあたりのデータ配置計画時間 Pt を 0 とすることができる．これは，式 5.2 の Wt が 0 となること意味し，式 5.5 に示すサービスを開始する始めの月の作業時間 (W0) のみとなる．

$$W0 = ItS/A \tag{5.5}$$

これを 2015 年に予測されるストレージが 100 台以上，VM が 50,000 台以上となる大規模なデータセンターでの作業を，5.2.2 節の仮定に加え，1 サービスの初期構築にかかる時間のみ提案方式の管理者の性能要件を設定する時間 5 分を加算した 65 分として算出すると，従来の方法ではサービスを開始する月の作業負荷は 174% であったのに対し，113%に低減し，翌月からの作業負荷は 70% が 0%となる．本提案方式により，サービスを開始する月の作業負荷は 61%，その翌月からの作業負荷は 100%削減の効果がある．

また，サービスを開始する月の作業負荷に関しては，依然 100%を超えているものの，サービスを開始する月のみのため，一時期のみの増員や開始するサービス数を 2ヶ月に分割するなどの対策が可能である．また，1 サービスの初期構築にかかる時間を短縮させる技術として，近年，VM のクローニング技術や一括デプロイ技術が進展しているため，65 分よりも短縮され管理者の負荷が削減されると予測する．

次に．管理者の運用担当の範囲より管理者の負荷について述べる．大規模・複雑化するデータセンターでは，2.2 節で述べたように水平分業管理が行われることが多い．本提案方式では，ストレージに関する情報を知識データ化したことで，要求性能を入力するだけで，ストレージの担当でない他レイヤの管理者でも，ストレージとサーバの両方を考慮したデータ移動の運用ができる目処がついた．また，5.4 節の実験により，本測定環境においては 6.2%以下の要求性能と実性能の性能差であり，各種ストレージの特性を生かしたデータ移動が実現できたと言える．こ

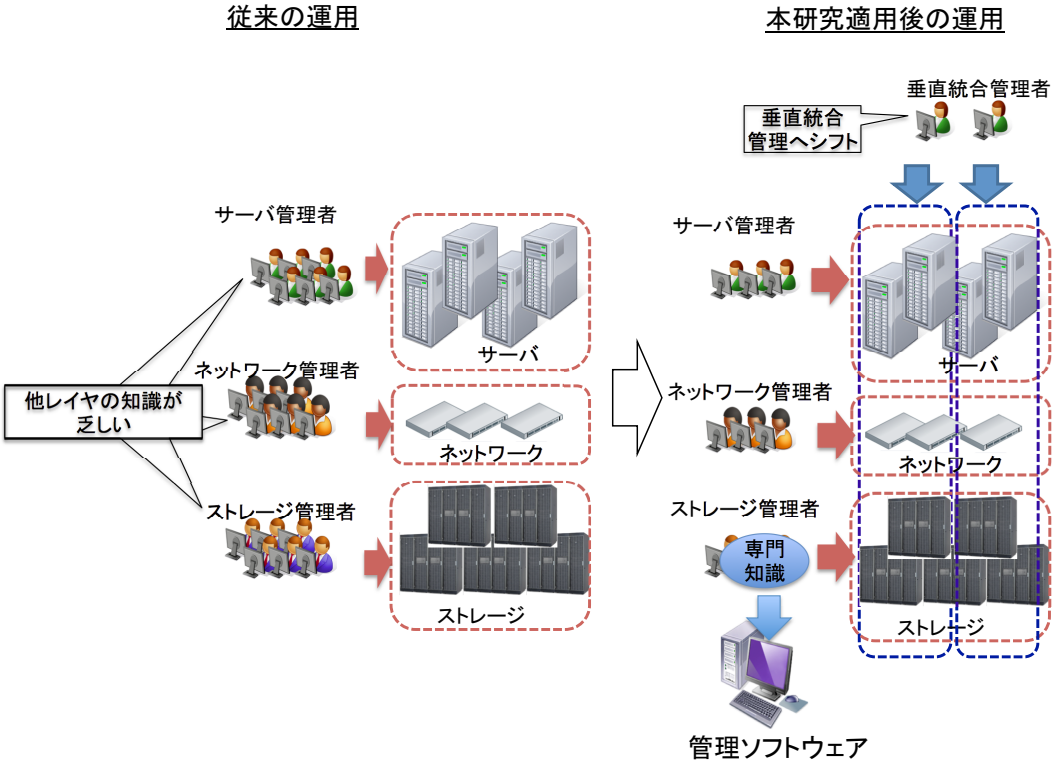

図 5.9: 本研究の適用により進化した運用

れにより，図 5.9 に示すように，サーバ・ネットワーク・ストレージの各レイヤに分かれていた管理者の一部を，データセンター全体で一貫した運用を行うことができる垂直統合管理者へとシフトできる可能性が生まれた．この管理者のシフトは，ストレージ管理者の負荷を軽減に留まらず，サーバ管理者とストレージ管理者が情報交換しつつデータ配置を決めざるをなかった運用から，垂直統合管理者のみで運用できるようになり管理者全体の負荷の削減につながると考える．

このように，本提案方式を大規模・複雑化するデータセンターに適用することで，3.1 節で述べた「ノウハウが属人化している (75.8%)」「特定の管理者に業務が偏りがち (61.4%)」「管理者が多忙で十分な管理ができていない ((53.1%))」の課題解決につながると考える．

5.6 結言

従来のデータ移動技術では，異なるレイヤのデータ移動技術の矛盾，性能とメディアコストの個別最適，管理者への高負荷，という 3 つの問題を抱えていた．そこで，本章では，従来管理者の知識と経験に頼っていたデータ配置の計画をアルゴリズムし，仮想サーバとストレージのデータ移動技術を使い分けることで，ユーザの性能要件を満たしつつデータセンター全体でのメディアコストが最も削減可能なメディアへ配置する自動データ適正配置方式を提案し，問題を解決した．

本提案方式を適用した試作プログラムを開発し測定した結果，2012 年時点のメディアの性能・メディアコストの場合において，管理者の性能要件を満たしつつ，全てのデータを SSD に配置したときに比べ 47%のメディアコストを削減できることを実証した．また，管理者の負荷に関しては，2015 年の大規模データセンターにおける管理者の負荷を 70%削減が見込める試算である．

本章では，2012 年のメディアの性能・コストを使い効果を実証したが，データを保存するメディアの性能・コストは，新メディアの登場によっても変動する．しかし，将来，新メディアが登場することで，2012 年の各種メディアの性能とメディアコストの差が大きくなると，本提案方式の効果はより一層大きくなる．さらに，近年ビックデータの分析が注目されており，I/O 性能がビックデータの高速な分析を実現するために重要な要素となっている状況を考えると，データの保存場所であるメディアの性能・コストに着目した本提案方式の重要度は益々大きくなる．このように，年々大規模化するデータセンターにおいて，特定のレイヤの管理者に依存せず，負担を削減しつつ，性能とメディアコストを適正化したデータの配置を行う本提案方式は有効であり，本提案方式がなければ，仮想サーバやストレージが提供しているデータ移動技術を使いこなし，データセンター内で一貫したデータ配置運用は難しいと考える．

第 6章

進化するデータセンターに追随可能な学習型管理技術

本章では，管理ソフトウェアが把握できないインフラストラクチャ機器の構成情報に対し，ログファイルを使い学習する推論手法を提案する．本提案では，機器構成の推定を実現するとともに，管理者の学習支援を支援することで，データセンターの進化に追随する学習型の管理技術について述べる．

6.1 緒言

大規模・複雑化しているデータセンターのインフラストラクチャ機器に対し，これを運用する管理者の数は一定の傾向にあるだけでなく知識の属人化が進んでいる．さらに，データセンターは仮想化技術の登場により物理的なインフラストラクチャ機器よりも容易に構成変更が行えるようになったことで，短期間での構成変更や未知の機器の導入が加速しただけでなく，構成の複雑化が進んでいる．そのため，管理ソフトウェアで把握できない環境が広がっている．特に障害発生時に，経験の浅い管理者ではインフラストラクチャ機器の把握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得ないケースが多くあった．

そこで，本章では，管理ソフトウェアが把握できないインフラストラクチャ構成においても，構成情報をインフラストラクチャ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．

以下では，6.2 節において関連研究を紹介し，6.3 節にて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，6.4 節にて評価プログラムを用いた実験結果を述べ，最後に 6.5 節で考察を述べる．

6.2 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 [56][55][44][91][92] 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 6.1，構成情報 DB のスキーマの例を図 6.2 に示す．

障害原因解析技術では，次のステップにて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ストラクチャ機器の接続関係・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) よりデータセンターのインフラストラクチャ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) インフラストラクチャ機器から 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，障害原因解析技術では，事前に管理者のノウハウを IF-THEN ルールの形式にて記述しておき，これを実際のインフラストラクチャ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．

また，インフラストラクチャ機器からの構成情報の収集に関しては，SMASH(Systems Management Architecture for Server Hardware)[93] や SMI-S(Storage Management Initiative-Specification)[85] などの標準仕様のインターフェースを利用することに

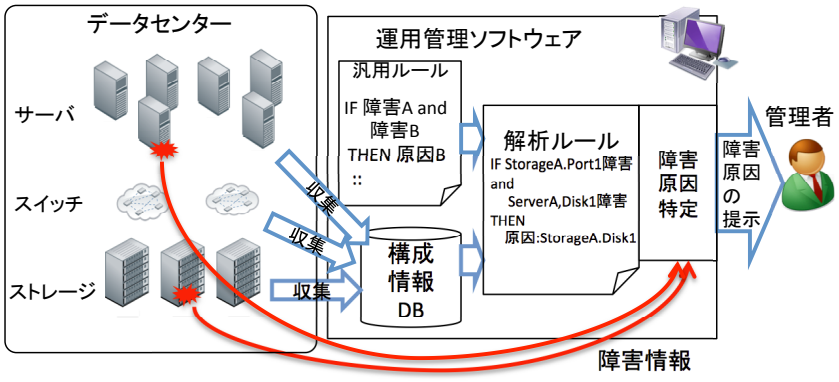

図 6.1: 障害原因解析技術

より，異なるベンダー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，標準仕様のインター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ストラクチャ機器の構成が隠蔽され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いる．

