Zipf 分布型の処理要求に適したスケールアウト手法における記憶域近似的最小化の拡張

山下高生[†1] 栗田弘之[†1] 高田直樹[†1] 南拓也[†1] 太田賢治[†1]

我々は，これまで，WWW(World Wide Web)やネットワーク装置の制御に用いられるデータ処理において，少量のデータに大半の処理要求が集中する一方で，処理要求頻度が低いデータが大量に存在するような Zipf 分布型の特性を持つ処理要求に対し，サーバ負荷の偏りを一定以下に保ちながら，サーバ全体で必要な記憶域の近似的最小化を実現する方法を提案してきた．これまでの提案方法は，要求頻度の違いに応じて，ラウンドロビン，ラウンドロビンとコンシステントハッシングのハイブリッド型処理，コンシステントハッシングの三通りの処理方法を使い分けるスケールアウト可能な負荷分散方法である．本論文では，最初に，これまで提案してきた方法が，処理要求に応答するために必要なデータのサイズが平均的に同一であるという既提案の条件から，処理要求の頻度に対して任意の変化をする条件に拡張可能であることを示す．次に，処理要求の頻度に比例したデータを管理する必要がある条件下において．シミュレーションによる評価を行い，負荷分散を実現しながら，記憶域近似的最小化により既存技術と比較して大幅に記憶域を削減可能であること，および，サーバ間の記憶域サイズの偏りについても実用上十分な範囲を実現できることを示す．

A Generalized Method of Load Balancing and Approximate Memory Minimization in Scale-Out System Architecture for Processing Requests That Follow Zipfian Distribution

TAKAO YAMASHITA[†1] HIROYUKI KURITA[†1] NAOKI TAKADA[†1]
TAKUYA MINAMI[†1] KENJI OTA[†1]

1. はじめに

近年のネットワーク技術，および，端末技術の発展や広帯域通信の普及にともない，WWW(World Wide Web)，電子商取引などのアプリケーションや，認証，名前解決などの通信の制御として，データ中心処理の応用範囲は，大幅に広がってきている．

このようなデータ中心処理は，銀行の OLTP(On-Line Transaction Processing)などの厳密な一貫性制御[1)2)]が必要な複雑な処理での利用に加え，WWW や名前解決など一貫性制約は弱いが，非常に多くの要求を処理する必要のある応用範囲での利用も急速に広がってきている．

これらの応用分野に対し，スケーラビリティを高める取り組みに加え[3)4)5)6)7)8)9)10)11)12)]，更に，コンシステントハッシング等を利用したスケールアウト技術が注目されている[13)14)]．スケールアウト技術は，ユーザ数やユーザからのサービス利用頻度などのスケールに応じたサービスの提供に柔軟に対応できることから，新しいサービスを迅速に提供し，その成長に柔軟に対応可能とする技術である．

一方で，データ中心処理の中には，少量のデータに大半の処理要求が集中しながら，処理要求頻度が低いデータが大量に存在するような処理要求特性をもった処理がある．このような処理要求特性の中で，WEB プロキシーへのアクセス頻度など， Zipf 分布[15)]に従うアプリケーション特性が報告されている[16)]．

コンシステントハッシングは，システムが提供するサービスを特定するためのキーを，複数のサーバが，分割して受け持つことで，サーバ間の処理負荷が平均化される．しかし，Zipf 分布のように処理要求頻度の偏りが大きい場合，サービスを提供するための処理負荷についてサーバ間で平均化しにくく，処理負荷に大きい偏りが発生し，結果としてサービスを提供するためのサーバ数が増加してしまう．

我々は，このような要求頻度の大きな偏りをもつサービス提供に利用可能なスケールアウト技術をこれまで提案してきた[17)18)19)20)]．本既存提案方法は，サービスを特定するキーを要求頻度の高いものから低いものの順序に並べ，その順序をランクとし，要求頻度の高いランク範囲，中程度のランク範囲，低いランク範囲に対して，それぞれ，異なる負荷分散手法を用いる方法である．すなわち，要求頻度の高いホットゾーンと呼ぶランク範囲に対しては，全サーバによるラウンドロビン，要求頻度の低いコールドゾーンと呼ぶランク範囲に対しては，コンシステントハッシング，要求頻度が中程度であるノーマルゾーンと呼ぶ範囲に対しては，複数のサーバによるラウンドロビン法を用いるものである．この方法は，ホットゾーン，および，ノーマルゾーンの範囲を広げると負荷分散としては有利であるが，一方で，同一のキーが，多くのサーバに複製されることで各

