修 士 論 文

サーバの高信頼化のための
動作温度を低下させる負荷分散法

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

大和 良介

2014 年 3 月

修 士 論 文

サーバの高信頼化のための
動作温度を低下させる負荷分散法

指導教員　井口 寧 教授

審査委員主査　井口 寧 教授
審査委員　田中 清史 准教授
審査委員　金子 峰雄 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

1210056 大和 良介

提出年月: 2014 年 2 月

概 要

近年，クラウドコンピューティングやグリッドコンピューティングなど，サーバの処理能力を限界まで使う技術が進歩している．しかしながら，サーバを酷使することで高温稼動が続き信頼性が低下する．機械的な見方をすると，信頼性は動作温度と密接に関係している．

タスクを統合して稼動サーバ数を減少させることで消費電力の削減に成功している研究が多く見られるが，CPU 使用率は常に 100%近くで稼働し続けることになる．そうすることによる信頼性の低下を考慮していない．また，具体的にサーバの故障時間を計算している研究は少なく，負荷分散アルゴリズムから一貫した研究も行われていない．

本研究では，動作温度を低下させて信頼性を向上させる．稼動サーバ数を増加させるほどサーバ毎の動作温度を低下させることができるが，消費電力は増大する．そこで，ユーザが求める信頼性の制約条件を満たしながら，低消費電力化のためできるだけ負荷をできるだけ集約して稼動サーバ数を最少にする．後にクラウドへの適用を目指しているため，負荷は仮想マシン単位とする．つまり，サーバへの仮想マシン最適配置問題を考えることになる．メンテナンスが困難であったり，サービスの継続を重視する場合など，使用環境によってはサーバの信頼性に重点を置くことがあることからも提案手法が有効であると考える．

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　背景

近年，クラウドコンピューティングやグリッドコンピューティングなど，サーバの処理能力を限界まで使う技術が進歩している．しかしながら，サーバを酷使することで高温稼動が続き信頼性が低下する．機械的な見方をすると，信頼性は動作温度と密接に関係している．

タスクを統合して稼動サーバ数を減少させることで消費電力の削減に成功している研究が多く見られるが，CPU 使用率は常に 100%近くで稼働し続けることになる．そうすることによる信頼性の低下を考慮していない．また，具体的にサーバの故障時間を計算している研究は少なく，負荷分散アルゴリズムから一貫した研究も行われていない．

1.2　目的と意義

本研究の目的は，サーバの高信頼化である．本研究で提案する動作温度を低下させる負荷分散法を用いて信頼性の向上を目指す．稼動サーバ数を増加させるほどサーバ毎の動作温度を低下させることができるが，消費電力は増大する．そこで，ユーザが求める信頼性の制約条件を満たしながら，低消費電力化のためできるだけ負荷をできるだけ集約して稼動サーバ数を最少にする．後にクラウドへの適用を目指しているため，負荷は仮想マシン単位とする．つまり，サーバへの仮想マシン最適配置問題を考えることになる．メンテナンスが困難であったり，サービスの継続を重視する場合など，使用環境によってはサーバの信頼性に重点を置くことがあることからも提案手法が有効であると考える．

1.3　本論文の構成

第 2 章では，動作温度と電子部品の故障時間から信頼性を考え，サイト消費電力の計算方法を示す．その後，先行研究を紹介する．第 3 章では，提案手法のシステムモデルとアルゴリズムを説明する．第 4 章では，温度と信頼性について，実測データからモデルを作成して，信頼性に関する推計をすることで評価する．第 5 章では，温度と消費電力について，実測データからモデルを作成して，消費電力に関する推計をすることで評価する．第 6 章では，クラウド環境シミュレータ CloudSim を用いて，LOT 負荷分散法の性能評価を

するとともに，クラウドへの適用可能性を検証する．最後に，第 7 章でまとめと今後の課題を述べる．

第2章　動作温度と信頼性及びサイト消費電力

2.1　はじめに

本章では，動作温度の上昇が電子部品の故障時間，つまりは信頼性にどの程度の影響を与えるか，実際のマザーボード上にあるアルミ電解コンデンサを電子部品の例として計算する．データセンタでサーバを運用する際には，サーバ自体の電力だけでなく，空調による電力についても考えなければならない．そこで，空調消費電力の計算方法を示す．最後に，サーバ効率を高めるためタスクの統合や集約配置をしている研究，負荷分散によってサーバ温度の低下を目指している研究を先行研究として紹介する．

2.2　動作温度と信頼性

2.2.1　使用するCPUコア数と温度

図2.1に示すように同じ量の負荷をかけるとき，使用するコア数が増加すると各コアの温度は低下する．左上がコア1つに負荷をかけた場合，右上がコア2つに負荷をかけた場合，左下がコア3つに負荷をかけた場合，右下がコア4つに負荷をかけた場合を示している．色は，温度の高さを示している．左上の濃い赤が最も温度が高く，右下の黄色がこの4つのパターンの中では最も温度が低いことを示している．一方で，使用コア数が増加するほど消費電力は増大する．CPUコア数を例とすると，負荷が集中し過ぎず使用コア数が増え過ぎない，2個もしくは3個の使用コア数が最適となると考えられる．

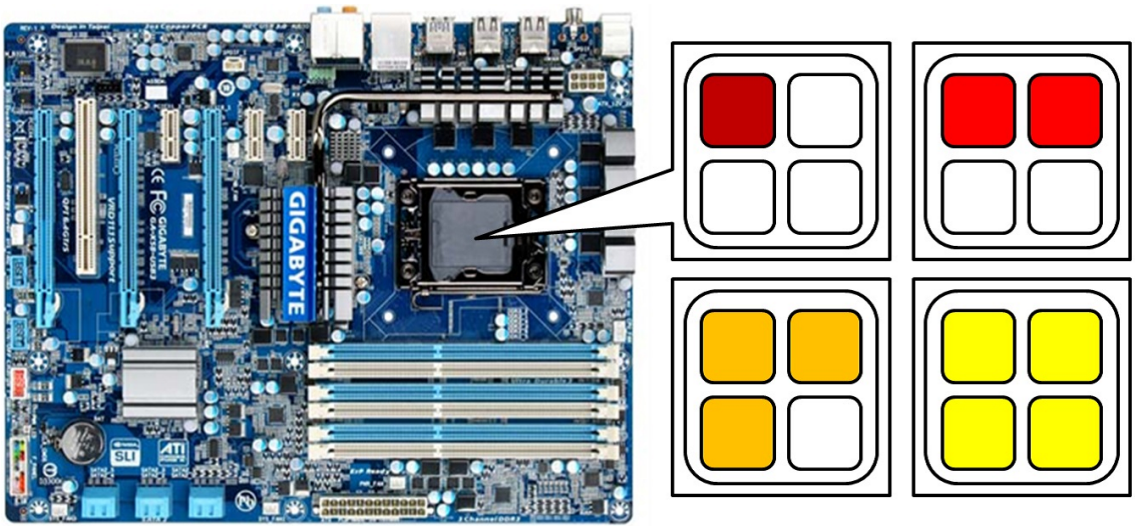

図 2.1: CPU コア数と温度

2.2.2 対象とするアルミ電解コンデンサ

本研究では，電子部品の中でも特に寿命が短く熱に弱いアルミ電解コンデンサに注目する．その中で，実際のマザーボード (GA-X58-USB3/GIGABYTE) 上にある CPU に最も近いコンデンサ (PXE シリーズ/日本ケミコン) について考える．2.2.3 の計算でツールに入力する定格条件と使用条件を表 2.1 と 2.2 に示す．周囲温度と表面温度 T(℃) を変化させる．

表 2.1: アルミ電解コンデンサ定格条件

温度 (℃)	105
許容リプル電流 (A)	5
電圧 (V)	2.5
寿命 (h)	1000

表 2.2: アルミ電解コンデンサ使用条件

周囲温度 (℃)	T
表面温度 (℃)	T
使用リプル電流 (A)	5
自己温度上昇 (℃)	5

2.2.3　動作温度と寿命

表面温度と 0.01%故障時間の関係を図 2.2 に示す．周囲温度と表面温度が上昇すると，0.01%故障時間が短くなる．このグラフは，Panasonic のアルミ電解コンデンサ寿命予測計算ツール [9] を用いた計算結果を基に作成した．計算する際に，周囲温度と排気温度は一致しているとする．このグラフより，ユーザが求める信頼性から温度の制約条件が分かる．本研究では，この 0.01%故障時間を製品の寿命とし，信頼性はこの寿命から決まるとしている．

0.01%故障時間とは，本研究で採用している指標であり，最終的にはこれを用いて製品として機能する平均寿命を算出する．電子部品の寿命試験をしたとき，10000 個に 1 個が故障するまでの時間を 0.01%故障時間としている．一方で，信頼性をあらわす指標として使われる MTBF(平均故障時間) は，稼働時間の和をその間に生じた故障回数で割った値であり，故障を予測する指標ではない．

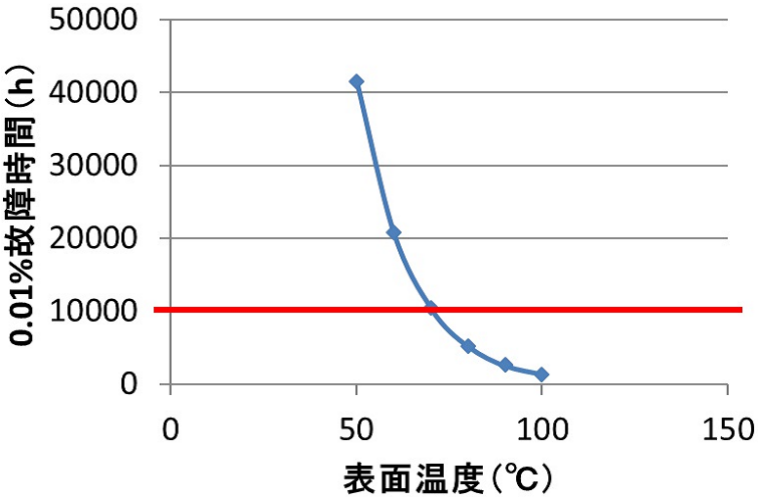

図 2.2: 動作温度と寿命

2.3　サイト消費電力

近年，データセンタの膨大な消費電力が注目されている．アイドル状態のサーバの消費電力が大きいことから，少ない台数のサーバに仮想マシンを集約配置して，使っていないサーバの電源を切る研究が多く行われている．本研究は，それらの研究とは相反する研究であるため，消費電力についても考える必要がある．基本的にデータセンタにおける消費電力とは，サーバを動かすために使われる電力と空調電力の 2 種類である．本研究では稼動サーバ数は増えるものの，サーバの動作温度が低下するため，外部からの侵入熱にかかる空調電力を削減できると考える．

提案手法では，物理マシンの排気温度が低下することで，サーバ室内の設定温度を上昇させることができる．外気と室内温度の差が小さくなり，侵入熱に必要な冷却能力が減少することにより，空調消費電力が削減できる．

例えば，図 2.3 のように，既存手法の排気温度は 46.0 ℃，提案手法の排気温度が 42.4 ℃となることで，既存手法の外気と室内温度の差は $T_{out} - T_{in}$，提案手法の外気と室内温度の差は $T_{out} - (T_{in} + 3.6\,℃)$ となる．T_{out}>T_{in} のとき，$T_{out} - T_{in}$>$T_{out} - (T_{in} + 3.6\,℃)$ であり，侵入熱の計算においてこの温度差 ΔT が係数になっていることから，侵入熱は P_1>P_2 となる．