6.3 構成情報の統計的推論方式

6.3.1 方針

ログファイルは，管理者が行ったインフラストラクチャ機器の設定内容や警告・障害メッセージなどインフラストラクチャ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ストラクチャ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とベイズ推定，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ーマシン (SVM) 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

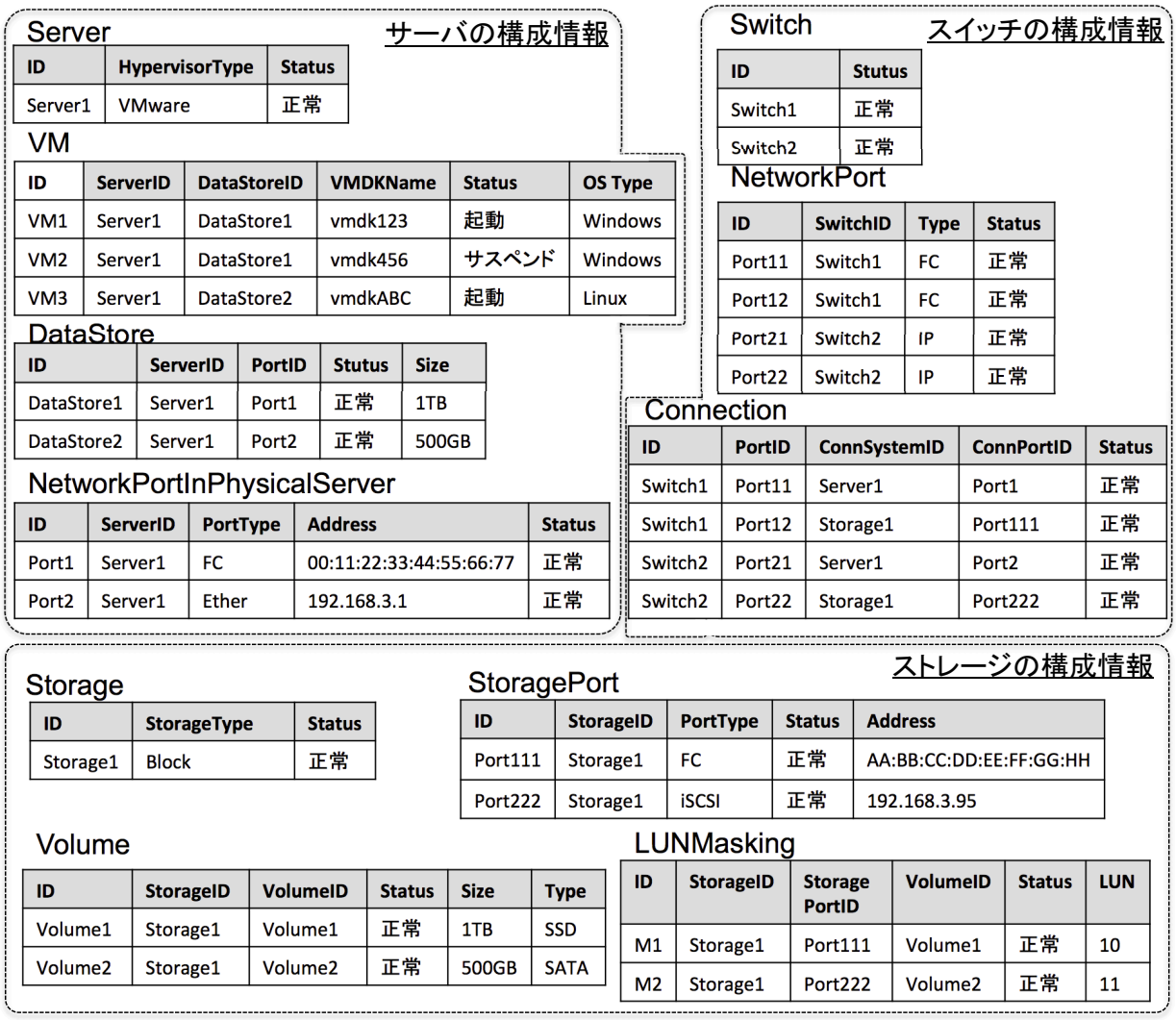

図 6.2: 構成情報 DB のスキーマの例

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，発生確率を用いて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 6.3 に示す．6.2 節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ストラクチャ機器に

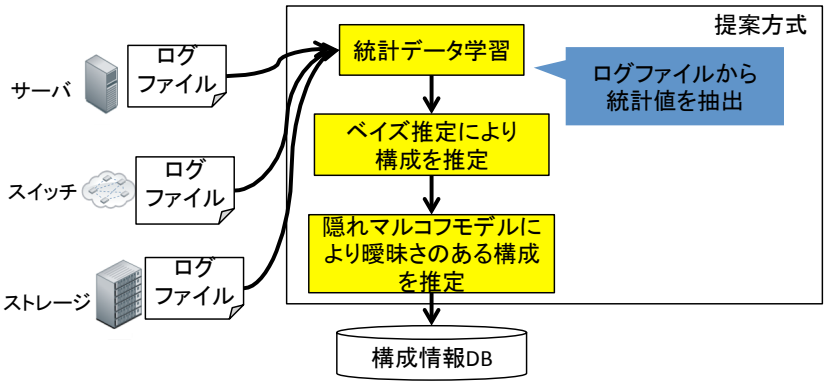

図 6.3: 提案方式による構成情報の推論の流れ

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，6.2 節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．

(1) インフラストラクチャ機器からログファイルを収集

(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

(3) ベイズ推定を使って大まかな構成を把握

(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ストラクチャ機器に対しても，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．

6.3.2 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 [94]．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 [95]．

この二つの推論方式は，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予め定まっている必要がある．

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」と考え適用する．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，観測可能な統計を取るためのデータである．この観測可能なデータとして，あらゆるインフラストラクチャ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ならびに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．

6.3.3 ベイズ推定による構成情報の推定

本節では，ベイズ推定を使った構成情報の推定方式 [96] について述べる．ベイズ推定では，取りうる可能性がある構成が予め判明している必要がある，そこで，取り得る可能性がある構成を求める方法として，DMTF が定めるシステムの標準モデルである CIM[52] 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 6.4 に示す．図 6.4 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，取り得る可能性がある構成を導出する．具体的な例を図 6.5 を用いて説明する．図 6.5 左図のように，Server1 が Switch1 を経由し Storage1 に接続されており，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 6.4 で示されたモデルに当てはめると，図 6.5 右図に示す 4 つのパターンの経路が取り得る可能性がある経路であることが導出できる．また，導

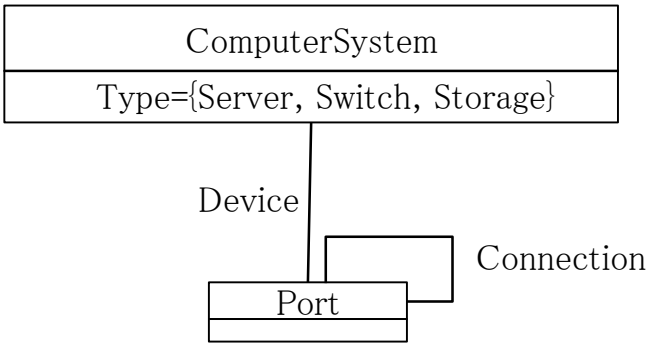

図 6.4: CIM の一例

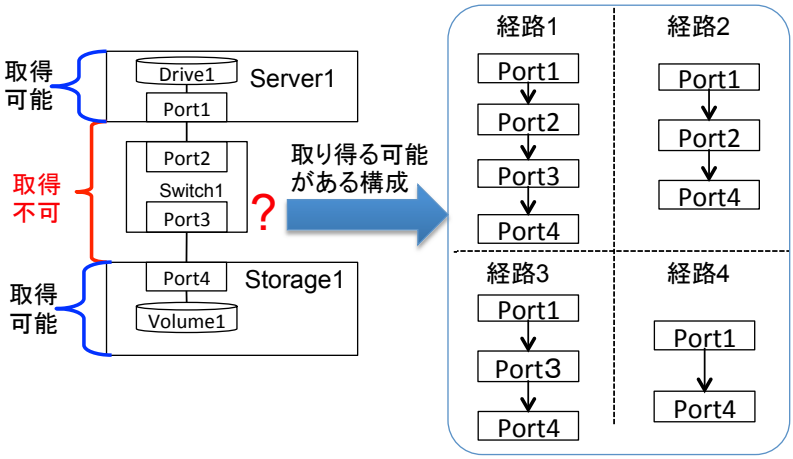

図 6.5: IT システムの構成例

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当てはまると考えると，例えば，Server1 の Port1 が Switch1 の Port2 と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 による取り得る可能生のある構成情報の導出を行った後，各インフラストラクチャ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 6.6 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 6.1 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

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 6.6: ログファイル例

表 6.1: ログファイルから抽出した構成情報と統計情報の例

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%

ルを用い説明しているが，実際の算出では複数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 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 6.1 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 6.5 右図に示す全ての経路に関し，ベイズ推定にて発生確率を算出し，比較を行うことで構成情報を推定する．

6.3.4 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て出現するメッセージの関連性に着目する．一例と

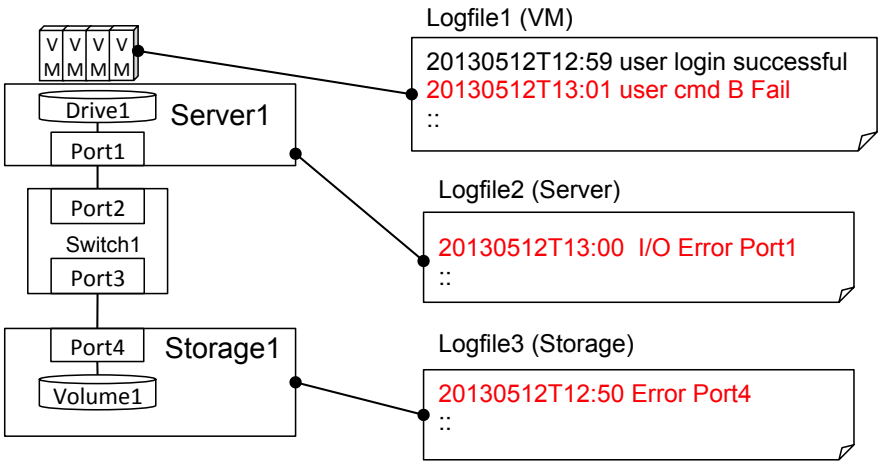

図 6.7: ログファイルのメッセージ例

して図 6.7 にログファイルのメッセージ例を示す．図 6.7 の例では，ストレージのログファイル Logfile3 にて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ストラクチャ機器の間では，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にて示すと，図 6.8 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 $P(Logfile2.\text{“}Port1\text{”}|Logfile3.\text{“}Port4\text{”})$ である．この確率は，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，$P(Logfile2.\text{“}Port1\text{”}|Logfile3.\text{“}Port4\text{”})$ がわかり，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．