†1 日本電信電話株式会社 NTT ネットワークサービスシステム研究所
IPSJ
DICOMO2013

サーバに必要な記憶域のサイズが増加し，その結果，サーバ毎のメモリやSSD(Solid State Drive)等の搭載量の制約から,サーバ数が増加することで,コストが増大してしまう．また，運用者による各ゾーンの設定は，運用者のスキルやオペレーションコストを増大させる．このため，サーバを効率的に利用し,運用コストを削減することを目的として，各キーを処理するために必要なデータのサイズが平均的に同程度である条件下で，記憶域を近似的に最小化するゾーンの分割方法を提案してきた[20]．

これに対して，要求頻度の大きな偏りをもつサービス提供が必要な応用範囲として,ネットワーク制御を考えた場合，利用頻度の高いキーほど，大きなデータもつ場合がある．これは，キーによって識別されるサービスを利用するユーザが増加するにつれて，サービスを提供するためのネットワーク装置が増加し，結果として，制御される対象となるネットワーク装置の数に比例して，キーを処理するためのデータのサイズが増加することによって発生する．

本論文では，最初に，これまで提案してきた記憶域を近似的に最小化するゾーンの分割方法について，各キーを処理するためのデータのサイズが，キーの要求頻度の変化に対して，任意の変化をする場合においても，同じ分割方法が利用可能であることを示す．そのことにより，サーバの負荷分散を実現し，且つ，必要な記憶域のサイズを近似的に最小化できるホットゾーン，ノーマルゾーン，および，コールドゾーンの自動分割を可能とする.次に,提案方法を，キーの要求頻度の高さに比例して，データのサイズが大きくなる条件に適用した場合に，サーバ負荷の偏りを抑え，十分な負荷分散を実現しながら，既存の方法と比較して処理に必要な記憶域のサイズが削減できていることについて評価した結果を議論する．更に，サーバ間の記憶域サイズの偏りについても実用上，十分な範囲を実現できることを示す.

本論文の構成について説明する．章2で，これまで提案してきた要求頻度の大きな偏りをもつサービス提供に利用可能なスケールアウト技術[17)18)19)20)]の概要を説明する．章3では，各キーを処理するために必要なデータのサイズが平均的に同程度である条件下での記憶域近似的最小化[20)]として提案したゾーン分割方法における，負荷分散を実現する条件を説明する．本条件は，各キーを処理するために必要なデータのサイズには無関係の条件であり，本論文で対象としている,各キーを処理するためのデータのサイズが,キーの要求頻度の変化に対して任意の変化をする場合においても同じ条件となる．章4では，各キーを処理するためのデータのサイズが，キーの要求頻度の変化に対して任意の変化をする場合について，記憶域近似的最小化を実現する方法について述べる．章5では，提案方法を用いることにより，十分な負荷分散を実現しながら，既存方法と比較して，記憶域の大幅な削減が実現できることについて説明する．更に，サーバ間の記憶域サイズの偏りについても実用上，十分な範囲を実現できることを示す．最後に，章6で，本論文のまとめを述べる．