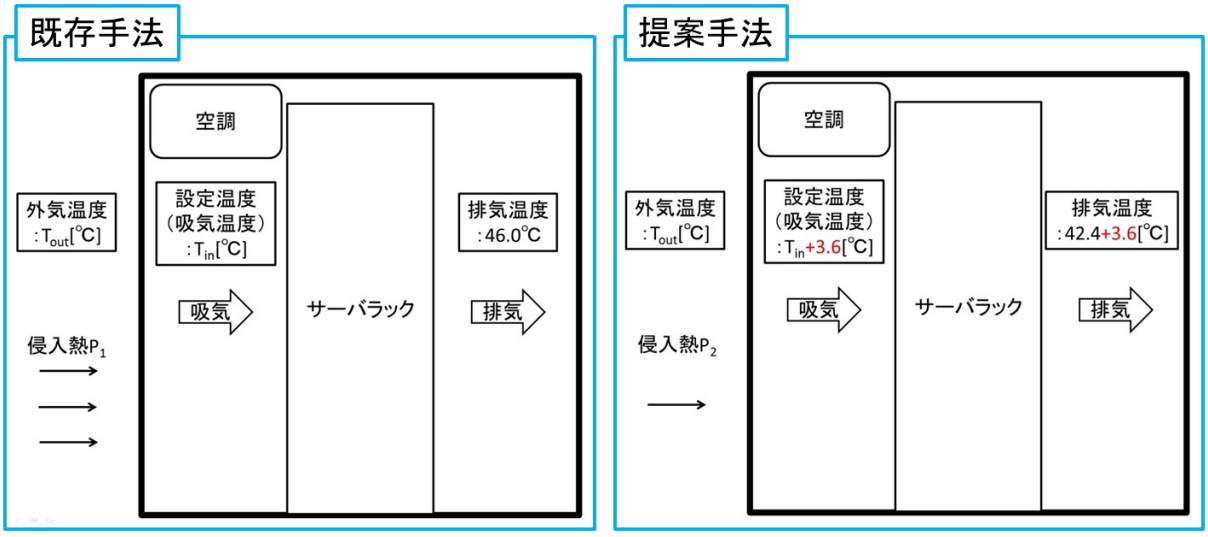

図 2.3: 侵入熱にかかる空調電力削減手法

2.3.1 空調消費電力の計算

空調消費電力の計算手順を示す．サーバ群によって発生する熱量を Q(W)，外部からの侵入熱 P(W) とすると，必要冷却能力 C_c(W) は，

$$C_c = Q + P \tag{2.1}$$

C_c と冷房消費電力 C_p(W) の関係は，

$$C_c \div C_p = COP \tag{2.2}$$

$$\Leftrightarrow C_p = C_c \div COP \tag{2.3}$$

式 (2.3) に式 (2.1) を代入すると，

$$C_p = (Q + P) \div COP \tag{2.4}$$

一方，侵入熱 P(W) は次のように計算できる．壁の熱通過率を U(W/(m^2・K))，部屋の表面積を S(m^2)，最高外気温度と室内設定温度の差を ΔT(K) とすると，

$$P = U \times S \times \Delta T \tag{2.5}$$

式 (2.3) に代入すると，

$$C_p = (Q + U \times S \times \Delta T) \div COP \tag{2.6}$$

提案手法 (C_{p2}) と既存手法 (C_{p1}) の差は，式 (2.7) で求めることができる．

$$C_{p2} - C_{p1} = ((Q_2 + U \times S \times \Delta T_2) - (Q_1 + U \times S \times \Delta T_1)) \div COP \tag{2.7}$$

表 2.3: 計算式の変数まとめ

Q_1(W)	サーバ消費電力合計 (既存)
Q_2(W)	サーバ消費電力合計 (提案)
U(W/(m^2・K))	熱通過率 (コンクリートの壁)
S(m^2)	サーバ室表面積
ΔT_1(℃)	外気温度と室内設定温度の差 (既存)
ΔT_2(℃)	外気温度と室内設定温度の差 (提案)
COP	冷暖房器具のエネルギー消費効率をチェックする係数

2.4 先行研究

2.4.1 Lee らによる方法 [1]

処理能力に余裕があるサーバをないようにタスクを統合して配置するアルゴリズムで稼動サーバ数を減少させ，消費電力の削減に成功しているが，CPU 使用率は常に 100%近くで稼動し続けることになる．そうすることによるサーバの信頼性の低下を考慮していない．

2.4.2 Nathani らによる方法 [2]

オープンソースで仮想マシンベースの管理アーキテクチャHaizea を用いて，プロバイダが提供するリソース割り当てポリシーを満足させているときに，リクエスト拒否率の最少化することを目指している．デッドラインを守ることをポリシーとすれば，様々なサイズ (必要ノード数，処理にかかる時間) のタスク統合させることで稼動サーバ数を減少させ，消費電力の削減に成功しているが，CPU 使用率は常に 100%近くで稼動し続けることになる．そうすることによるサーバの信頼性の低下を考慮していない．

2.4.3 Beloglazov らによる方法 [3]

エネルギー効率の良いクラウドアーキテクチャのフレームワークを定義した後，SLA 違反を抑えてマイグレーション回数を最少化する仮想マシン割り当てアルゴリズムで，消費電力の削減に成功している．また，具体的な今後の展望として，サーバの動作温度を考慮した仮想マシン配置アルゴリズムの構想が挙げられている．

2.4.4 Wang らによる方法 [4]

処理サーバに一定の温度閾値を設けてそれを超えないようにタスクを配置するアルゴリズムで，サーバの稼動温度上昇による信頼性の低下を防ぐとともに，空調電力の削減に成功している．

2.4.5 矢嶋らによる方法 [5]

QoS を考慮した仮想マシンマイグレーションにより，クラウドの低消費電力化に成功している．オリジナル矢嶋法は，QoS が一定の閾値以下にならないように仮想マシンを集約配置するという手法であるが，電力の削減手法としてはサーバに可能な限り仮想マシンを集約することを目指している．本研究ではその考え方を既存手法とし簡略化矢嶋法と呼ぶ．また，この研究では，使用していないサーバの電源ををを全てオフにすることで電力を

削減している．低消費電力家には最も効果的と言えるが，過度なパワーサイクルにより信頼性が下がると予想される．

2.4.6 問題点

[1] と [2] にあるようなタスクの統合等により CPU 使用率を常に限界まで高めてサーバを稼動させる手法では，高温稼動により電子部品の故障率が上昇して，信頼性が低下していると考えられる．それを考えると [4] にあるような稼動温度を考慮した負荷分散アルゴリズムが必要がであると言える．しかしながら，稼動サーバ数を増加させると消費電力が増大することも事実である．

2.5 まとめ

本章では，実際のマザーボード上にあるアルミ電解コンデンサを電子部品の例として故障時間を計算することで，動作温度の上昇が電子部品の故障時間を短くすることを明らかにした．さらに，データセンタでサーバを運用する際の空調消費電力の計算方法を示した．最後に，サーバ効率を高めるたて稼動サーバ数を減少させるためタスクの統合をしている研究，負荷分散によってサーバ温度の低下を目指している研究を先行研究として紹介した．

第3章　LOT負荷分散法(提案手法)

3.1　はじめに

本章では，本研究で提案するLOT(Low Operating Temperature)負荷分散法の説明をする．まず，本研究が想定する環境のシステムモデルを示して，各構成要素とその間のネットワークの説明をする．その後，システムモデルのような環境で仮想マシンをサーバに配置するとき，ユーザが求める信頼性から決定した排気温度閾値を超えないような仮想マシン配置を導くことができる．アルゴリズムの考え方を図を用いて動きを追いながら説明する．アルゴリズムで使われる変数を表にまとめて示す．

3.2　システムモデル

システムモデルを図3.1に示す．外枠はラックで，内枠はサーバをそれぞれ示している．HOSTとは，管理サーバに繋がったサーバを示している．また，TSは温度センサ，VMは仮想マシンを示している．管理サーバと各ホストはネットワークでつながっており，温度や仮想マシンの配置状況が分かるようになっている．排気温度が低くなるように仮想マシンを移動する．

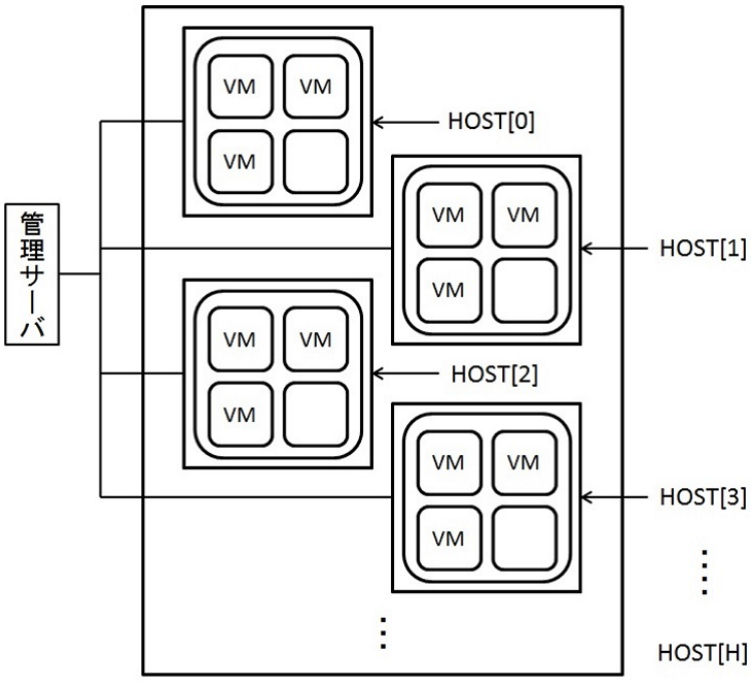

図 3.1: システムモデル

3.3 アルゴリズム

提案アルゴリズムである LOT 負荷分散法では，ユーザが求める信頼性から決定した排気温度閾値 T_{ex}_threshold を超えないような仮想マシン配置を導くことができる．既存アルゴリズムである簡略化矢嶋法では，CPU 使用率 100%を超えないような仮想マシン配置を導くことができる．温度閾値なし．図 3.2 にアルゴリズムの考え方を示す．左図では，VM1 と VM2 を HOST1 に配置して，VM3 と VM4 を HOST2 に配置すると，HOST2 が温度閾値を超えてしまう．一方，右図では，VM1 と VM2 を HOST1 に配置して，VM3 と VM4 を HOST2 に配置すると，HOST2 が温度閾値を超えてしまうので，新たにスタンバイ状態だった HOST3 を立ち上げ，VM3 を HOST2 に，VM4 を HOST3 へ配置することで，温度閾値を超えない仮想マシン配置になっている．

アルゴリズム中の変数及び関数の説明を表 3.1 に示す．CPU_HOST とは，サーバ 1 台あたりの全コアでの表示 CPU 使用率である．CPU_VM とは，仮想マシンを配置した際の CPU_HOST 増加量である．