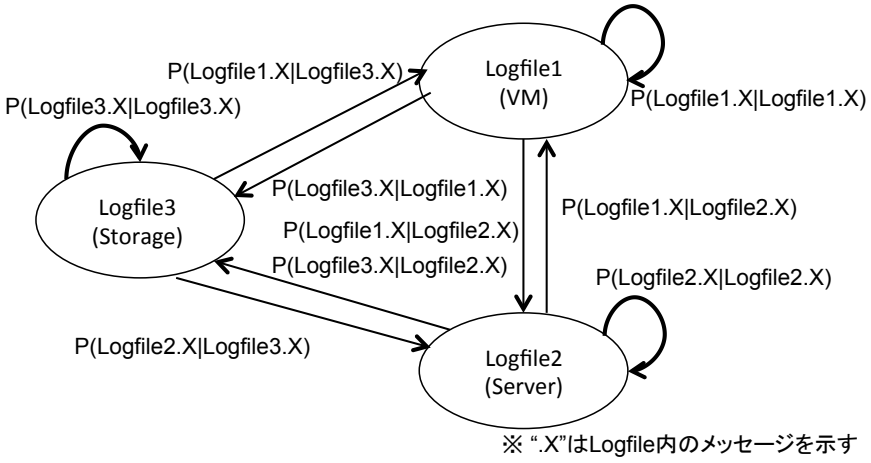

図 6.8: 隠れマルコフモデルによる構成情報の推定方式

6.4 実験

6.4.1 概要

本実験では，各推論方式の特徴を明らかにし，6.3 節にて示した提案方式の有効性を検証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伴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 6.9 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，提案方式にて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 6.9 の例では取りうる接続経路の可能性として 4 パターンある．このうち，実構成の経路である経路 1 を正解の経路とし，推論結果と照らし合わせ正解率を算出する．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

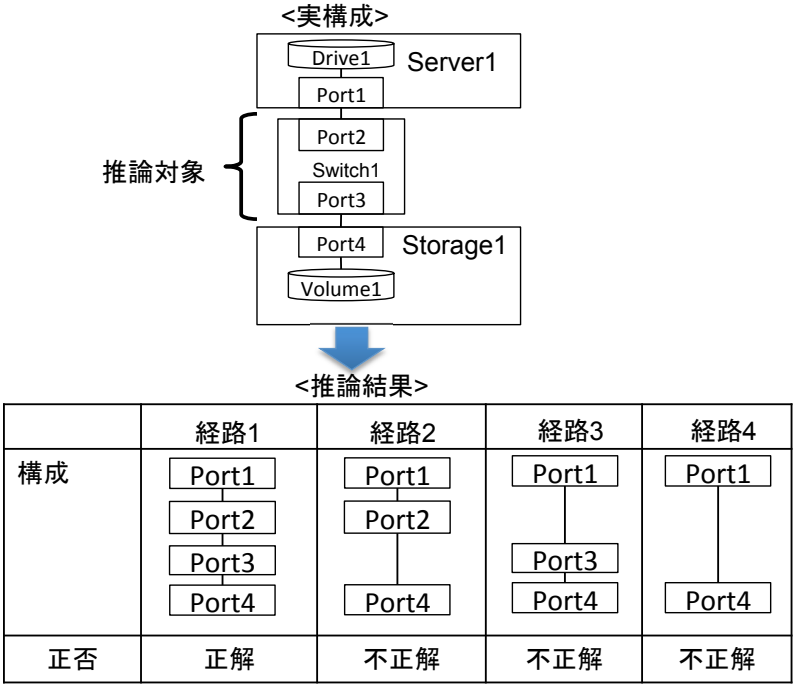

図 6.9: 推論結果の例

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 6.10 の左図のシステム構成を正解の構成とする．正解の構成の一例として Server1 の Port1 → Switch1 の PortA → Switch1 の PortE → Storage1 の Port α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．

6.4.2 評価プログラムと実験環境

図 6.10 と表 6.2 に本提案方式の評価プログラムおよび評価環境を示す．

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

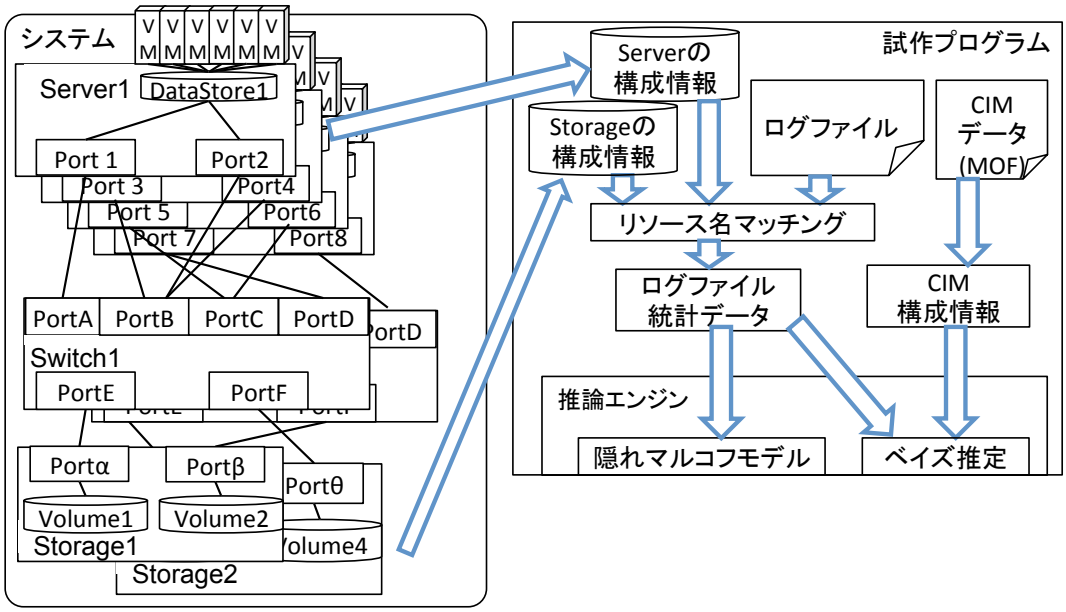

図 6.10: 評価プログラムと評価環境

<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 6.11: ログファイル統計データのフォーマット

バックレベルの設定で運用している．本実験では，図 6.10 左図に示すシステムにて，スイッチの構成が取得できないケースを想定し実験を行った．

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，6.3.2 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 6.10 のリソース名マッチングモジュールにて，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 6.11 にログファイル統計データのフォーマットと例を示す．

表 6.2: 開発・実験環境

物理サーバ	PC x 4 台 (CPU: Intel Corei7 3.4GHz, Memory16GB，Network Port 数 2)
仮想サーバ	VMware ESXi5
VM	24 VM
ストレージ	Hitachi AMS 2100(Network Port 数 4, うち 2Port のみ結線し利用)x 2 台
スイッチ	Brocade SilkWorm 3200(Network Port 数 8, うち 6Port のみ結線し利用) x 2 台
開発環境	MacBook Air(Intel Corei5, Memory 4GB) Mac OS 10.8.4, Java6
構成情報 DB	MySQL5.5.2
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 (Size1.4GB，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(Intel Xeon 2.66GHz (1536Core 中 32Core 使用), Memory12TB(うち 256GB 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

CIM データからの構成情報の抽出では，図 6.10 の CIM 統計モジュールにて，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，推論エンジンにて，ログファイル統計データを用いベイズ推定と隠れマルコフモデルにて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．学習にかかる時間は，表 6.2 の仮想サーバ，ストレージ，スイッチの 16 週間の全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 回の平均を取ったところ，ベイズ推定では 0.8 秒，隠れマルコフモデルでは 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，学習頻度は週や月単位の定期実行，構成変更時に行えば良く，推論時に毎

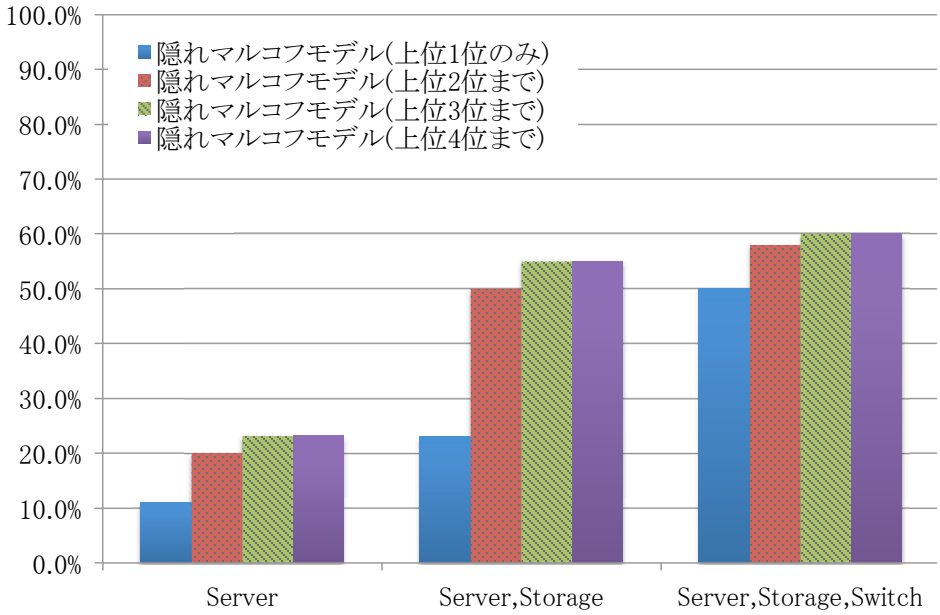

図 6.12: ログファイル種類による正解率 (隠れマルコフモデル)