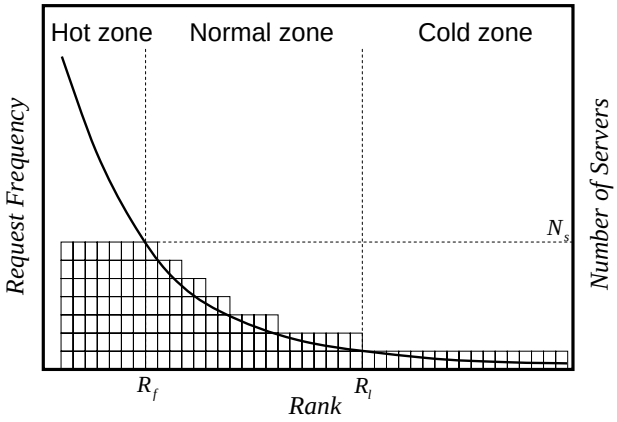

図 1　Zipf 分布と既定案方法における負荷分散用のサーバ台数の関係

2. Zipf 分布型処理要求のための負荷分散方法の概要

処理を特定するためのキーについて，あるキーの処理要求に含まれる頻度(以降，キーの要求頻度と呼ぶ)が，Zipf分布に近い分布である場合，**図 1** に示す分布となる．本図は，横軸が，キーの要求頻度を，頻度の高いものから，低いものの順序に並べたときの順位(以降,キーのランクと呼ぶ)であり,縦軸が,キーの要求頻度である.本図のように，ランクの小さいキーの要求頻度が非常に高く，ランクが大きくなるにつれて劇的に要求頻度が低くなる特徴を持つ．本章では，まず，これまで提案してきた，上記のような特徴を持つ要求を処理するシステムにおける負荷分散を実現するための既存提案方法を説明する．既存提案方法では，ランクの範囲を,三つの範囲に分割する.これらの範囲は，要求頻度の高い領域，中間程度の領域，低い領域の三つの領域であり，それぞれ，**図 1** に示すように，ホットゾーン(Hot zone)，ノーマルゾーン(Normal zone)，および，コールドゾーン(Cold zone)と呼ぶ．ここで，ノーマルゾーン，および，コールドゾーンの最も要求頻度の高いランクを，それぞれ，R_f(≧1)，および，R_l (≧1)と表す．ホットゾーンに属するランクのキーに対する処理要求は，全サーバによってラウンドロビン処理することで，負荷分散を実現する．コールドゾーンに属するランクのキーに対する処理要求は，全サーバを用いてコンシステントハッシングによる処理を行うことで，負荷分散を実現する．ノーマルゾーンに属するランクのキーに対する処理要求は，全サーバ台数未満且つ2台以上のサーバによってラウンドロビン処理することで，負荷分散を実現する．どのサーバを使用するかについては，コンシステントハッシングと同様に，キーのハッシュ値を用いて決定する．ここで，各キーを処理するために

用いるサーバ台数を，分散度と呼ぶ．ランク r の要求頻度，および，分散度を，それぞれ，$f(r)$，および，$d(r)$で表すと，分散度は，N_s，および，N_r を，それぞれ，全サーバの台数，および，全キー数として，下記のように定義される．

$$d(r)=\begin{cases} N_s & (1\le r\le R_f-1) \\ g(r)=\lceil f(r)/f(R_l)\rceil & (R_f\le r\le R_l-1) \\ 1 & (R_l\le r\le N_r) \end{cases} \quad (1)$$

ここで，ランク R_l における要求頻度を，基本要求頻度と呼び，ノーマルゾーンに含まれるランクのキーの分散度は，この基本要求頻度をもとに決定する．また，ノーマルゾーンの中のホットゾーンとの境界である R_f も，基本要求頻度によって決まり，R_f は，$g(R_f\text{-}1)\geqq N_s$，且つ，$g(R_f)<N_s$ となるように定める．

上式において，要求頻度を一般的に $f(r)$として表したが，$f(r)$ が完全に Zipf 分布に一致する場合，以下の式で表される．ここで，A と p は定数である．

$$f(r)=A/r^p \quad (2)$$

図 1 の中に描いた四角形の一つ一つは，それぞれが，各ランクのキーを処理するサーバが一台存在することを意味している．本図において，ホットゾーンとコールドゾーンではサーバ台数が一定であり，ノーマルゾーンでは，ホットゾーンとの境界からコールドゾーンとの境界に向けて階段状に減少する様子を表している．

3. 負荷分散条件

3.1 概要

前章で説明した既存提案方法において，Zipf 分布である $f(r)$が式(2)で与えられたとき，ノーマルゾーン，および，コールドゾーンの最も要求頻度の高いランクである，R_f，および，R_l に応じて，各ランクを処理するためのサーバ台数である分散度が，式(1)のように決定される．このことにより，サーバ間の負荷分散の度合い，および，各サーバに必要な記憶域のサイズが変化する．

各ゾーンにおける負荷分散の実現性について考えると，ホットゾーンに含まれるキーに対する処理要求は，全サーバを用いたラウンドロビン処理により，全てのサーバの間で負荷分散が実現される．コールドゾーンについては，各キーの要求頻度が，サーバにより処理可能な要求頻度に比べて十分小さく，比較的，偏りが少なく，且つ，サーバ数に対して十分大きな数のキーが，コールドゾーンに含まれることで，コンシステントハッシングによりサーバ全体で負荷分散が実現される．一方で，ノーマルゾーンについては，比較的要求頻度が高いため，サーバ全体にどのように負荷を与えるかは，ノーマルゾーンの領域をどのように決めるかによって，すなわち，R_f と R_l をどのように決めるかによって変化することになる．本章では，章 2 で説明した既存提案方法のうち，章 2 の最後から 3 番目のパラグラフで説明した R_f の決定方法のみについては用いず，R_l と R_f を独立に考え，ノーマルゾーンが十分な負荷分散を実現できるランクの領域となる条件について説明する．本条件は，その条件下で，R_l と R_f を変化させて記憶域の最小化を行うため，ノーマルゾーンの位置に依存しない十分な条件として定めたものである．本章で説明する条件を用いて記憶域を近似的に最小化する方法を，次章で説明する．次章で説明する記憶域の近似的最小化では，各キーを処理するためのデータサイズが，キーの要求頻度の変化に対して任意の変化をする場合においても，既存提案 [20]のゾーン分割方法が利用可能であることを示す．