まず，LOT 負荷分散法のアルゴリズム (Algorithm1) を説明する．入力は，サーバの数と仮想マシンの数と使用率である．出力は，仮想マシン配置となる．3 行目から 10 行目の for 文では，探索する仮想マシンを 1 から V までシフトする．4 行目から 9 行目の for 文では，探索するサーバを 1 から H までシフトする．5 行目から 8 行目の if 文は，CPU_HOST に CPU_VM を足しあわせた値が 100 を超えない，かつ，CPU_HOST に CPU_VM を足し

あわせたときの排気温度が排気温度閾値を超えないならば，仮想マシンをサーバに配置するという内容である．配置できた場合，break する．

次に，簡略化矢嶋法のアルゴリズム (Algorithm2) を説明する．オリジナル矢嶋法は，QoS が一定の閾値以下にならないように仮想マシンを集約配置するという手法であるが，電力の削減手法としてはサーバに可能な限り仮想マシンを集約することを目指している．本研究ではその考え方を既存手法とし簡略化矢嶋法と呼ぶ．入力は，サーバの数と仮想マシンの数と使用率である．出力は，仮想マシン配置となる．3 行目から 10 行目の for 文では，探索する仮想マシンを 1 から V までシフトする．4 行目から 9 行目の for 文では，探索するサーバを 1 から H までシフトする．5 行目から 8 行目の if 文では，CPU_HOST に CPU_VM を足しあわせた値が 100 を超えないならば，仮想マシンをサーバに配置するという内容である．配置できた場合，break する．

LOT 負荷分散法は，仮想マシンの負荷が静的，もしくは，上限が設定されている場合に有効である．また，動的な負荷に対しても仮想マシンの初期配置アルゴリズムとしては有効である．一方，仮想マシンの負荷が動的で再配置が必要になる場合，LOT 負荷分散法はうまく機能しない．

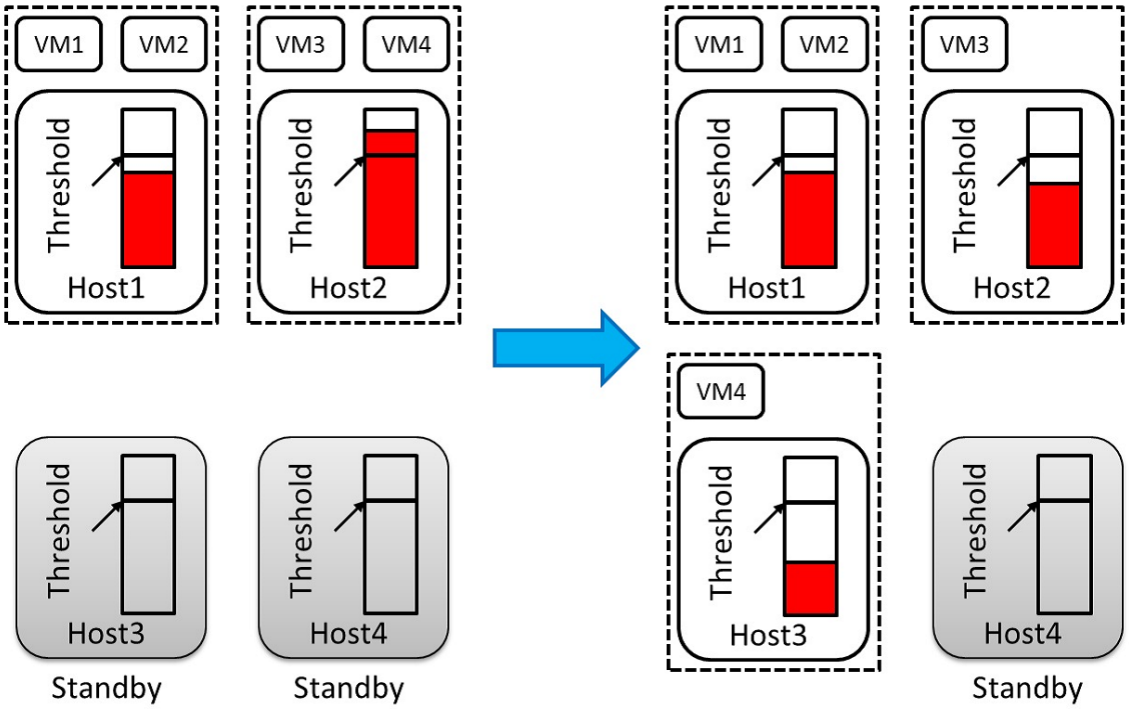

図 3.2: 提案アルゴリズムの考え方

Algorithm 1 LOT 負荷分散法

```
1: Input : number of HOSTs , number and utilization of VMs
2: Output : allocation of VMs
3: for v in 1..V do
4:   for h in 1..H do
5:     if CPU_HOST[h] + CPU_VM[v] ≤ 100 &&
       T_ex(CPU_HOST[h] + CPU_VM[v]) ≤ T_ex_threshold then
6:       assign VM[v] for HOST[h]
7:       break
8:     end if
9:   end for
10: end for
```

Algorithm 2 簡略化矢嶋法

```
1: Input : number of HOSTs , number and utilization of VMs
2: Output : allocation of VMs
3: for v in 1..V do
4:   for h in 1..H do
5:     if CPU_HOST[h] + CPU_VM[v] ≤ 100 then
6:       assign VM[v] for HOST[h]
7:       break
8:     end if
9:   end for
10: end for
```

表 3.1: アルゴリズムの変数まとめ

h	現在探索しているホスト
v	現在探索している仮想マシン
H	ホスト数
V	仮想マシン数
CPU_HOST	サーバ 1 台あたりの全コアでの表示 CPU 使用率
CPU_VM	仮想マシン配置時の CPU_HOST 増加量
T_{ex}(CPU utilization)	CPU 使用率を排気温度に変換する関数
T_ex_threshold	排気温度閾値 (定数)

3.4 まとめ

本章では，本研究で提案アルゴリズムである LOT 負荷分散法と既存アルゴリズムである簡略化矢嶋法の説明をした．本研究が想定する環境のシステムモデルを示して，各構成要素とその間のネットワークの説明をした．その後，システムモデルのような環境で仮想マシンをサーバに配置するとき，ユーザが求める信頼性から決定した排気温度閾値を超えないような仮想マシン配置を導くことができるアルゴリズムを考え方の図を用いて動きを追いながら解説した．アルゴリズムで使われる変数を表にまとめて示した．

第4章　温度と信頼性の評価

4.1　はじめに

本章では，温度と信頼性の評価をする．まず 4.2 では，実験で使った実験サーバ・温度計の仕様と使い方を説明する．その後，それらを組み合わせた温度測定実験の環境構成を示して，データの取得方法を説明する．本研究では仮想化技術を多用するため，仮想化によって発生する負荷を考慮する必要がある．4.4 では，それを明らかにするための予備実験の結果を示す．具体的には，仮想マシン単体と複数で同じ量の負荷をサーバにかけたとき，CPU 使用率と動作温度を比較して評価する実験である．4.5 では，対象とするサーバを決めて実測データより動作温度モデルを作成する．CPU 使用率と動作温度の関係を数式化する．その後，そのモデルを用いて推計を行う．4.6 では，推計のために作成したプログラムの概要，想定する環境や制約条件を示した推計モデルを説明する．その後，信頼性に関する推計結果を示す．

4.2　実験機材

4.2.1　実験サーバ

本研究では，Caplyst 1U Standard-1366/10G を対象として動作温度モデルと消費電力モデルを作成した．また，このマシンには CloudSim 環境を構築してシミュレーションをする際にも使用している．写真と仕様を図 4.1 と表 4.1 に示す．

図 4.1: 実験サーバ

表 4.1: 実験サーバの仕様

CPU	Intel Core i7 950
Memory	DDR3(8G)
OS	Ubuntu12.04(64bit)

4.2.2　温度計

本研究では，温度センサとして TR-701AW を使用した．写真と仕様を図 4.2 と表 4.2 に示す．

図 4.2: 温度センサ

表 4.2: 温度センサの仕様

測定範囲	-40〜110 ℃
精度	± 0.3 ℃
測定分解能	0.1 ℃
チャンネル	温度 2ch

4.3　実測実験の環境構成

4.3.1　温度測定実験の環境構成

T_{ex} を決定するために温度測定実験を行った．環境構成を図 4.3 に示す．実験サーバ・温度センサ・モニタリング PC の 3 つで構成されている．排気温度データは温度センサによって直接取得する．CPU 使用率と内部温度データは，モニタリング PC から実験サーバにリモートログインして取得する．点線は，有線ネットワークを示す．

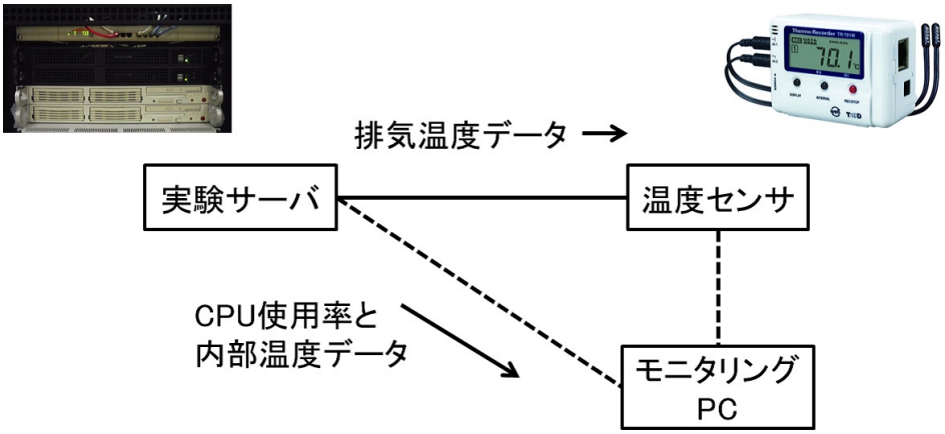

図 4.3: 温度測定実験の環境構成

4.4　予備実験

予備実験では，仮想化において考慮すべき大きな負荷が発生しているかどうかを明らかにする．具体的には，仮想マシン単体と複数で同じ量の負荷をサーバにかけたとき，CPU 使用率と排気温度を比較して評価する．

4.4.1 仮想マシン数と使用率・排気温度

1CPU の仮想マシンの数を変動させたときの CPU 使用率と排気温度の変化を図 4.4 に示す．仮想マシンの数を増加させると，サーバの使用率と排気温度は線形に上昇する．