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共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．

6.4.3 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 4 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 6.12，図 6.13 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%に向上する結果となった．さらに，上位

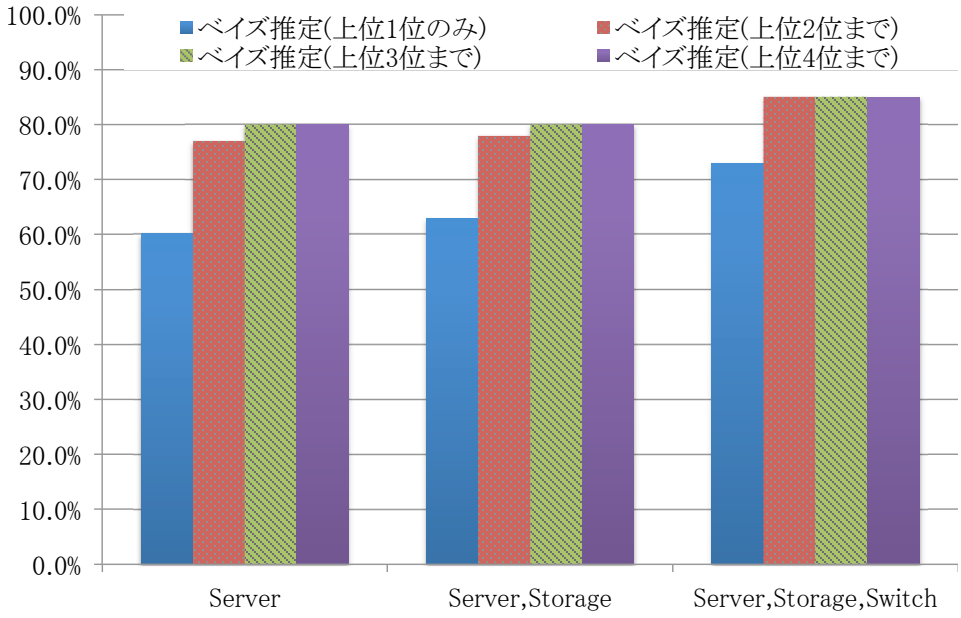

図 6.13: ログファイル種類による正解率 (ベイズ推定)

1 位だけでなく，上位 4 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 16%以上正解率が向上した．上位 3 位と上位 4 位の正解率は隠れマルコフモデル，ベイズ推定ともに同じ正解率であり，上位 3 位で収束していると言える．また，誤って推定された構成の上位は，Server1 の Port1 と Storage1 の Port α を直接接続した構成，Server1 の Port1，Switch1 の PortA と Storage1 の Port α 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．

6.4.4 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用いた．ログファイルの収集期間の変化についての実験結果を図 6.14 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

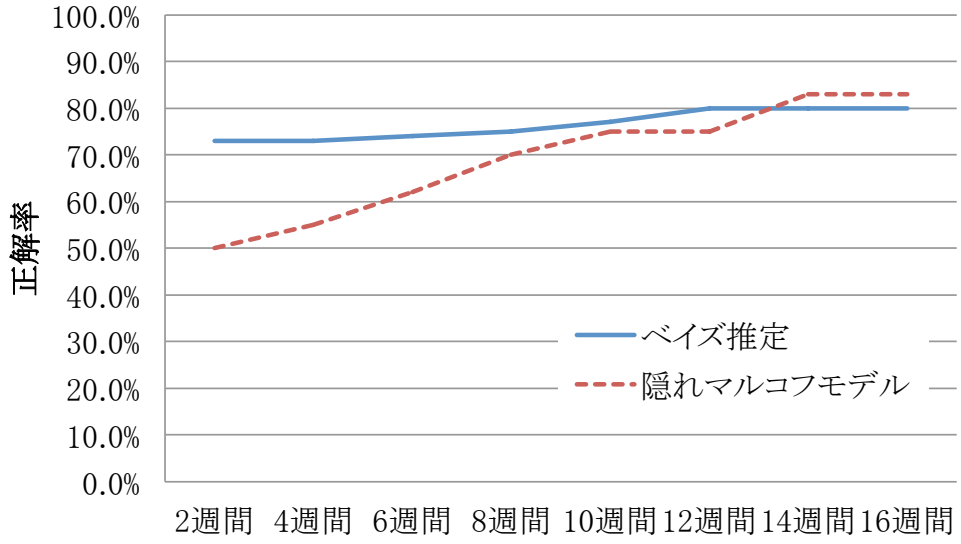

図 6.14: ログファイルの収集期間による正解率

し始める傾向であった．一方，ベイズ推定の正解率は，2 週目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 12%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．

6.4.5 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ー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予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ーを利用するユーザー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

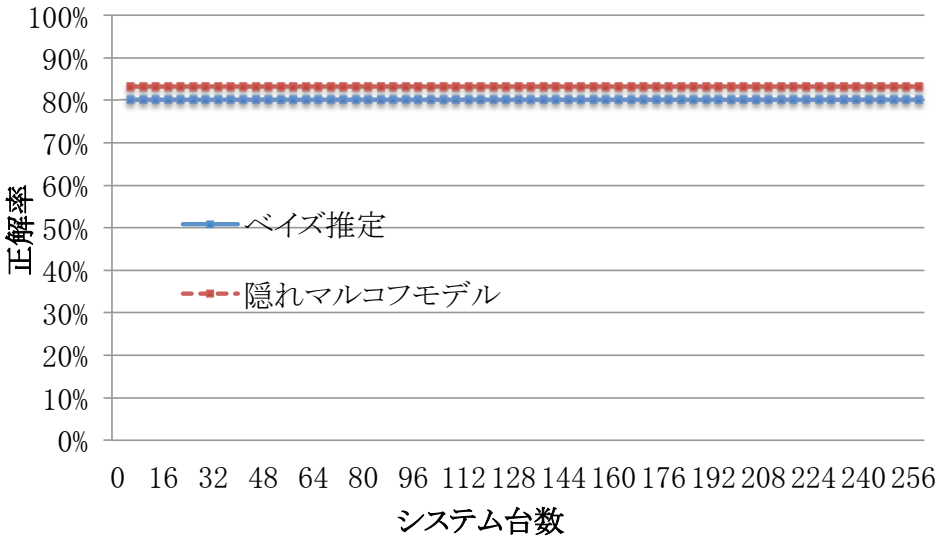

図 6.15: スケールアウトによる正解率

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ープライズのデータセンターに代表されるケースである．このケースでは，CPU や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，6.4.2 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た．本測定では，6.4.2 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一意の値となるように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 6.2 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 6.15 に示す．

本実験の結果，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

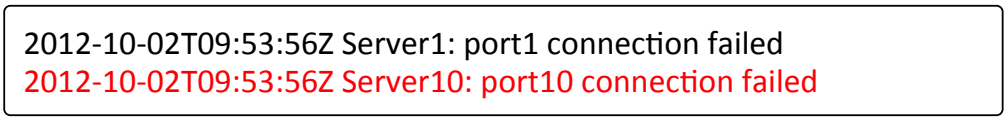

図 6.16: スイッチのログファイル例

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，6.4.2 節の環境での 16 週間分のログファイル使い, 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，6.4.2 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一意の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内の置換前のサーバ名の出現箇所の直後に，サーバ名を置換後のメッセージを追加し，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 6.16 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 6.2 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 6.17 に示す．

この図より，隠れマルコフモデル，ベイズ推定ともに，サーバ数の増加と共に正解率の低下が見られるが，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．

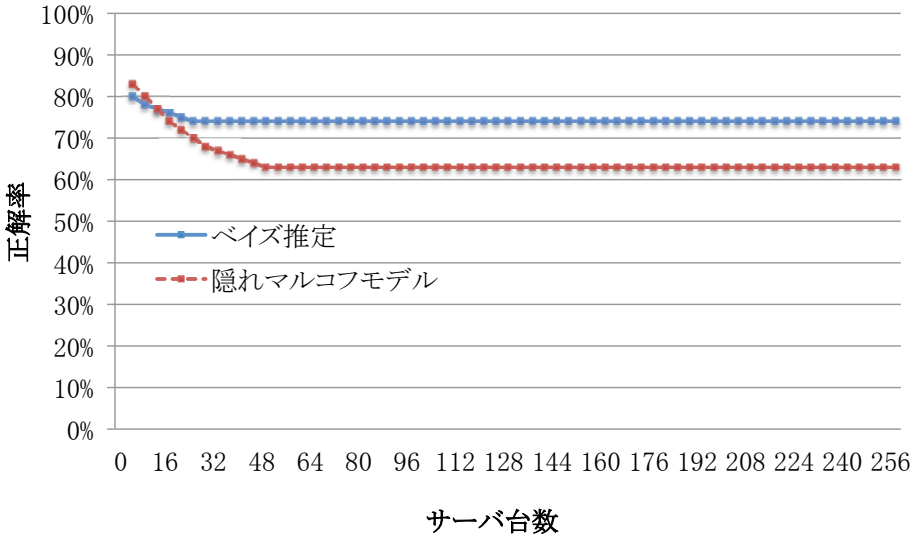

図 6.17: スケールアップよる正解率

6.5 考察

6.5.1 統計的推論の有効性

6.4 節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．6.4 節で行った実験結果を表 6.3 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ストラクチャ機器において，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ストラクチャ機器の構成に対しても，ログファイルが長い期間分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言える．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，ログファイルの収集期間が短い場合でも，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，インフラストラクチャ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言える．

次に，データセンターの特徴の視点から各推論の特徴を図 6.18 にて示す．汎用

表 6.3: 各推論手法の実験結果

推論手法 / 実験項目	隠れマルコフモデル	ベイズ推定
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%

的なインフラストラクチャ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ーフェースの違いなどによりインフラストラクチャ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，ベイズ推定にて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得る一般的な手段が提供されていない未知のインフラストラクチャ構成に対し，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特徴がある．