3.2 ノーマルゾーンの負荷分散の条件

章 3.1 で説明したように，ノーマルゾーンの領域の決め方により，ノーマルゾーンによる負荷分散への影響が決まる．ノーマルゾーンに属するキーの処理の負荷分散は，**図 1** に描かれている，章 2 で説明した要求処理のサーバ割り当てを表す四角形が，全サーバにどのように分散されるかによって決まる．負荷分散を実現するためには，この四角形が全サーバに均等に割り当てられることが必要となり，その為には，全サーバ台数に対して十分に大きな数のサーバ割り当てを表す四角形が存在する必要がある．言い換えると，ノーマルゾーンに属する各キーを処理するためのサーバ台数である分散度の総和を，全サーバ台数と比較して十分に大きくする必要がある．

本条件を求めるために，まず，ノーマルゾーンの分散度について，その総和の近似的な下限を以下に求める．

$$\begin{aligned} \sum_{r=R_f}^{R_l-1}\lceil g(r)\rceil &= \sum_{r=R_f}^{R_l-1}\left\lceil \frac{f(r)}{f(R_l)}\right\rceil = \sum_{r=R_f}^{R_l-1}\left\lceil \frac{R_l^p}{r^p}\right\rceil \ge \int_{r=R_f}^{R_l}\frac{R_l^p}{r^p}dr \\ &= \begin{cases} \dfrac{R_l^p}{1-p}\left(R_l^{1-p}-R_f^{1-p}\right) & (p\ne 1) \\ R_l^p\left(\log R_l-\log R_f\right) & (p=1) \end{cases} \end{aligned} \quad (3)$$

式(3)の不等式の右辺が，サーバ数 N_s の K 倍となる条件は下記となる．

$$\begin{cases} \dfrac{R_l^p}{1-p}\left(R_l^{1-p}-R_f^{1-p}\right)\ge KN_s & (p\ne 1) \\ R_l^p\left(\log R_l-\log R_f\right)\ge KN_s & (p=1) \end{cases} \quad (4)$$

ここで，K が 1 と比較して十分に大きな値となることで，ノーマルゾーンの分散度の総和が，全サーバ数と比較して十分に大きな値となる．上記の条件を，R_f を R_l で表すようにまとめると次のようになる．

$$R_f\le\begin{cases} \left[R_l^{1-p}-(1-p)KN_sR_l^{-p}\right]^{\frac{1}{1-p}} & (p\ne 1) \\ R_l\exp\left(-\dfrac{KN_s}{R_l^p}\right) & (p=1) \end{cases} \quad (5)$$

式(5)の条件に加え，R_f と R_l には，ノーマルゾーンにおける

分散度が，N_sの値を超えないようにしなければならないことから，次の条件を満たさなければならない．

$$N_s \geq \left\lceil \frac{R_l^p}{R_f^p} \right\rceil \geq \frac{R_l^p}{R_f^p} \tag{6}$$

式(6)について，R_fをR_lで表すようにまとめると次のようになる．

$$R_f \geq N_s^{-\frac{1}{p}} R_l \tag{7}$$

以上の議論から，ノーマルゾーンに属するキーの処理の負荷分散を実現するためには，式(5)と(7)の条件を，R_f≧1，および，R_l≧1 と同時に満足する必要がある．

式(5)と(7)の条件について，R_f，および，R_l を，それぞれ，縦軸，および，横軸とするグラフにプロットするために，式(5)と(7)の等号が成立する条件同士の交点となる R_l を求めると，以下の式で表される(付録参照)．

$$R_l = \begin{cases} \dfrac{(1-p)KN_s}{1-N_s^{\frac{p-1}{p}}} & (p \neq 1) \\ \dfrac{pKN_s}{\log N_s} & (p = 1) \end{cases} \tag{8}$$

このときの R_fは，上式の R_lを，式(7)の等号が成立する条件に代入することで求めることができ，下記となる．

$$R_f = \begin{cases} \dfrac{(1-p)KN_s}{1-N_s^{\frac{p-1}{p}}} N_s^{-\frac{1}{p}} = \dfrac{(1-p)K}{N_s^{\frac{1-p}{p}} - 1} & (p \neq 1) \\ \dfrac{pKN_s}{\log N_s} N_s^{-\frac{1}{p}} = \dfrac{pKN_s^{\frac{p-1}{p}}}{\log N_s} & (p = 1) \end{cases} \tag{9}$$