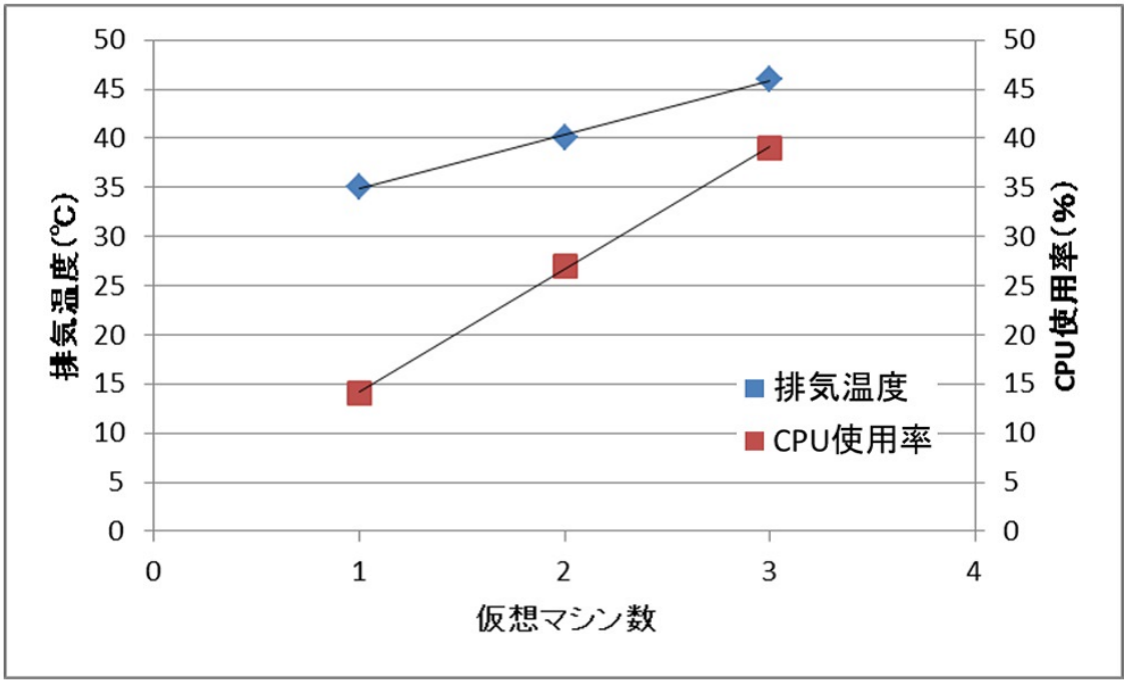

図 4.4: 仮想マシン数と使用率・排気温度の関係

4.4.2 仮想マシン CPU 数と使用率・排気温度

仮想マシン CPU 数を変動させたときの CPU 使用率と排気温度の変化を図 4.5 に示す．仮想マシン CPU 数を増加させると，サーバの使用率と排気温度は線形に上昇する．

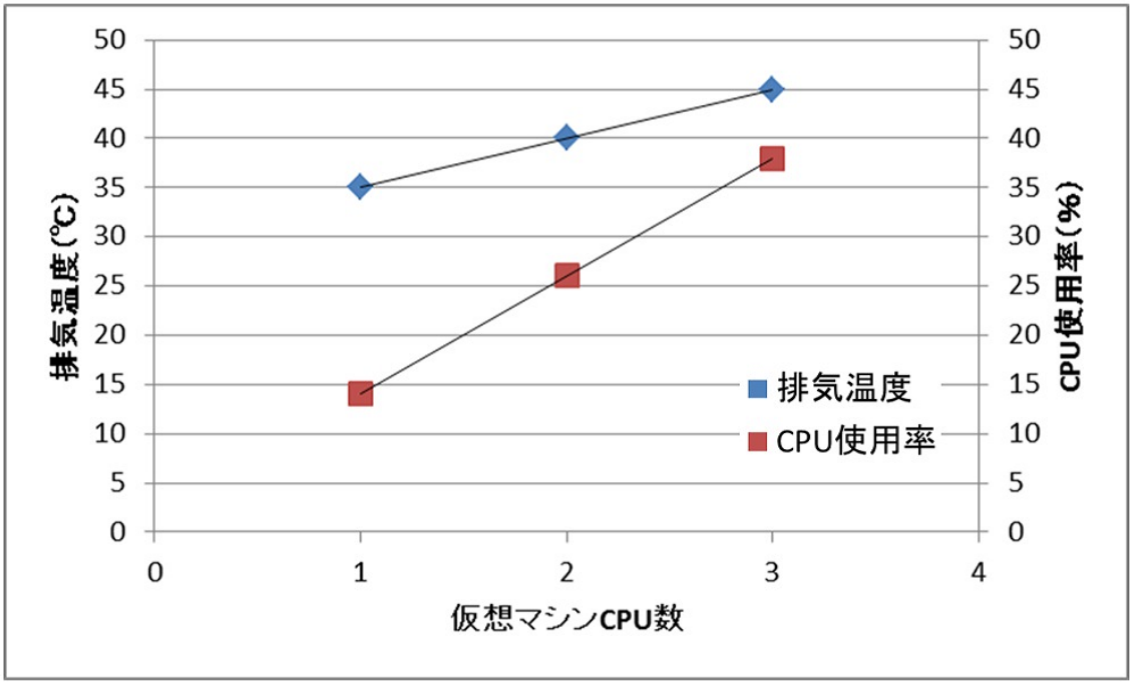

図 4.5: 仮想マシン CPU 数と使用率・排気温度の関係

4.4.3 評価

図 4.4 と図 4.5 を比較すると，変化の仕方・上昇量・最低値・最高値は，いずれもほぼ一致していることが分かる．これより，仮想化を多用することによって考慮すべき大きな負荷は発生しないと言える．

4.5　実測データよりモデル化

4.5.1　測定結果

CPU 使用率を増加させたときの動作温度と消費電力の変化のグラフを図 4.6 に示す．グラフを見ると，CPU 使用率とともに消費電力と温度が上がることが分かる．測定結果より T_{ex} を決定する．また，物理マシンがスタンバイモード時の消費電力は 3.30W である．これは，Intel Core i7 950 を対象とした実測データの一例である．

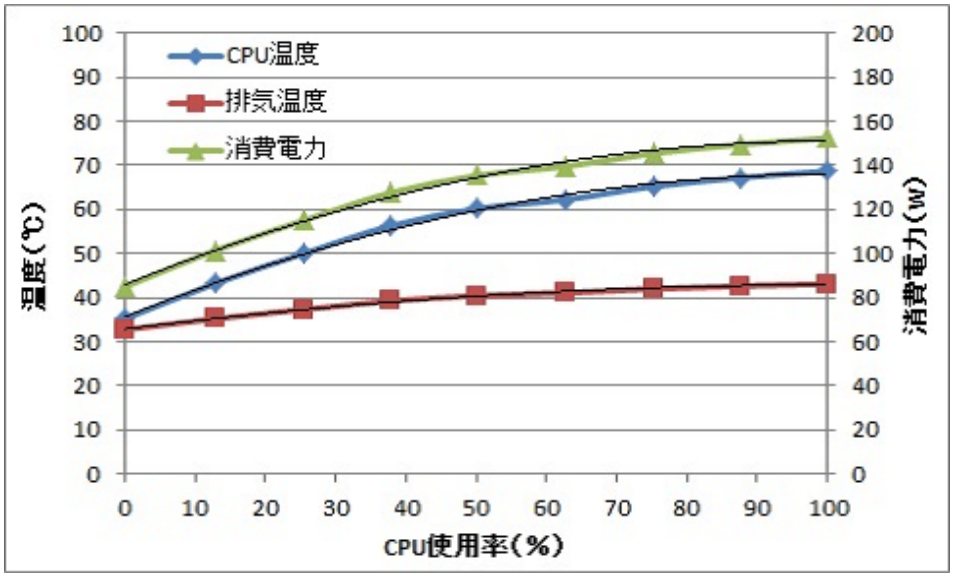

図 4.6: CPU 使用率と動作温度・消費電力の関係

4.5.2　動作温度モデル

排気温度 T_{ex}(℃) と CPU_HOST(%) の関係式は，

$$T_{ex} = 0.12 \times CPU\_HOST + 34.0 \tag{4.1}$$

室内温度が変わる場合には，関係式の作成に必要なパラメータを再度測定する必要がある．

4.6　推計による評価

推計プログラムを用いて，LOT 負荷分散法と簡略化矢嶋法でサーバを運用する際の信頼性を比較する．

4.6.1　推計プログラム概要

本研究では，推計をするにあたって推計プログラムを作成した．言語は Ruby で，2 次元配列をサーバに見立てて様々な大きさを持つ仮想マシンを配列に配置する．配置温度閾値なしとありの場合の実行結果を図 4.7 と 4.8 に示す．配置する仮想マシンのセットを図 4.9 に示す．

例えば，図 4.9 に示すような仮想マシンセットを配置しなければならないとする．排気温度閾値なしの場合，VM①(20%) と VM②(30%) は node00 に配置することができるが，VM③(60%) は node00 に配置できないため，サーバを 1 つシフトして node01 に配置される．VM④(50%) は node00 の空いている部分に配置される．VM⑤(50%) は node00 と node01 どちらの空いている部分にも配置することができないため，node02 に配置される．その他のサーバは，スタンバイ状態である．

同様に，図 4.9 に示すような仮想マシンセットを配置しなければならないとする．排気温度閾値ありの場合，VM①(20%) と VM②(30%) は node00 に配置することができるが，VM③(60%) は node00 に配置できないため，サーバを 1 つシフトして node01 に配置される．VM④(50%) は node00 の空いている部分に配置することができるが，配置すると排気温度閾値を超えてしまうため，サーバを 2 つシフトして node02 に配置される．VM⑤(50%) はと node01 の空いている部分には配置することができない．node00 と node02 の空いている部分に配置することができるが，配置すると排気温度閾値を超えてしまうため，サーバを 3 つシフトして node03 に配置される．その他のサーバは，スタンバイ状態である．

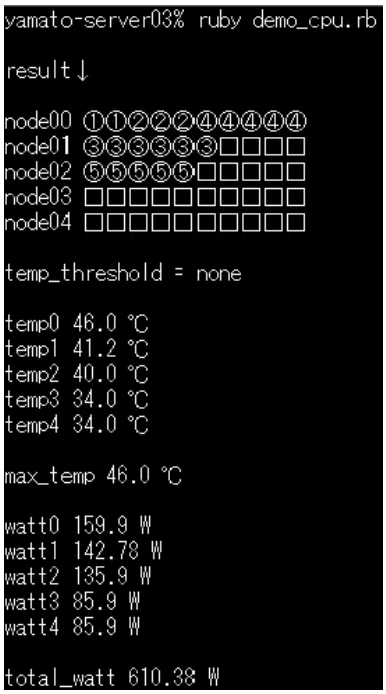

図 4.7: 実行結果 (排気温度閾値なし)

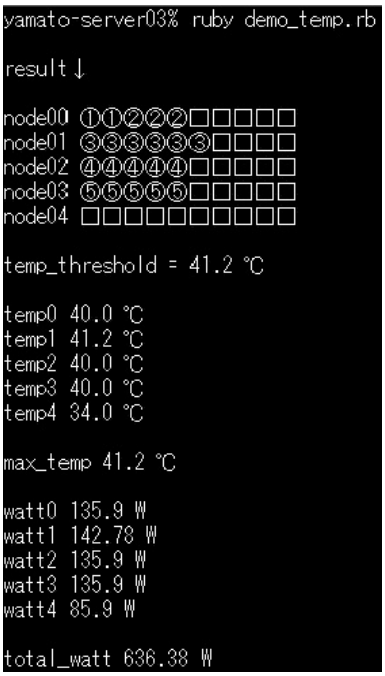

図 4.8: 実行結果 (排気温度閾値あり)

図 4.9: 仮想マシンセット

4.6.2　推計モデル

以下の条件で推計を行う．

- 1 架分のサーバラック (1U × 40) を想定，内最大で 30 台が稼動する．
- 仮想マシン要求は予め分かるとして選別する．
- 推計では，メモリ容量・ストレージ容量・バンド幅・仮想マシン CPU コア数とイメージサイズを考慮していない．
- 全てのタスクは均一と仮定する．
- 使用していない物理マシンはスタンバイモードにする．

4.6.3　信頼性に関する推計 (手順)