6.3 節に示す本提案方式では，ベイズ推定にて構成情報を推定した後，長期間のログファイルが取得できている場合に限り隠れマルコフモデルによって推定を行い，その後，隠れマルコフモデルの推論結果とベイズ推定の推論結果の比較を行い確率の高い方を採用した．また，このログファイルの収集期間が，長期間か否かの判断は，本実験の環境においては図 6.14 より隠れマルコフモデルの正解率が，ベイズ推定の正解率を上回り始める 14 週目以降が良いと考える．また，本期間の判断については，ベイズ推定の推定結果と隠れマルコフモデルの推定結果の比較を定期的に行い，推定結果が一致した回数が任意の回数を上回った時点を，ベイズ推定の正解率と隠れマルコフモデルの正解率が交差する点と擬似的にみなす方式も考えられる．このベイズ推定と隠れマルコフモデルを組み合わせる本提案方式は，ログファイルの収集期間が短期間の状況においても，ベイズ推定により 73%の正

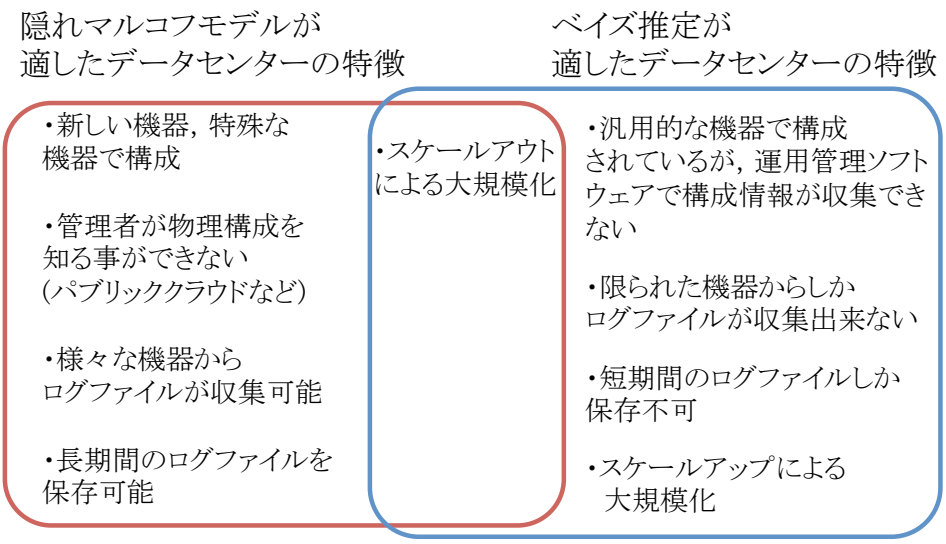

図 6.18: データセンターの特徴に合わせた推論方式

解率で構成情報を推定することができ，長期間のログファイルが収集できるようになれば，隠れマルコフモデルにより 83%まで精度を向上させた推定ができると考える．このように両方の推論手法を組み合わせることで，ログファイルの収集期間に関わらず単独の推論手法で構成情報を推定するよりも高い確率で推定でき，さらには，図 6.18 に示す各推論手法が適したデータセンターの特徴の両方に対応できる方式である．

6.5.2 管理者の知識向上に向けた支援

6.4 節の結果から本提案方式では，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ー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 [36] では，管理者の知識向上のため，熟練管理者と共に行動させ経験させることが有効であると報告されている．しかし一方で，文献 [97] によると，企業の役員が認識するレベルの重要障害は企業あたり年間 1.5 件，事業の中断に至る重大なシステム障害は，企業あたり年間 0.5 件程度で推移している．そのため，経験のない管理者が経験を積

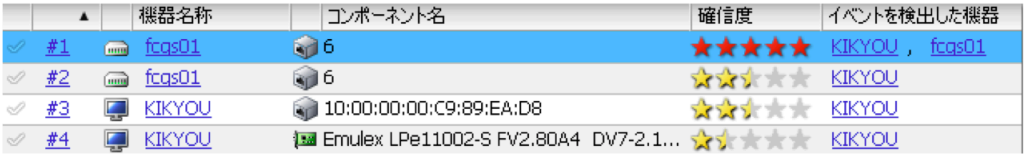

図 6.19: 障害原因解析の表示画面

もうとしても，そもそもの機会が限られている．また，システム障害が発生したとしても，熟練管理者が必ず行動を共にできるとは限らない．このような状況に対し，6.2 節で述べた障害原因解析技術を備えた管理ソフトウェアを使うことで，熟練管理者が行動を共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ができ，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，管理者の知識を向上させる可能性がある．また，障害発生時の状態を管理ソフトウェアにて記録しておくことで，障害発生をシミュレーションすることも可能である．これにより，実際に障害が発生しなくても，擬似的に数多くの経験値を積むことにつながると考える．

障害原因の候補の提示例を，従来研究の障害原因解析の結果表示画面 (図 6.19)[98] 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にて，予め 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．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予め用意しておいた 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合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，5.4 節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．6.4.3 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．そのため，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ーにおいて，管理者の障害原因解析の知識向上を支援できると考える．

6.6 結言

本章では，インフラストラクチャ機器が出力するログファイルを使った統計的推論方式にて構成情報を推定する方式を提案した．多くのデータセンターでは大規模・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抱えている．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組み合わせることで，構成情報が収集できなかったインフラストラクチャ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．さらには，推定結果の効果的な提示と管理ソフトウェアによる障害のシミュレーションにより，障害対応の経験の少ない管理者に対し擬似的に多くの経験を積ませ，知識を向上させる可能性を示唆した．

第 7章

結論

本章では，本研究の全体のまとめと，今後の課題を述べる．

7.1 本研究のまとめ

IT システムの利用が普及し，データセンターが提供する Web サービスやメールサービスなど様々なサービスが社会基盤の一つとなっている．このようなサービスを提供するデータセンターでは大規模・複雑化が進んでいる．一方，データセンターの運用では，運用コストを下げるべく管理者数は一定傾向にあり，管理者の負荷が年々増加してる状況である．さらに，管理者の体系もサーバ，ネットワーク，ストレージと各レイヤにわかれた水平分業管理が進み，管理者の知識が特定機器の知識に偏りがちとなりつつある．そのため，データセンター全体を把握し運用できる技術と経験をもったごく僅かな数の熟練管理者に依存し運用されている状況である．

そこで，本研究では，図 7.1 に示す 3 つのアプローチにより，大規模・複雑化するデータセンターの運用を支援する運用管理技術の高度化について取り組んだ．

まず，始めの研究である 4 章の研究では，管理ソフトウェアによるインフラストラクチャ機器の構成情報の収集時間を 4.6.3 節に示すように 90%短縮し，2020 年に想定される大規模データセンターを一元管理可能とする管理基盤の構築を実現した．さらに，本研究で示した管理ソフトウェアの構成情報収集の並列化方式を使い，構成情報取得を多段に組み合わせるスケールアップ方式の研究 [99][100] を

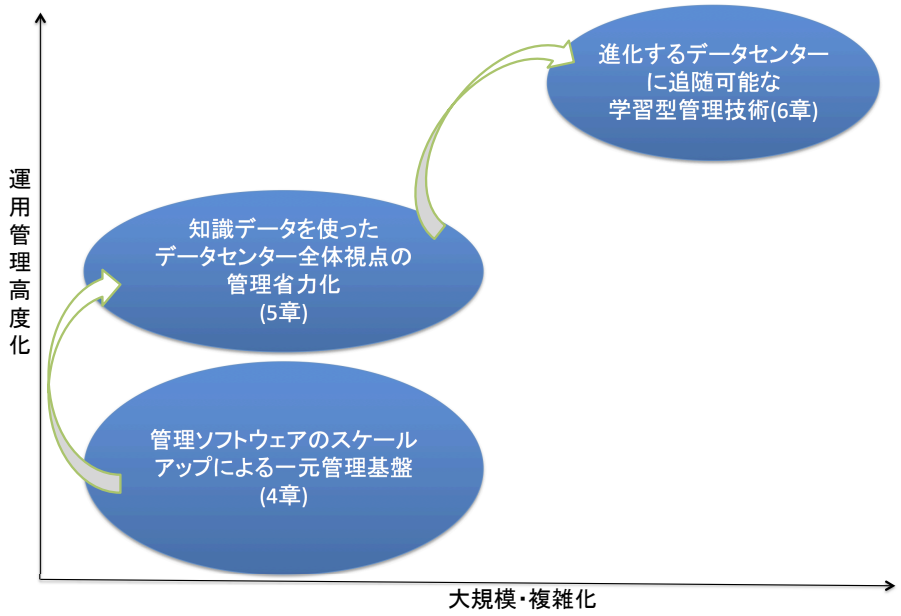

図 7.1: 研究アプローチと位置付け

行った．この研究により，2020 年以降も指数関数的に増加し続けるデータに対し，構成情報取得を複数台多段に設置することで，構成取得の取得は追随できると考える．しかし，構成情報取得を多段が増えれば増えるだけ，取得した構成情報が分散してしまい，収集した構成情報を利用する際の参照性能が劣化する懸念がある．これは，管理ソフトウェアの操作性や 5.3 節に示したデータ適正配置方式や 6.2 節に示した障害原因解析技術など構成情報を参照し利用する運用管理技術の処理性能を劣化させてしまう．そのため，さらなる管理基盤のスケールアップでは，構成情報の取得を多段にするのみだけでなく，構成情報の利用目的に応じた多段方式や取得する構成情報の情報量自身の削減に取り組む必要があると考える．

次に，5 章の研究では，4 章で実現した構成情報をベースとし，ストレージ管理者が持つ専門知識をデータ化することで，5.5.3 項にて示すように定期的に行っていた運用の管理者負荷を削除することに成功した．さらに，ストレージ管理者が持つ専門知識をデータ化したことで，ストレージの担当でない他のレイヤの管理者でも，各種ストレージの性能・コストの特性を生かした運用ができる目処をつけた．これにより，サーバ・ネットワーク・ストレージの各レイヤに分かれていた運用の一部を，データセンター全体で一貫した運用を行うことができる垂直統合

管理者へとシフトが可能できる可能性が生まれた．この研究の提案方式を大規模・複雑化するデータセンターに適用することで，3.1 節で述べた運用の課題「ノウハウが属人化している (75.8%)」「特定の管理者に業務が偏りがち (61.4%)」「管理者が多忙で十分な管理ができていない (53.1%)」の解決につながると考える．また，垂直統合管理者が，熟練管理者と同様にデータセンター全体を見渡し一貫した運用を迅速に実施できるようになることで，定期的な運用にかかる作業時間を減らすこととなり，運用コストの低減につながる．この管理者の負荷の軽減と生み出されたコストは，データセンターが提供する新たなサービスを生み出す一要因となる．

最後の，6 章の研究では，4 章に示す方式だけでは構成情報の把握が難しい規格外の仕様のインフラストラクチャ機器に対し，6.5 節に示すように，ログファイルを使った推論手法を用い構成情報を 83%の正解率で構成情報を推定できる方式を実現した．本提案方式の構成情報の推定は，障害原因解析技術への適用を試み，これまでは構成情報が取得できない機器には適用できなかった障害原因解析技術の適用範囲の拡大を実現した．さらには，データセンター全体のインフラストラクチャ機器の知識を有する熟練管理者に作業が偏りがちであった障害発生の対応について，推定結果の効果的な提示と管理ソフトウェアによる障害のシミュレーションにより，障害対応の経験の少ない管理者に擬似的に多くの経験を積ませ，知識を向上させる可能性を示唆した．これにより，3.1 節で述べた運用の課題「人材育成ができていない (55.0%)」の解決につながると考える．5 章の研究で行った定期的な作業の削減に加え，この管理者の知識向上を行うことで，障害発生時のように不定期に突如発生する作業に関しても，管理者の知識を向上させることで，熟練管理者に負荷が集中せず対応することが可能になると考える．

このように，データセンターが抱える運用の課題解決に向け，大規模・複雑化するデータセンターに追随できる管理技術の高度化と，これを運用する管理者の知識の向上を支える運用管理技術を実現し，少数の管理者でも運用が破綻することなく 24 時間 365 日無停止・安定稼働のデータセンターの運用を実現する礎を築いた．

7.2 今後の課題

今後のプライベートクラウドのデータセンターでは，運用対象が従来のデータセンターが備えるサーバ・ネットワーク・ストレージなどのインフラストラクチャ機器だけに留まらず，これまで管理者が経験したことのない機器にまで拡大することが予見される．このような動向として，個人所有のスマートフォンやタブレットなどを使い，企業や大学のデータセンターのリソースにアクセスする BYOD(Bring your own device)[101][102] や，道路や建物，車などあらゆるところに設置されたセンサーなどの「もの」をインターネットに接続し，新たな価値を生み出そうとする試みである IoT(Internet of Things)[103][104][105] が登場してきている．

今後の課題は，BYOD や IoT の普及を睨み，これまでの管理者が経験したことのないインフラストラクチャ機器を備えるデータセンターに対し，さらなる運用管理技術の高度化を目指す．具体的な課題を 3 つ以下に示す．

1. BYOD や IoT の利用目的を考慮した構成情報の多段方式と情報量の削減より，指数関数的なデータ増加に追随できる管理基盤を実現すること．

2. これまで管理者が自身の知り得る知識から人手で作成を行っていた 6.6 節で述べた障害原因解析技術の解析ルールや 5.3 節で述べたインフラストラクチャ機器の設定に関する知識のデータ化に留まらず，各インフラストラクチャ機器が出力するログファイルや設定ファイルの解析を PCFG(Probabilistic Context-Free Grammar)[106] などを用い，管理者が気がつかない知識までもデータ化することで，さらなる管理省力化を実現すること．

3. BYOD や IoT の普及により，一人の管理者の知り得る知識だけでは導くことのできない未知の障害が予見されるため，複数の管理者の知識共有により，未知の障害を事前に防ぐことのできる管理者集団への進化を支援する運用管理技術を確立すること．

謝辞

本研究の全過程を通じ，懇切なるご指導とご鞭撻を賜わりました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 敷田幹文教授に心から感謝申し上げます.

情報科学研究科博士課程において，情報工学全般に関しまして親切なるご指導とご助言を賜るとともに，本研究をまとめるにあたり貴重なお時間を割いて頂きご教示頂いた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 篠田陽一教授，情報科学研究科 東条敏教授に深く感謝致します.

本研究において，データセンターに関し多大なる情報と親切なるご助言を賜るとともに，貴重なお時間を割いて頂きご教示頂いた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 宇多仁助教，小坂秀一技術専門職員，中野裕晶技術専門職員，上埜元嗣技術専門職員，宮下夏苗主任技術職員に深く感謝致します.

本研究の機会を与えて頂くとともに，暖かいご指導とご鞭撻を賜わりました株式会社 日立製作所 横浜研究所 所長 山足公也博士，前所長 堀田多加志博士，情報プラットフォーム研究センター 小田原宏明 センター長，松並直人 前センター長，主管研究長 岩嵜正明博士，前主管研究長 新井利明博士，運用管理システム研究部 部長 工藤裕博士，主管研究員 細谷睦博士に心から御礼申し上げます.

本研究にあたり，共同研究者として様々なご検討ご助言を頂きました株式会社 日立製作所 横浜研究所 運用管理システム研究部 江丸裕教博士，森村知弘博士，柴山司研究員，名倉正剛博士，中島淳研究員，河野泰隆研究員，三神京子研究員，市川雄二研究員，金子聡研究員，ストレージシステム研究部 牧晋広博士，森宣仁研究員，ならびに株式会社 日立製作所 情報・通信システム社 IT プラットフォーム事業本部 担当本部長 青島達人氏，草間隆人氏，淺野正靖氏，池田博和博士，原純一氏に感謝致します.

また，日頃より多大なるご助言と励ましを頂きました北陸先端科学技術大学院

大学 情報科学研究科 敷田研究室 窪田清氏，Arunee Rarikan 博士，加藤裕氏，増田雄氏，石井香織氏，東家崇裕氏，川井俊輝氏，西野博之氏，石原俊氏，岸克弥氏に感謝致します.

最後に，いつも体調を気遣い暖かく見守ってくれた祖母，父母，妻子に心から感謝致します.

参考文献

[1] IDC. Digital Universe with Research & Analysis, 2014.

[2] 日本総務省. ICT サービスレイヤーのグローバル展開. 平成 25 年度版情報通信白書, 2013.

[3] 島伸行. 運用実態徹底調査 2013 現場が疲弊する理由. 日経 BP システム運用ナレッジ, 2013.

[4] Bhaskar Prasad Rimal, Eunmi Choi, and Ian Lumb. A taxonomy and survey of cloud computing systems. In *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, pp. 44–51. Ieee, 2009.

[5] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, Vol. 53, No. 4, pp. 50–58, 2010.

[6] R. Buyya, Chee Shin Yeo, and S. Venugopal. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. In *High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference on*, pp. 5–13, Sept 2008.

[7] Amazon Web Services Inc. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/jp/ec2/.

[8] Google Inc. Google Cloud Platform. https://cloud.google.com.

[9] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*, Vol. 38, pp. 63–74. ACM, 2008.

[10] Simson L Garfinkel, et al. *Commodity grid computing with Amazon's S3 and EC2*. Defense Technical Information Center, 2007.

[11] Enrico Bocchi, Marco Mellia, and Sofiane Sarni. Cloud Storage Service Benchmarking: Methodologies and Experimentations. In *Proceedings of the 3rd IEEE International Conference on Cloud Networking (IEEE CloudNet 2014)*, No. EPFL-CONF-200923, 2014.

[12] Luiz André Barroso and Urs Hölzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan & Claypool, 2009.

[13] Sherif Sakr, Anna Liu, Daniel M Batista, and Mohammad Alomari. A survey of large scale data management approaches in cloud environments. *Communications Surveys & Tutorials, IEEE*, Vol. 13, No. 3, pp. 311–336, 2011.

[14] Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The Cost of a Cloud: Research Problems in Data Center Networks. *SIGCOMM Comput. Commun. Rev.*, Vol. 39, No. 1, pp. 68–73, 12 2008.

[15] Anton Beloglazov and Rajkumar Buyya. Energy efficient allocation of virtual machines in cloud data centers. In *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pp. 577–578. IEEE, 2010.

[16] Peter Mell and Tim Grance. Effectively and securely using the cloud computing paradigm. *NIST, Information Technology Lab*, 2009.

[17] Balachandra Reddy Kandukuri, V Ramakrishna Paturi, and Atanu Rakshit. Cloud security issues. In *Services Computing, 2009. SCC'09. IEEE International Conference on*, pp. 517–520. IEEE, 2009.

[18] Jianfeng Yang and Zhibin Chen. Cloud computing research and security issues. In *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on*, pp. 1–3. IEEE, 2010.

[19] S Ramgovind, Mariki M Eloff, and E Smith. The management of security in cloud computing. In *Information Security for South Africa (ISSA), 2010*, pp. 1–7. IEEE, 2010.

[20] Gary Anthes. Security in the cloud. *Communications of the ACM*, Vol. 53, No. 11, pp. 16–18, 2010.

[21] Kai Hwang and Deyi Li. Trusted cloud computing with secure resources and data coloring. *Internet Computing, IEEE*, Vol. 14, No. 5, pp. 14–22, 2010.

[22] Dimitrios Zissis and Dimitrios Lekkas. Addressing cloud computing security issues. *Future Generation Computer Systems*, Vol. 28, No. 3, pp. 583–592, 2012.

[23] Borja Sotomayor, Rubén S Montero, Ignacio M Llorente, and Ian Foster. Virtual infrastructure management in private and hybrid clouds. *Internet Computing, IEEE*, Vol. 13, No. 5, pp. 14–22, 2009.

[24] Junjie Peng, Xuejun Zhang, Zhou Lei, Bofeng Zhang, Wu Zhang, and Qing Li. Comparison of several cloud computing platforms. In *Information Science and Engineering (ISISE), 2009 Second International Symposium on*, pp. 23–27. IEEE, 2009.

[25] Jeanna Matthews, Tal Garfinkel, Christofer Hoff, and Jeff Wheeler. Virtual machine contracts for datacenter and cloud computing environments. In *Proceedings of the 1st workshop on Automated control for datacenters and clouds*, pp. 25–30. ACM, 2009.