推計手順を図 4.10 に示す．

①目標とする 0.01 %故障時間を決定する．
②設定すべき T_{ex}_threshold を図 2.2 より計算する．
③稼動サーバ数と最高排気温度を推計プログラムで計算する．
④最高排気温度から正確な 0.01 %故障時間を計算する．

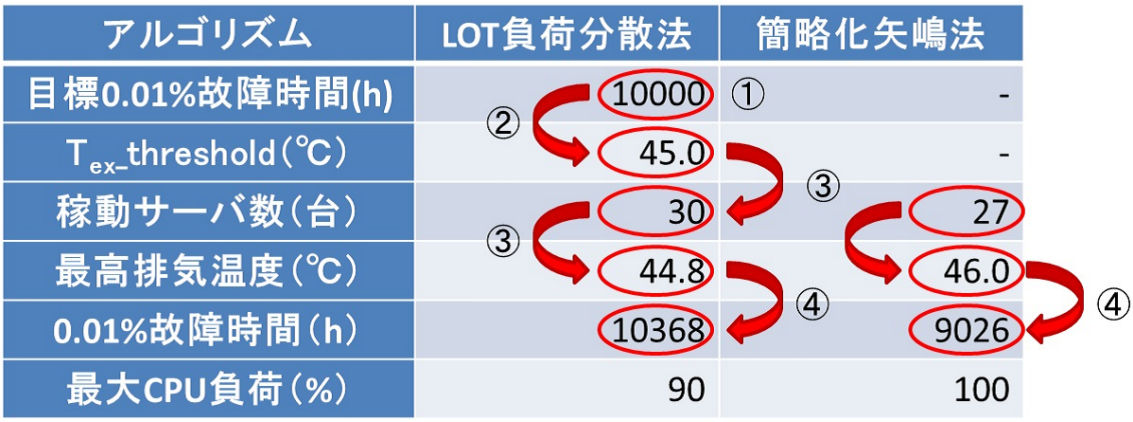

図 4.10: 推計手順

4.6.4 信頼性に関する推計 (結果)

信頼性に関する推計結果を図 4.3 に示す．LOT 負荷分散法の推計結果に注目すると，目標とする 0.01%故障時間を 10000h と決めたとき，T_{ex}_threshold は図 2.2 より 45.0 に設定すればよいと分かる．推計プログラムを実行すると，稼動サーバ数は 30 台，最高排気温度は 44.8 ℃であると分かる．この 44.8 ℃から CPU 温度を逆算して，[9] を使って正確な 0.01%故障時間を計算する．10368h となり，ユーザが求める信頼性の制約条件を満たしている．最大 CPU 負荷は 90%である．

表 4.3: 信頼性に関する推計結果

アルゴリズム	LOT 負荷分散法	簡略化矢嶋法
目標 0.01%故障時間 (h)	10000	-
T_{ex}_threshold	45.0	-
稼働サーバ数 (台)	30	27
最高排気温度 (℃)	44.8	46.0
0.01%故障時間 (h)	10368	9026
最大 CPU 負荷 (%)	90	100

4.6.5 CPU 使用率と最高排気温度

CPU 使用率と最高排気温度の関係を図 4.11 に示す．CPU 使用率の上昇とともに，最高排気温度が上昇する．

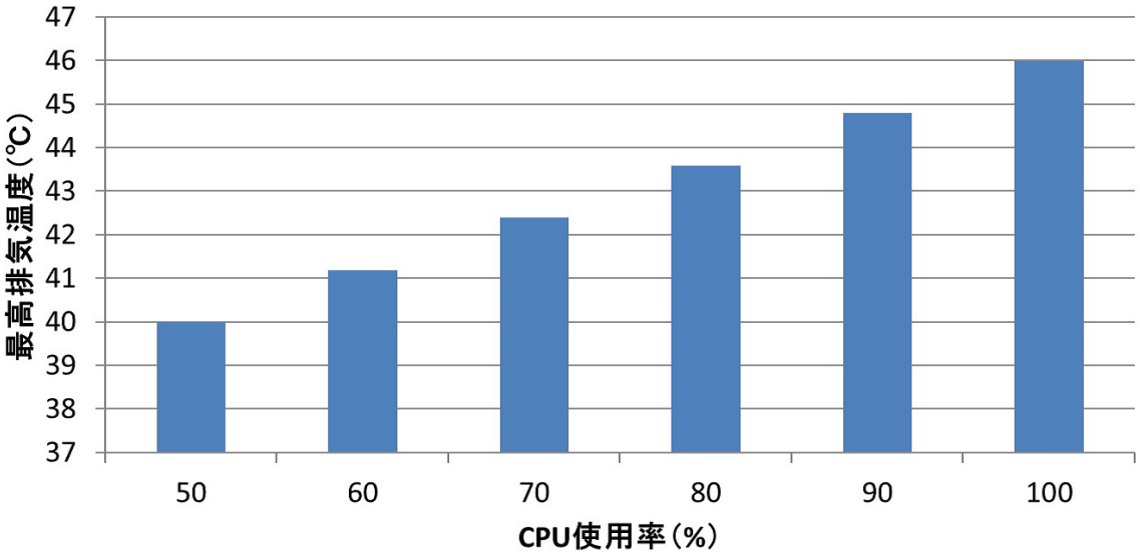

図 4.11: CPU 使用率と最高排気温度

4.6.6 T_{ex}_threshold と故障時間

T_{ex}_threshold と故障時間の関係を図 4.12 に示す．T_{ex}_threshold が大きくすると，0.01% 故障時間は短くなる．簡略化矢嶋法は，温度閾値なしの場合である．

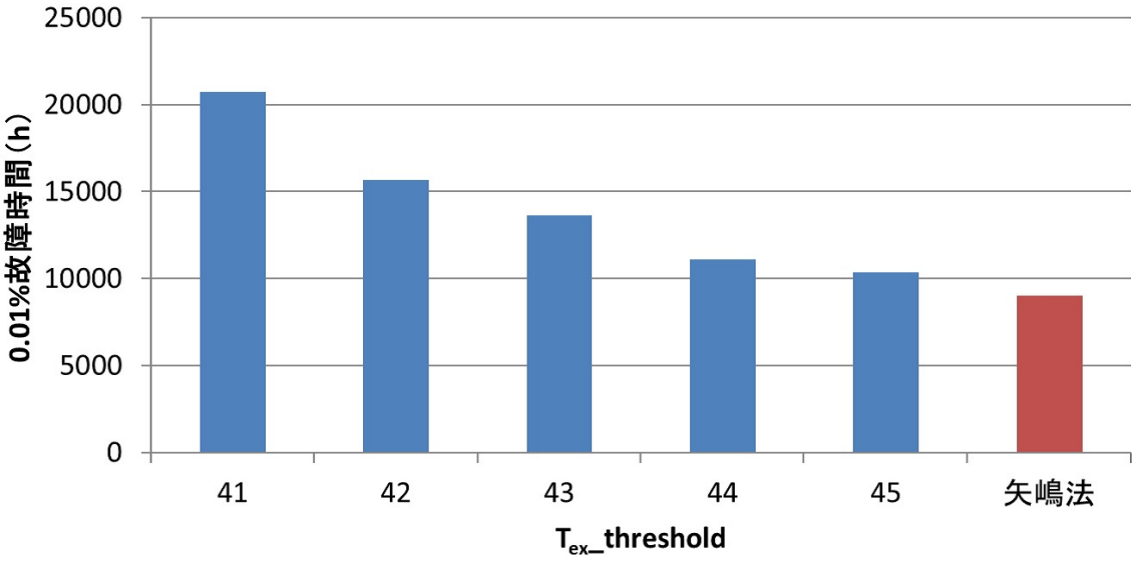

図 4.12: T_{ex}_threshold と故障時間

4.7 まとめ

本章では，温度と信頼性の評価を示した．まず 4.2 では，実験で使った実験サーバ・温度計の仕様と使い方を説明した．その後，それらを組み合わせた温度測定実験の環境構成を示して，データの取得方法を説明した．4.4 では，仮想マシン単体と複数で同じ量の負荷をサーバにかけたとき，CPU 使用率と動作温度を比較して評価する予備実験を行い，仮想化を多用することによって考慮すべき大きな負荷は発生しないと分かった．4.5 では，対象とするサーバを決めて実測データより動作温度モデルを作成した．CPU 使用率と動作温度の関係を数式化した．その後，そのモデルを用いて推計を行った．4.6 では，推計のために作成したプログラムの概要，想定する環境や制約条件を示した推計モデルを説明した．その後，信頼性に関する推計結果を示した．

第5章　温度と消費電力の評価

5.1　はじめに

本章では，温度と消費電力の評価をする．まず5.2では，実験で使った電力計の仕様と使い方を説明する．その後，それらを組み合わせた消費電力測定実験の環境構成を示して，データの取得方法を説明する．5.4では，対象とするサーバを決めて実測データより消費電力モデルを作成する．CPU使用率と消費電力の関係を数式化する．その後，そのモデルを用いて推計を行う．5.5では，消費電力に関する推計結果を示す．推計プログラムと推計モデルは，4.6と同様である．

5.2　実験機材

5.2.1　電力計

本研究では，電力計としてメガチップス社の電力測定システムを使用した．この製品には様々な測定機器が同梱されているが，その中で機器とコンセントの間に設置することで電力を測定することができるタップセンサを使用した．収集ゲートウェイは，消費電力量のデータを収集し，サーバにデータ転送するゲートウェイ機能を有した情報集中装置である．タップセンサは，測定結果を無線通信と電力線通信を用いて収集ゲートウェイへ送信する．収集ゲートウェイとタップセンサの写真を図 5.1 と図 5.2 に示す．

図 5.1: 収集ゲートウェイ

図 5.2: タップセンサ

5.3　実測実験の環境構成

5.3.1　消費電力測定実験の環境構成

消費電力モデルを作るための電力測定実験の環境構成を図 5.3 に示す．実験サーバ・タップセンサ・収集ゲートウェイ・モニタリング PC の４つで構成されている．タップセンサのデータを無線通信で収集ゲートウェイが収集して，モニタリング PC で取得する．

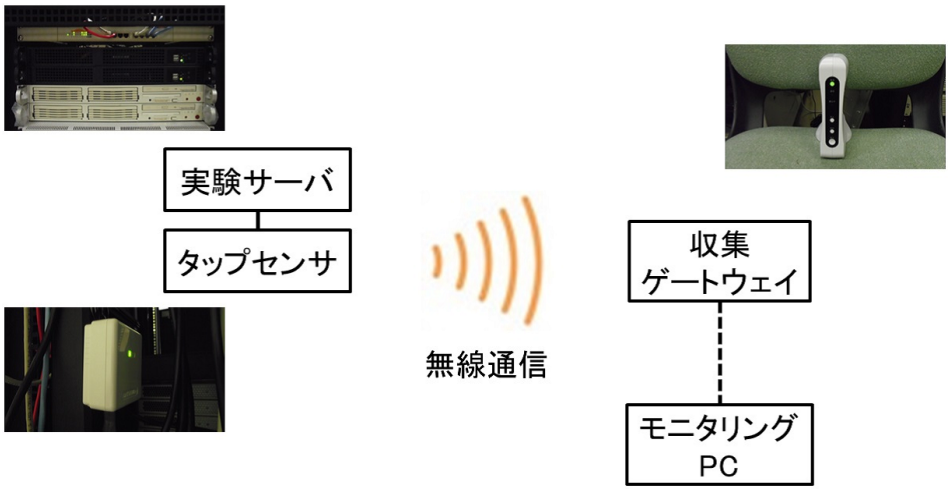

図 5.3: 消費電力測定実験の環境構成

5.4　実測データよりモデル化

5.4.1　消費電力モデル

消費電力 E(W) と CPU_HOST(%) の関係式は，

$$E = (-5.20 \times 10^{-3}) \times (CPU\_HOST)^2 + 1.26 \times (CPU\_HOST) + 85.9 \qquad (5.1)$$