[26] Tharam Dillon, Chen Wu, and Elizabeth Chang. Cloud computing: issues and challenges. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 27–33. Ieee, 2010.

[27] IDC Japan. 2013 年 国内サーバー市場 ユーザー動向調査. サーバー仮想化環境のワークロード, 2013.

[28] Chuanxiong Guo, Guohan Lu, Helen J Wang, Shuang Yang, Chao Kong, Peng Sun, Wenfei Wu, and Yongguang Zhang. Secondnet: a data center network virtualization architecture with bandwidth guarantees. In *Proceedings of the 6th International COnference*, p. 15. ACM, 2010.

[29] (株) 日立製作所. Hitachi Universal Volume Manager. http://www.hitachi.co.jp/products/it/storage-solutions/products/software/allsofts/index.html?hard=Virtual%20Storage%20Platform%20G1000&soft=Universal%20Volume%20Manager.

[30] (株) 日立製作所. Hitachi Dynamic Provisioning. http://www.hitachi.co.jp/products/it/storage-solutions/products/software/allsofts/index.html?hard=Virtual%20Storage%20Platform&soft=Dynamic%20Provisioning.

[31] (株) 日立製作所. Hitachi Dynamic Tiering. http://www.hitachi.co.jp/products/it/storage-solutions/products/software/function.html#02.

[32] Mikifumi Shikida, Hiroaki Nakano, Shuichi Kozaka, Masato Mato, and Satoshi Uda. A Centralized Storage System with Automated Data Tiering for Private Cloud Environment. *In Proceedings of the 41st annual ACM SIGUCCS conference*, pp. 1–5, 2013.

[33] Timothy Wood, Alexandre Gerber, KK Ramakrishnan, Prashant Shenoy, and Jacobus Van der Merwe. The case for enterprise-ready virtual private clouds. *Usenix HotCloud*, 2009.

[34] Mikifumi Shikida, Kanae Miyashita, Motostugu Ueno, and Satoshi Uda. An Envaluation of Private Cloud System for Desktop Environments. *In*

Proceedings of the 40th annual ACM SIGUCCS conference, pp. 131–134, 2012.

[35] 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.

[36] 技術本部ソフトウェア・エンジニアリング・センター（編）. 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.

[37] Jeffrey O.Kephart and David M.Chess. The Vision of Autonomic Computing. *IEEE Computer Society*, pp. 41–50, 2003.

[38] A. G. Ganek and T. A. Corbi. Toward a New Landscape of Systems Management in an Autonomic Computing Environment. *IBM SYSTEMS JOURNAL*, Vol. 42, No. 1, pp. 5–18, 2003.

[39] Alan G Ganek and Thomas A Corbi. The dawning of the autonomic computing era. *IBM systems Journal*, Vol. 42, No. 1, pp. 5–18, 2003.

[40] Rob Barrett, Paul P Maglio, Eser Kandogan, and John Bailey. Usable autonomic computing systems: the administrator's perspective. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pp. 18–25. IEEE, 2004.

[41] 森一, 敷田幹文. サーバの依存関係を利用したシステム構成管理の支援法. 情報処理学会 分散システム/インターネット運用技術研究会 (DSM), No. 31, pp. 7–12, 2003.

[42] 笠原浩介, 敷田幹文. 大規模ネットワークでのフレキシブルな階層構造によるサーバ監視システムの提案. 情報処理学会 マルチメディア通信と分散処理研究会 (DPS), No. 120, pp. 61–66, 2004.

[43] 森一, 敷田幹文. サーバの依存関係を考慮したシステム構成管理の支援法. 情報処理学会論文誌, Vol. 46, No. 4, pp. 940–948, 2005.

[44] 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.

[45] 奥井裕, 敷田幹文. 大規模サーバにおける部品依存関係の動的抽出方式の提案. 情報処理学会 インターネットと運用技術シンポジウム, pp. 37–42, 2009.

[46] 敷田幹文, 後藤宏志. 大規模サーバ間の部品依存関係に基づくログ管理支援法. 情報処理学会論文誌, Vol. 49, No. 3, pp. 1081–1089, 2008.

[47] Y. Kudo and S. M. Batra. Hitachi IT Operations Analyzer – Topological List View Enhances Ability to Monitor Complex IT Systems. http://itoperations.hds.com/Documents/en/Product%20 Resources/Hitachi%20IT%20Operations%20Analyzer%20Topological%20List.pdf, 2009.

[48] Chengwei Wang, Karsten Schwan, Vanish Talwar, Greg Eisenhauer, Liting Hu, and Matthew Wolf. A flexible architecture integrating monitoring and analytics for managing large-scale data centers. In *Proceedings of the 8th ACM international conference on Autonomic computing*, pp. 141–150. ACM, 2011.

[49] 藤澤恵一朗, 敷田幹文. 大規模システムにおける利用実態を考慮したメンテナンス通知手法の提案. 情報処理学会 第3回インターネットと運用技術シンポジウム, 2010.

[50] 敷田幹文, 藤澤恵一朗. 大規模サーバ間の部品依存関係を利用したユーザ指向通知方式. 情報処理学会論文誌, Vol. 53, No. 3, pp. 978–986, 2012.

[51] DMTF. Distributed Management Task Force. http://www.dmtf.org/.

[52] DMTF. Common Information Model. http://www.dmtf.org/standards/cim/.

[53] DMTF. WBEM. http:/www.dmtf.org/standards/wbem/.

[54] 中村友洋. Web アプリケーションの障害を予測するアクセス時間解析方式の提案. 情報処理学会論文誌. コンピューティングシステム, Vol. 47, No. 12, pp. 349–357, 2006.

[55] 工藤裕, 森村知弘, 菅内公徳, 増石哲也, 薦田憲久. 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.C132(10), pp. 1689–1697, 2012.

[56] 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.

[57] 工藤裕, 森村知弘, 菅内公徳, 薦田憲久. 障害原因解析のためのルール記述方法とその実行方式. 電気学会情報システム研究会 IS-09-71, pp. 1–6, 2009.

[58] Y. Kudo and S. M. Batra. Hitachi IT Operations Analyzer – Root Cause Analysis for Supporting Fault Identification. http://itoperations.hds.com/Documents/en/Product%20 Resources/IT%20Operations%20Analyzer%20Root%20Cause.pdf, 2009.

[59] 佐藤彰洋, 長尾真宏, 小出和秀, 木下哲男, 白鳥則郎. ネットワーク管理におけるイベント発生状況の効率的な把握を実現するイベント分析価値評価手法の提案と評価. 情報処理学会論文誌, Vol. 50, No. 3, pp. 992–1001, 2009.

[60] 宮澤雅典, 西村公佐. サービス品質管理を考慮した障害原因解析手法の提案 (エレメント管理, 管理機能, 理論・運用方法論, 及び一般). 電子情報通信学会技術研究報告. ICM, 情報通信マネジメント, Vol. 110, No. 466, pp. 7–10, 2011.

[61] 加藤裕, 敷田幹文. 障害予測における最適な障害回避手段の提示法. 情報処理学会 第 5 回インターネットと運用技術シンポジウム, 2012.

[62] 野口繁一, 吉瀬謙二, 片桐孝洋, 弓場敏嗣. 不均質なクラスタ環境を対象とするデータ再配置による動的負荷分散機構の設計と実装. 情報処理学会 分散システム/インターネット運用技術研究会 (DSM), No. 38, pp. 109–114, 2006.

[63] 松原正純, 鈴木和宏, 勝野昭. 自律コンピューティングに向けた hpc 向け動的負荷分散機構 (動的負荷分散). 情報処理学会論文誌, Vol. 44, No. 11, pp. 89–100, 2003.

[64] Shinji Kikuchi and Yasuhide Matsumoto. Performance Modeling of Concurrent Live Migration Operations in Cloud Computing System using PRISM Probabilistic Model Checker. *IEEE 4th International Conference on Cloud Computing*, pp. 49–56, 2011.

[65] 広渕崇宏, 小川宏高, 中田秀基, 伊藤智, 関口智嗣. 仮想計算機遠隔ライブマイグレーションのための透過的なストレージ再配置機構. 情処学論, vol. ACS26, pp. 152–165, 2009.

[66] Hikotoshi Nakazato, Manabu Nishio, and Masakatsu Fujiwara. Data allocation method considering server performance and data access frequency with consistent hashing. In *Network Operations and Management Symposium (APNOMS), 2012 14th Asia-Pacific*, pp. 1–8. IEEE, 2012.

[67] 江丸裕教, 高井昌彰. 仮想ボリュームクラスタリング法による動的階層制御ストレージの性能管理. 情報処理学会論文誌, Vol. 52, No. 7, pp. 2234–2244, 2011.

[68] Jeffrey O Kephart and William E Walsh. An artificial intelligence perspective on autonomic computing policies. In *Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on*, pp. 3–12. IEEE, 2004.

[69] Hoi Chan and Thomas Kwok. A Policy-based Management System with Automatic Policy Selection and Creation Capabilities by using a Singular Value Decomposition Technique. *Proceedings of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks*, pp. 99–102, 2006.

[70] G.G Jabbour and D.A Menasee. Autonomic and Autonomous Systems. *ICAS 2008. Fourth International Conference on*, pp. 188–197, 2008.

[71] Chen Xiao-su, Ni Jun, and Wu Jin-Hua. Wireless Communications. *Networking and Mobile Computing 2007. International Conference on,*, pp. 2278–2281, 2007.

[72] Rohit M Lotlikar, Ranga Raju Vatsavai, Mukesh Mohania, and Sharma Chakravarthy. Policy schedule advisor for performance management. In *Autonomic Computing, 2005. ICAC 2005. Proceedings. Second International Conference on*, pp. 183–192. IEEE, 2005.

[73] 工藤裕, 森村知弘, 増岡義政, 薦田憲久. 業務システム動的構成変更のためのポリシー実行スケジューリング方式. 電気学会情報システム研究会 IS-10-74, pp. 5–10, 2010.

[74] Lu Guan, Ying Wang, and Yanfei Li. A dynamic resource allocation method in IaaS based on deadline time. In *Network Operations and Management Symposium (APNOMS), 2012 14th Asia-Pacific*, pp. 1–4. IEEE, 2012.

[75] IT エンタープライズ. Amazon，EC2 の大規模障害について正式に謝罪. IT media エンタープライズ 2011 年 4 月 30 日掲載, 2011.

[76] 日本経済新聞社. ニフティで障害，顧客企業に影響 装置交換し復旧. 日本経済新聞 2012 年 6 月 8 日掲載, 2012.

[77] 日本経済新聞社. ファーストサーバ障害，深刻化する大規模「データ消失」. 日本経済新聞 2012 年 6 月 8 日掲載, 2012.

[78] ファーストサーバ株式会社第三者調査委員会. 調査報告書 (最終報告書). 2012.

[79] IDC. Worldwide Enterprise Storage System 2009-2013 Forecast Update. 2009.

[80] 宮崎扶美, 兼田泰典, 篠原大輔, 藤田高弘, 古橋亮慈. ストレージシステム管理における CIM/WBEM 適用方式の研究. 情報処理学会全国大会講演論文集, pp. 285–286, 2003.

[81] Ramani Routray, Sandeep Gopisetty, Pallavi Galgali, Amit Modi, and Shripad Nadgowda. iSAN: Storage Area Network Management Modeling Simulation. *IEEE Networking, Architecture, and Storage, 2007. International Conference on*, pp. 199–208, 2007.

[82] (株) 日立製作所. Hitachi Command Suite. http://www.hitachi.co.jp/products/it/storage-solutions/products/software/hsms/index.html.

[83] EMC. EMC Ionix Control Center. http://japan.emc.com/products/family/controlcenter-family.htm.

[84] IBM. Ibm Tivoli Storage Productivity Center. http://www-06.ibm.com/systems/jp/storage/software/productivity/.

[85] SNIA. SMI-S. http://www.snia.org/forums/smi/tech_programs/smis_home/.

[86] SMI Technical Steering Group. Ballot Details: Approve addition of experimental client pull operation support. https://members.snia.org/apps/org/workgroup/techcouncil/tsg_smi/ ballot.php?id=4460.

[87] VMware. Storage Distributed Resource Scheduler(SDRS). http://www.vmware.com/jp/products/datacenter-virtualization/vsphere/vsphere-storage-drs/overview.html.

[88] Anne Holler and Manish Lohani. VMware Storage Distributed Resource Scheduler. In *vmworld 2011*, pp. 1–54, 2011.

[89] Larry Freeman. What's Old is New Again Storage Tiering. In *SNW Spring2012*.
https://www.eiseverywhere.com/ehome/SNWS2012/56399/, 2012.

[90] SNIA Data Protection and Capacity Optimization Product Selection Guideß.
http://www.snia.org/forums/dpco/.

[91] M. Gupta and M Subramanian. Preprocessor Algorithm for Network Management Codebook. *Proc. Workshop on Intrusion Detection and Network Monitoring*, pp. 93–102, 1999.

[92] Wang M, Holub V, and Parsons T. Scalable Run-Time Correlation Engine for Monitoring in a Cloud Computing Environment. *Proc.17th IEEE International Conference and Workshops on Engineering Component-Based Systems*, pp. 29–38, 2010.

[93] DMTF. Systems Management Architecture for Server Hardware.
http://www.dmtf.org/standards/smash/.

[94] 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.

[95] 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.

[96] 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, Vol. 20, No. 32, pp. 1–6, 2013.

[97] 一般社団法人日本情報システム・ユーザー協会 (JUAS). 企業 IT 動向調査 2012.
http://www.juas.or.jp/servey/it12/, 2012.

[98] (株) 日立製作所. Hitachi IT Operations Analyzer.
http://www.hitachi.co.jp/Prod/comp/soft1/itoperations/analyzer/.

[99] 森宣仁, 原純一, 坂下幸徳. 大規模データセンタ向け運用管理ソフトウェアにおける情報取得方式の決定手法. 情報処理学会 DICOMO2012 シンポジウム, No. 1, pp. 1815–1821, 2012.

[100] 森宣仁, 原純一, 坂下幸徳, 牧晋広. 大規模データセンタ向け運用管理ソフトウェアにおける情報取得方式の決定手法. 情報処理学会論文誌, Vol. 54, No. 8, pp. 2071–2078, 2013.

[101] Keith W. Miller, Jeffrey Voas, and George F. Hurlburt. BYOD: Security and Privacy Considerations. *IT Professional*, No. 5, pp. 53–55, 2012.

[102] Cisco. Cisco Study: IT Saying Yes to BYOD. http://newsroom.cisco.com/release/854754 Cisco-Study-IT-Saying-Yes-To-BYOD, 2012.

[103] Kevin Ashton. That 'internet of things' thing. *RFiD Journal*, Vol. 22, pp. 97–114, 2009.

[104] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, Vol. 54, No. 15, pp. 2787–2805, 2010.

[105] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, Vol. 29, No. 7, pp. 1645–1660, 2013.

[106] Springer Berlin Heidelberg, editor. *Basic methods of probabilistic context free grammars*, 1992.

本研究に関する発表論文

A. 学術論文誌

[1] 坂下幸徳, 河野泰隆, 柴山司, 中島淳, 敷田幹文. 大規模データセンタにおけるシステム構成情報の高速収集方式の提案. 情報処理学会論文誌, Vol. 53, No. 3, pp. 969-977, 2012.

[2] 坂下幸徳, 三神京子, 金子聡, 敷田幹文. 仮想環境向け自動データ適正配置方式の提案. 情報処理学会論文誌, Vol. 54, No. 3, pp. 1131-1140, 2013.

[3] 森宣仁, 原純一, 坂下幸徳, 牧晋広. 大規模データセンタ向け運用管理ソフトウェアにおける情報取得方式の決定手法. 情報処理学会論文誌, Vol. 54, No. 8, pp. 2071-2078, 2013.

[4] 中島淳, 名倉正剛, 柴山司, 坂下幸徳. 仮想化機構を利用する大規模サービス基盤のためのストレージ設定高速化機構. 情報処理学会論文誌, Vol. 56, No. 2, pp. 1-13, 2015(掲載予定).

[5] 坂下幸徳, 東条敏, 敷田幹文. 障害原因解析における構成情報の統計的推論方式. 情報処理学会論文誌, Vol. 56, No. 3, pp. 1-10, 2015(掲載予定).

B. 国際会議

[1] Yukinori Sakashita, Yutaka Kudo, Masataka Nagura, and Takato Kusama. IT Resource Management Technology for Reducing Operating Costs of Large Cloud Data Centers. Hitachi Review, Vol. 61, No. 6, pp. 279-283, 2012.

[2] Yukinori Sakashita, Kanae Miyashita, Shuichi Kozaka, Satoshi Uda, and Mikifumi Shikida. Simulation of power saving in private cloud environment. In Proceedings of the 41th annual ACM SIGUCCS conference, pp. 1-5, 2014.

C. 学会講演 (査読あり)

[1] 坂下幸徳, 河野泰隆, 柴山司, 中島淳, 敷田幹文. ストレージ管理標準仕様を用いた大規模環境向け構成情報収集方式の提案. 情報処理学会インターネットと運用技術シンポジウム, pp. 67-74, 2010.

[2] 坂下幸徳, 工藤裕, 名倉正剛, 草間隆人. 大規模クラウドデータセンターの運用管理コスト削減を可能とする IT リソース管理技術. 日立評論, pp. 1-4, 2012.

[3] 河野泰隆, 坂下幸徳. ユースケースと IT リソース間の関連情報に着目したシステム構成情報の収集方式. 情報処理学会 DICOMO2012 シンポジウム, No. 1, pp. 1838-1846, 2012.

[4] 森宣仁, 原純一, 坂下幸徳. 大規模データセンタ向け運用管理ソフトウェアにおける情報取得方式の決定手法. 情報処理学会 DICOMO2012 シンポジウム, No. 1, pp. 1815-1821, 2012.

[5] 金子聡, 中島淳, 坂下幸徳. 大規模クラウドにおける IaaS 向け VM 作成手順削減方式. 情報処理学会インターネットと運用技術シンポジウム, pp. 102-109, 2012.

[6] 西野博之, 坂下幸徳, 敷田幹文. 大規模データセンターにおける運用ノウハウ共有による障害再発防止方式の提案. 情報処理学会 第 6 回インターネットと 運用技術シンポジウム, pp. 87-94, 2013.

D. 学会講演 (査読なし)

[1] 宮崎扶美, 田口雄一, 佐藤雅英, 篠原大輔, 兼田泰典, 坂下幸徳. 仮想ストレージ向け性能管理方法の提案. 情報科学技術フォーラム一般講演論文集, Vol. 5, No. 1, pp. 187-188, 2006.

[2] 柴山司, 篠原大輔, 坂下幸徳, 小野卓也, 守島浩. SMI-S 準拠コピーサービ スの実現方式. 情報科学技術フォーラム一般講演論文集, Vol. 5, No. 1, pp. 189-190, 2006.

[3] 金子聡, 河野泰隆, 坂下幸徳. 異種仮想サーバ混在環境向け構成情報の統一管理方式の提案. 情報処理学会研究報告インターネットと運用技術, Vol. 16, No. 44, pp. 1-6, 2012.

[4] 市川雄二郎, 坂下幸徳, 名倉正剛. ハードウェア仮想化環境向けウォッチドックタイマによる障害復旧方式の提案. 電子情報通信学会情報通信マネジメント研究会報告書, Vol. 112, No. 492, pp. 53-58, 2013.

[5] 中島淳, 柴山司, 坂下幸徳, 名倉正剛, 篠原大輔. 複数リクエストのキューイング方式によるストレージ設定の高速化機構. 電子情報通信学会情報通信マネジメント研究会報告書, Vol. 112, No. 492, pp. 83-88, 2013.

[6] 柴山司, 名倉正剛, 坂下幸徳. システム運用管理向け大規模構成管理リポジトリ更新高速化手法の提案. 情報処理学会研究報告インターネットと運用技術, Vol. 20, No. 31, pp. 1-6, 2013.

[7] 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, Vol. 20, No. 32, pp. 1-6, 2013.