5.5 推計による評価

本評価は，図 2.3 の考え方と計算方法を基に行っている．

5.5.1 消費電力に関する推計結果

排気温度閾値を 45.0 ℃ (CPU 使用率 90 %) に設定して，270 台の仮想マシンを配置した場合の消費電力に関する推計結果例を表 5.1 に示す．電力・温度の計算には，実測データから作成したモデルを使用している．結果を上から見ると，稼動サーバ数は提案手法では 30 台，既存手法では 27 台で 11%増加している．しかし，最高排気温度は提案手法では 44.8 ℃，既存手法では 46.0 ℃で 3%低下している．それによって，サーバ自体の消費電力合計は提案手法では 4287W，既存手法では 3420W で 25%増大している．空調消費電力の差を計算すると，既存手法よりも提案手法の方が 123W 増大している．侵入熱にかかる空調消費電力を-50.4W((P2-P1)/5) 削減できている．

表 5.1: 消費電力に関する推計結果

アルゴリズム	LOT 負荷分散法	簡略化矢嶋法	差
T_{ex}_threshold	45.0	-	-
稼働サーバ数 (台)	30	27	+11%
最高排気温度 (℃)	44.8	46.0	-3%
サーバ消費電力合計 (W)	4287	3420	+25%
空調消費電力 (W)	(4287+P_2)/5	(3420+P_1)/5	+123W

5.5.2 T_{ex}_threshold とサーバ消費電力合計

T_{ex}_threshold とサーバ消費電力合計の関係を図 5.4 に示す．T_{ex}_threshold を大きくすると，サーバ 1 台あたりに集約できる仮想マシン数が増えるため，稼動サーバ数が減る．それによってサーバ自体の消費電力合計は減少する．簡略化矢嶋法は，温度閾値なしの場合である．

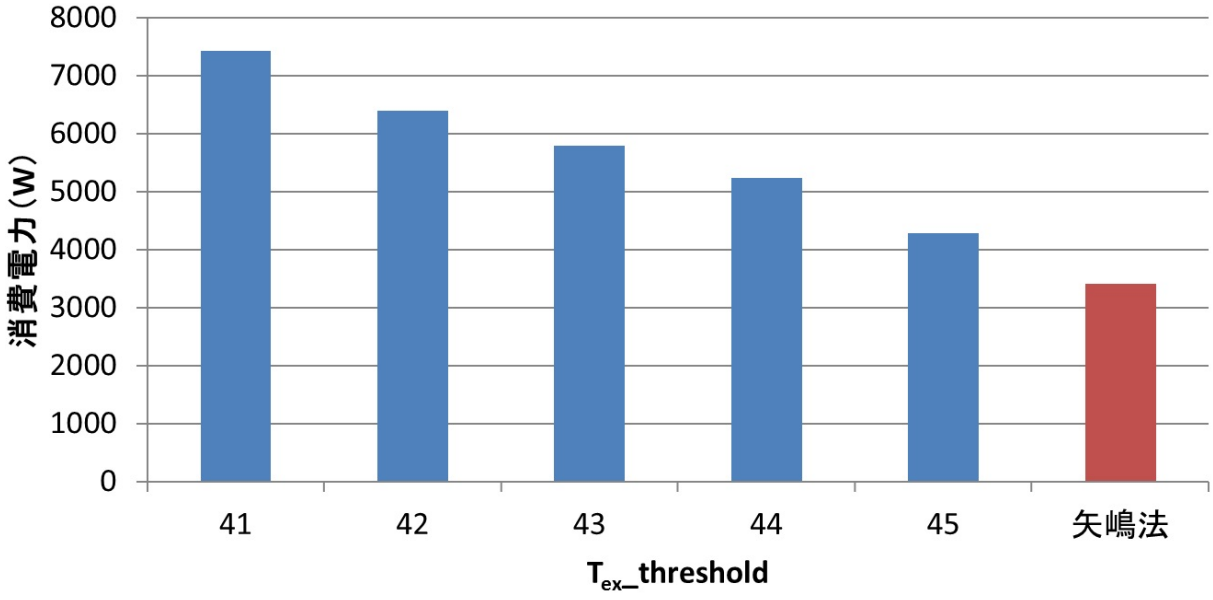

図 5.4: T_{ex}_threshold とサーバ消費電力合計

5.5.3 T_{ex}_threshold と侵入熱にかかる空調消費電力削減量

T_{ex}_threshold と侵入熱にかかる空調消費電力削減量の関係を図 5.5 に示す．2.3 で示したように，動作温度を低下させるほど侵入熱にかかる空調消費電力を削減することができる．

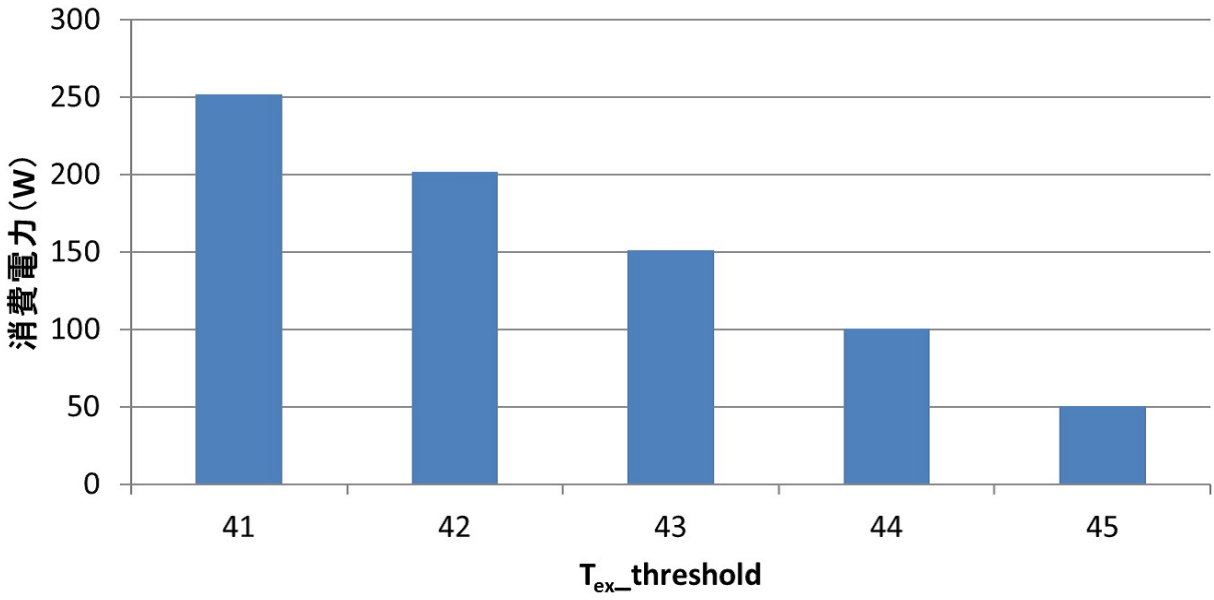

図 5.5: T_{ex}_threshold と侵入熱による空調消費電力削減量

5.6 まとめ

本章では，温度と消費電力の評価をした．まず 5.2 では，実験で使った電力計の仕様と使い方を説明した．その後，それらを組み合わせた消費電力測定実験の環境構成を示して，データの取得方法を説明した．5.4 では，対象とするサーバを決めて実測データより消費電力モデルを作成した．CPU 使用率と消費電力の関係を数式化した．その後，そのモデルを用いて推計を行った．5.5 では，消費電力に関する推計結果を示した．動作温度を低下させるほど侵入熱にかかる空調消費電力を削減できるが，稼動サーバ数が増加することで消費電力合計は大きく増大することが分かった．推計プログラムと推計モデルは，4.6 と同様である．

第6章　CloudSimによる評価

6.1　はじめに

LOT 負荷分散法は，仮想マシン単位で負荷分散をしているため，クラウドサーバへの適用が期待される．6.2 では，本研究におけるクラウドの考え方を述べる．6.3 では，本研究で使用したクラウド環境シミュレータ CloudSim に関して，概要・サンプルコードの実行手順・記述したコードの内容・動作制御の仕方・推計との比較を示す．6.4 では，メモリ・バンド幅・イメージサイズがそれぞれ異なる仮想マシンを LOT 負荷分散法と簡略化矢嶋法を用いてサーバに配置した場合のシミュレーションを行い，LOT 負荷分散法の性能評価とクラウドへの適用可能性を考える．

6.2 クラウドとは

クラウドは，米国 NIST(National Institute of Standards and Technology) が次のように定義している.「クラウドコンピューティングは，共用の構成可能なコンピューティングリソース (ネットワーク，サーバー，ストレージ，アプリケーション，サービス) の集積に，どこからでも，簡便に，必要に応じて，ネットワーク経由でアクセスすることを可能とするモデルであり，最小限の利用手続きまたはサービスプロバイダとのやりとりで速やかに割当てられ提供されるものである.」[8] このコンピューティングリソースを集積するアプローチにより，サーバの利用効率や電力効率を高めることができる．しかしながら，過剰な集積はサーバの信頼性を低下させ，物理サーバ毎の負荷に偏りがあると機械寿命にもばらつきが生じ，メンテナンス効率も低下すると予想される．

本研究においても，仮想化技術を用いて物理サーバに多くの仮想マシンを集約して管理する環境を想定している．6 章までにサーバの信頼性について述べてきたが，これはクラウドサーバにおいても重要である．一般的なクラウドでは，一部の物理サーバが故障し停止しても，処理能力は低下するがシステムは稼動し続け，定期的なメンテナンスで故障対応をしてリソースプールに戻すという管理をする．例えば，コンテナサーバなど小さなユニット毎にサーバの故障が発生するまでの時間を制御することが可能になれば，メンテナンス予定時期前に多くの物理サーバや部品が寿命となるように制御することで，故障対応を一度に行うことができる．結果として必要なメンテナンス回数が減り，管理者の負担を減らすことに繋がる．また，クラウド全体として高い処理能力を常に維持することができる．

6.3 CloudSim

CloudSim は，Java で作られたクラウド環境シミュレータであり，データセンタ，サーバ，VM等のクラウド構成要素は全てオブジェクトとして表現されている．Datacenter と DatacenterBroker というオブジェクトがあり，図 4.1 に示した手順でシミュレーションが進行する．

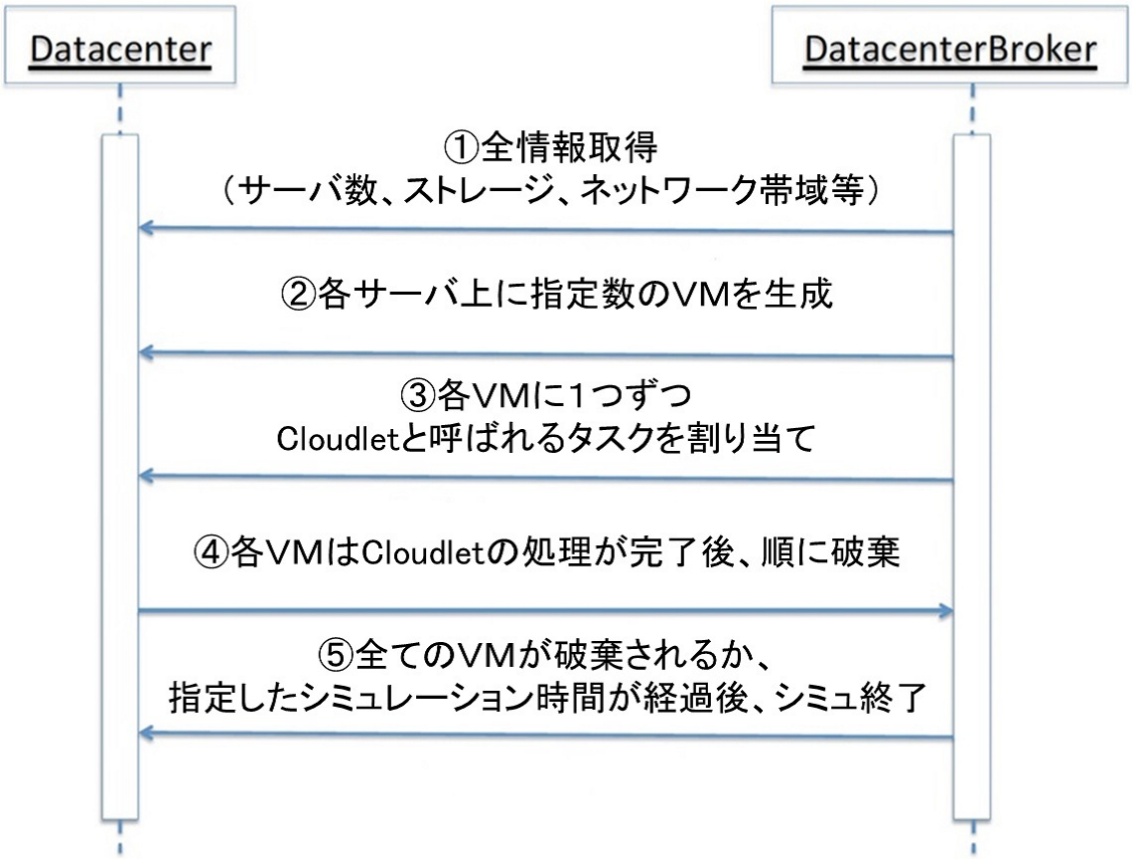

図 6.1: CloudSim フロー図

6.3.1 サンプルコードのコンパイルと実行手順

- Java を最新版にする
- CloudSim インストールと解凍

```
#yum install cloudsim
```

- パッケージのディレクトリに移動

```
#cd <PATH TO CLOUDSIM PACKAGE>
```

- コンパイル (CloudSimExample*.java)

```
#javac -classpath jars/cloudsim-<VERSION>.jar:examples examples/org/cloudbus/cloudsim/examples/CloudSimExample*.java
```

- 実行 (CloudSimExample*)

```
#java -classpath jars/cloudsim-<VERSION>.jar:examples org.cloudbus.cloudsim.examples.CloudSimExample*
```

6.3.2 サンプルコード実行例

CloudSimExample3 の実行結果を図 6.2 に示す．まず，シミュレーションを開始すると，スタートアップメッセージが流れる．次に，Broker が操作手順を報告する．最後に，シャットダウンメッセージが流れて，タスクである Cloudlet が割り当てられたデータセンタと仮想マシンが表示され，処理にかかる時間と開始時間と完了時間が表示される．

```
Starting CloudSimExample3...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: Trying to Create VM #1 in Datacenter_0
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: VM #1 has been created in Datacenter #2, Host #1
0.1: Broker: Sending cloudlet 0 to VM #0
0.1: Broker: Sending cloudlet 1 to VM #1
80.1: Broker: Cloudlet 1 received
160.1: Broker: Cloudlet 0 received
160.1: Broker: All Cloudlets executed. Finishing...
160.1: Broker: Destroying VM #0
160.1: Broker: Destroying VM #1
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    1        SUCCESS        2            1        80        0.1        80.1
    0        SUCCESS        2            0        160        0.1        160.1
CloudSimExample3 finished!
```

図 6.2: CloudSimExample3 実行結果

6.3.3 実行環境と各種パラメータ設定

実行環境と CloudSim・Java のバージョン，CloudSim 内変数の設定値を表 6.1 から表 6.5 に示す．表 6.3 の Cloudlet はタスクのことで，web のインタラクション量を示す．表 6.4 と表 6.5 に出てくる bw はバンド幅を示す.

表 6.1: CloudSim の実行環境

CPU	Intel Core i7
Memory	DDR3(8G)
OS	Ubuntu12.04(64bit)

表 6.2: CloudSim・Java のバージョン

CloudSim	cloudsim-3.0.3
Java	Java：jdk-1.7.0_40(Oracle)

表 6.3: Cloudlet の設定値

length	40000
file size	300MB
output size	300MB

表 6.4: サーバの仕様

mips	1000
host strage	1TB
host memory	16GB
bw	1Gbps

表 6.5: 仮想マシンの仕様

mips	200-800
image size	10GB
vm memory	1GB
number of cpus	1
bw	1Mbps

6.3.4 シミュレーションコード

CloudSim には，幾つか簡単なサンプルコードが入っていて，それらを組み合わせや変更を加えることで作成した．本コードは，その中でも記述を加えることでVM数とホスト数や各種性能を自由に変更することができる CloudSimExample3.java をベースに作成している．新たに追加した機能として，自動スケールアップ機能を実装した．これはホストマシンの性能不足により，指定したVM数が作成不可能だった場合，可能になるまでホスト数を増加させてシミュレーションをくり返す仕様である．その他，作成したVM数と使用したホスト数を表示させるなど，デフォルトにはない結果出力を可能にした．

6.3.5 閾値による動作制御

本研究のシミュレーションをするに当たって，ホストマシンに設定した温度閾値を超えないように仮想マシンを配置するシステムにしなければならない．これは言い換えると，CPU 使用率を制限可能なシステムにしなければならないということになる．しかしながらシステムの構成上，CPU 使用率の直接制限は困難であった．一方，仮想マシンやホストマシンのの MIPS 制限は容易である．そこで例えば，ホストマシンが 1000MIPS のとき CPU 使用率 100%であるとして，MIPS を 700MIPS に制限することは，CPU 使用率を 70%に制限していることと同じであると考えることとする．仮想マシンの CPU 使用率も MIPS を変更することで変動させる．

6.3.6　推計とシミュレーションの比較

同じ数の仮想マシンをサーバへ配置することを試みたとき，推計とシミュレーションで稼動サーバ数の結果が一致することを確認した．

6.4　シミュレーション結果

クラウド環境シミュレータ CloudSim を用いて，LOT 負荷分散法の性能を評価する．メモリ・バンド幅・仮想マシンのイメージサイズがそれぞれ異なる仮想マシンが複数ある場合，LOT 負荷分散法と簡略化矢嶋法を用いて仮想マシン配置を試みる．ユーザが求める信頼性の条件を各ノードが満たしているか確認する．

6.4.1　シミュレーションモデル

本研究では，以下の条件でシミュレーションを行う．

- 仮想マシン要求は予め分かるとして選別する
- 単純化の為，仮想マシン 1 台あたりの CPU 使用率は 10%に固定する
- 使用していない物理マシンはスタンバイモードにする
- 配置アルゴリズムは，デフォルトでリソースを最大限使う仕様になっているため，そちらを利用する

物理資源は，以下の仕様のサーバが 30 台あるとする．

- mips：1000
- メモリ 16GB
- ネットワークバンド幅：1Gbps
- イメージサイズ：10TB

要求資源を以下の 3 パターンでシミュレーションする．

- メモリ：4GB × 10，8GB × 10，12GB × 10
- ネットワークバンド幅：1Mbps × 10，10Mbps × 10，100Mbps × 10
- イメージサイズ：10GB × 10，100GB × 10，1TB × 10

6.4.2　メモリを変化させた場合

仮想マシンのメモリを変化させた場合の配置結果を図 6.3 に示す．4GB × 10，8GB × 10，12GB × 10，合計 30 台の仮想マシンを生成する．他のパラメータは，すべて同じとする．サーバのメモリは 16GB である．結果より，ユーザが求める信頼性の制約条件を満たしている．しかしながら，信頼性の制約条件よりもメモリの制約条件が強いため，LOT 負荷分散法は使われず，２つの手法で同じ配置結果となった．

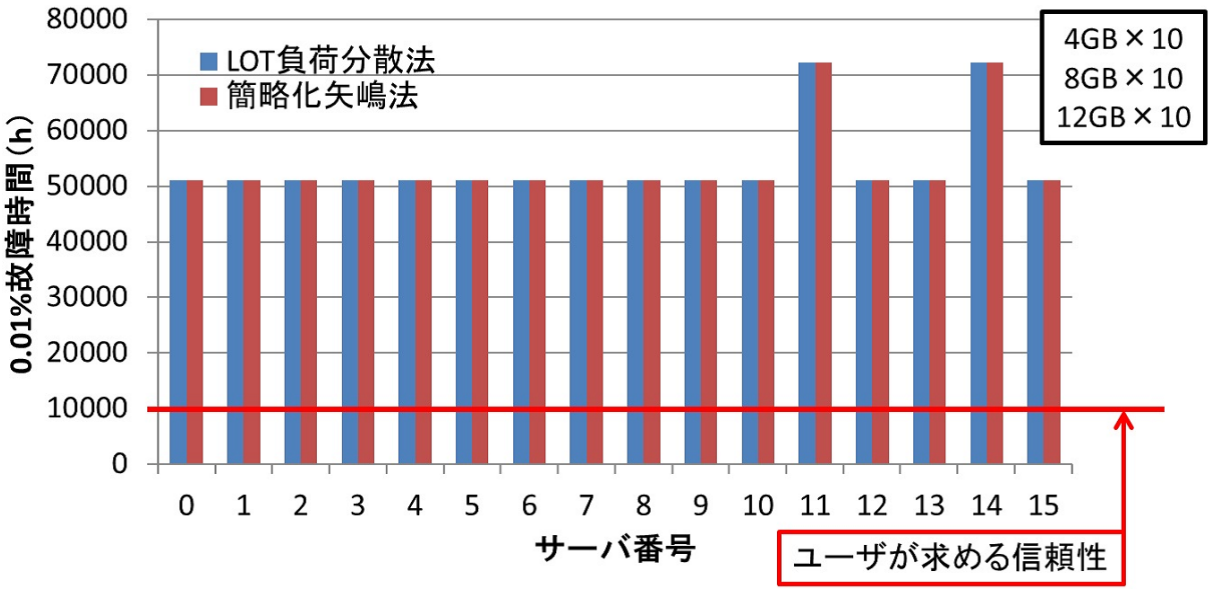

図 6.3: メモリを変化させた場合

6.4.3　バンド幅を変化させた場合

仮想マシンのバンド幅を変化させた場合の配置結果を図6.3に示す．1Mbps × 10，10Mbps × 10，100Mbps × 10，合計 30 台の仮想マシンを生成する．他のパラメータは，すべて同じとする．サーバのバンド幅は 1Gbps である．結果より，LOT 負荷分散法を用いた場合，ユーザが求める信頼性の制約条件を満たしている．稼動サーバ数は，簡略化矢嶋法よりも LOT 負荷分散法の方が 1 台多くなっている．

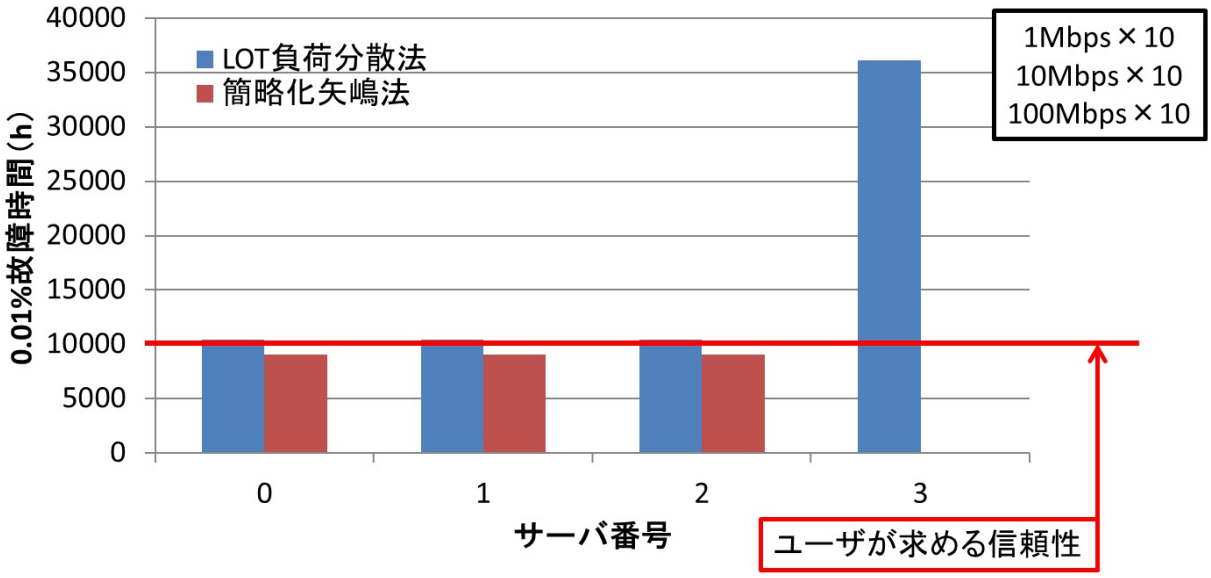

図 6.4: バンド幅を変化させた場合

6.4.4　イメージサイズを変化させた場合

仮想マシンのイメージサイズを変化させた場合の配置結果を図 6.3 に示す．10GB × 10，100GB × 10，1TB × 10，合計 30 台の仮想マシンを生成する．他のパラメータは，すべて同じとする．サーバのストレージ容量は 1TB である．結果より，LOT 負荷分散法を用いた場合，ユーザが求める信頼性の制約条件を満たしている．稼動サーバ数は，簡略化矢嶋法よりも LOT 負荷分散法の方が 1 台多くなっている．

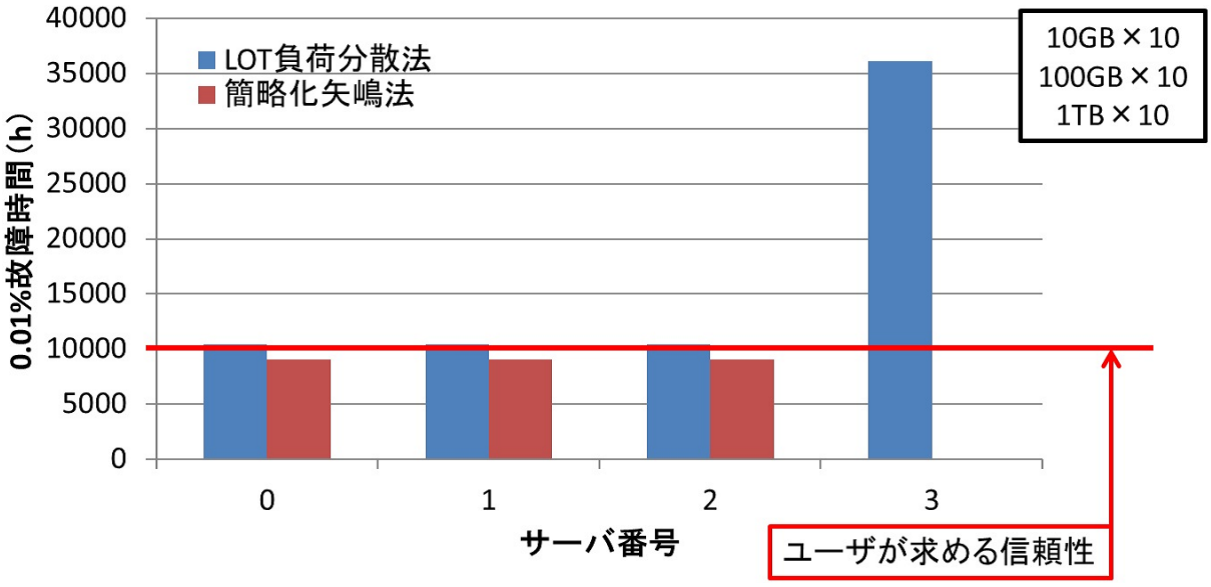

図 6.5: イメージサイズを変化させた場合

6.5　まとめ

6.2 では，本研究におけるクラウドの考え方を述べた．6.3 では，本研究で使用したクラウド環境シミュレータ CloudSim に関して，概要・サンプルコードの実行手順・記述したコードの内容・動作制御の仕方・推計との比較を示した．6.4 では，メモリ・バンド幅・イメージサイズがそれぞれ異なる仮想マシンを LOT 負荷分散法と簡略化矢嶋法を用いてサーバに配置した場合のシミュレーションを行い，LOT 負荷分散法が試験したすべての条件下でユーザが求める信頼性の制約条件を満たすことができると分かった．また，サーバのリソースを十分確保することが LOT 負荷分散法を効果的に使用する条件であると分かった．

第7章　まとめ

第 1 章では，本研究の背景と意義を述べた．第 2 章では，動作温度と電子部品の故障時間から信頼性を考え，サイト消費電力の計算方法を示した．その後，先行研究を紹介した．第 3 章では，提案手法のシステムモデルとアルゴリズムを説明した．第 4 章では，温度と信頼性について，実測データからモデルを作成して，信頼性に関する推計をすることで評価した．第 5 章では，温度と消費電力について，実測データからモデルを作成して，消費電力に関する推計をすることで評価した．第 6 章では，クラウド環境シミュレータ CloudSim を用いて，LOT 負荷分散法の性能評価をするとともに，クラウドへの適用可能性を検証した．

本研究では，ユーザが求める信頼性から決定した動作温度を超えない制約条件を満たしながら，低消費電力化のためできるだけ仮想マシンを集約配置して稼動サーバ数が最少になるような LOT 負荷分散法を提案した．LOT 負荷分散法を用いた推計結果より，稼動サーバ数は増えるが動作温度が低下することで，ユーザが求める信頼性の制約条件を満たすことが分かった．LOT 負荷分散法の性能評価とクラウドへの適用可能性を検証するため，クラウド環境シミュレータ CloudSim を用いて様々な制約条件下でシミュレーションを行った．結果より，LOT 負荷分散法を効果的に使うために最適な環境を明らかにした．また，これまで別々に考えられることが多い仮想マシンの配置アルゴリズムから機器の故障，空調消費電力の計算まで一貫した研究を行った．

7.1 今後の課題

- 対象とした電子部品がアルミ電解コンデンサだけであるため，他の部品についても考察する．
- 熱について複雑なモデルを作成していないため，流体モデルを作成するなど，より厳密なシミュレーションをする．
- 本研究では，仮想マシンの CPU 使用率は静的であると仮定しているため，動的にしてより現実に近いシミュレーションをする．
- アルゴリズムも未だ最適配置とは言い難いため，ナップザック問題のアルゴリズムなどを参考にして改良をする．
- 時間経過で仮想マシン配置を変更する場合に関して検討する．

参考文献

[1] Y.C.Lee, A.Y.Zomaya, “ Energy efficient utilization of resources in cloud computing systems ” The Journal of Supercomputing, Volume 60, Issue 2, pp. 268 - 280, May 2012

[2] Amit Nathani, Sanjay Chaudhary, Gaurav Somani, “ Policy based resource allocation in IaaS cloud ” Future Generation Computer Systems, Vol. 28, Issue 1, pp 94-103, January 2012

[3] A.Beloglazov, J.Abawajy, R.Buyya, “ Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing ” Future Generation Computer Systems, Volume 28, Issue 5, pp. 755 - 768, May 2012

[4] L.Wang, S.U.Khan, J.Dayal, “ Thermal aware workload placement with task-temperature profiles in a data center ” The Journal of Supercomputing, Vol. 61, pp. 780-803, June 2011

[5] 矢嶋秀敏, “ QoS を考慮した仮想マシンマイグレーションによる低消費電力クラウドに関する研究 ”, 北陸先端科学技術大学院大学修士論文, 2012 年 3 月

[6] X.Fan, W.Weber, L.A.Barroso, “ Power Provisioning for a Warehouse-sized Computer ” In Proceedings of the ACM International Symposium on Computer Architecture, San Diego, CA, June 2007

[7] 橋本 雅至, “ データセンタにおけるアプリケーションへの影響を考慮. した消費電力削減のためのサーバ統合手法の提案と評価 ”, 奈良先端科学技術大学院大学修士論文, NAIST-IS-MT1151082, 2013 年 2 月 7 日

[8] Peter Mell, Timothy Grance, “ NIST によるクラウドコンピューティングの定義 (Draft) 米国国立標準技術研究所による推奨 ”, 独立行政法人 情報処理推進機構, 2011 年 1 月

[9] Panasonic アルミ電解コンデンサの寿命予測計算 (http://www2.panasonic.co.jp/aec/reliability/aluminum.html)

謝辞

本研究を行うにあたり，ご指導を受け賜りました北陸先端科学技術大学院大学の情報社会基盤センター井口 寧教授に深く感謝致します.

貴重な御助言，御意見を頂いた金子 峰雄教授，田中 清史准教授に深く感謝致します. 副テーマでお世話になりました赤木 正人教授に深く感謝致します. 井口研究室でのゼミで貴重な御助言，御意見を頂いた佐藤 幸紀助教と井口研究室メンバーに深く感謝致します.

ハードウェア合同ゼミでの貴重な御助言，御意見を頂いた田中研究室の請園 智玲助教に深く感謝致します. 研究を進める上で，様々なアドバイスをして頂き，また相談に乗って下さった荒木 光一氏に深く感謝致します.

研究の議論やアドバイスをしてくれ，研究を共に頑張った佐々木 泰氏，中吉 達二氏に深く感謝致します. 仮配属から共に頑張った長谷川研究室の吉良 元氏，敷田研究室の川井 俊輝氏，西野 博之氏に深く感謝致します.

最後に，ここまでの学生生活を暖かく見守って下さった両親と妹に深く感謝致します.