4. 任意のデータサイズ変化における記憶域近似的最小化

章 3.2 の議論をもとに，サーバ間の負荷分散を実現するための条件を，グラフにプロットすると，**図 2**，および，**図 3** となる．**図 2**，および，**図 3** の縦軸，および，横軸は，それぞれ，R_f，および，R_lである．

まず，**図 2** は，式(9)の R_fの値が 1 以上の場合であり，同図の(a)，および，(b)は，それぞれ，p≧1，および，0<p<1 の場合である．同図の斜線の領域は，式(5)と(7)の条件，R_f≧1，および，R_l≧1 の条件を満足する領域である．

上記の領域の中で，記憶域の最小化を行うために R_f，および，R_lを，どのように決めるかを説明する．まず，各ランク r のキーを処理するために必要なデータのサイズを $e(r)$(> 0)と表す．

今，ノーマルゾーン，および，コールドゾーンの最も要求頻度の高いランクを，それぞれ，R_f，および，R_l としたときの記憶域のサイズを，$m(R_f, R_l)$と表すこととする．この値と，R_fを R_f+1 に変化させたときの記憶域のサイズ $m(R_f+1, R_l)$，および，R_lを R_l +1 に変化させたときの記憶域のサイズ $m(R_f, R_l+1)$と比較して，R_fと R_lの変化による記憶域のサイズの変化を，以下に導く．

まず，ノーマルゾーン，および，コールドゾーンの最も要求頻度の高いランクを，それぞれ，R_f，および，R_l としたときの記憶域のサイズ $m(R_f, R_l)$は，次式で表される．

$$m(R_f, R_l) = N_s \sum_{r=1}^{R_f-1} e(r) + \sum_{r=R_f}^{R_l-1} e(r) \left\lceil \frac{R_l^p}{r^p} \right\rceil + \sum_{r=R_l}^{N_r} e(r) \tag{10}$$

今，R_fを R_f+1 に変化させたときの記憶域のサイズ $m(R_f+1, R_l)$は，次式となる．

$$m(R_f+1, R_l) = N_s \sum_{r=1}^{R_f} e(r) + \sum_{r=R_f+1}^{R_l-1} e(r) \left\lceil \frac{R_l^p}{r^p} \right\rceil + \sum_{r=R_l}^{N_r} e(r) \tag{11}$$

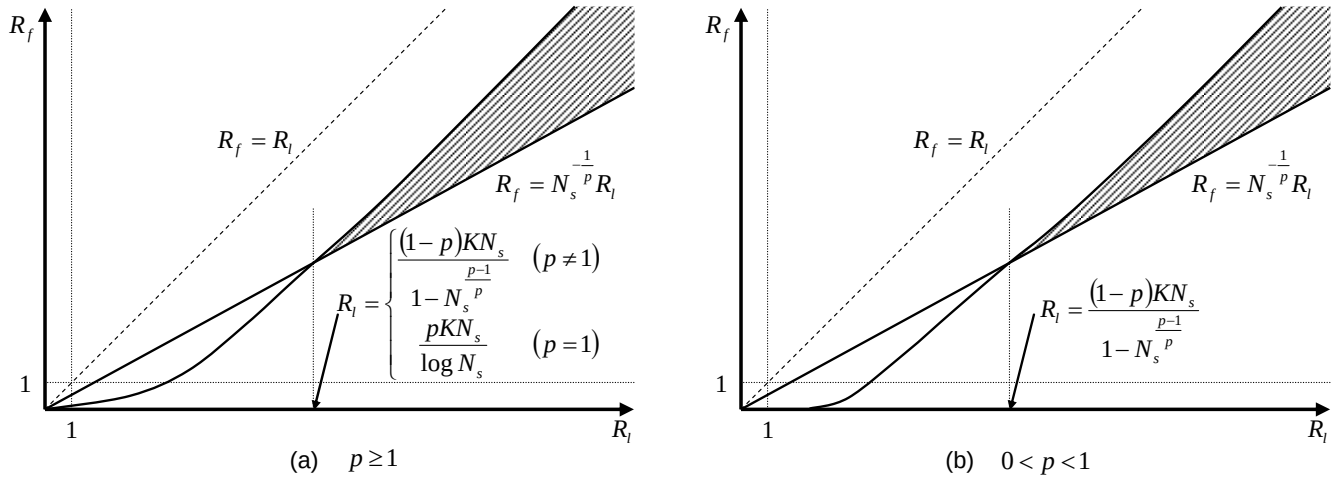

図 2　負荷分散条件とサーバ台数条件の交点となる R_f が 1 以上における二条件の関係

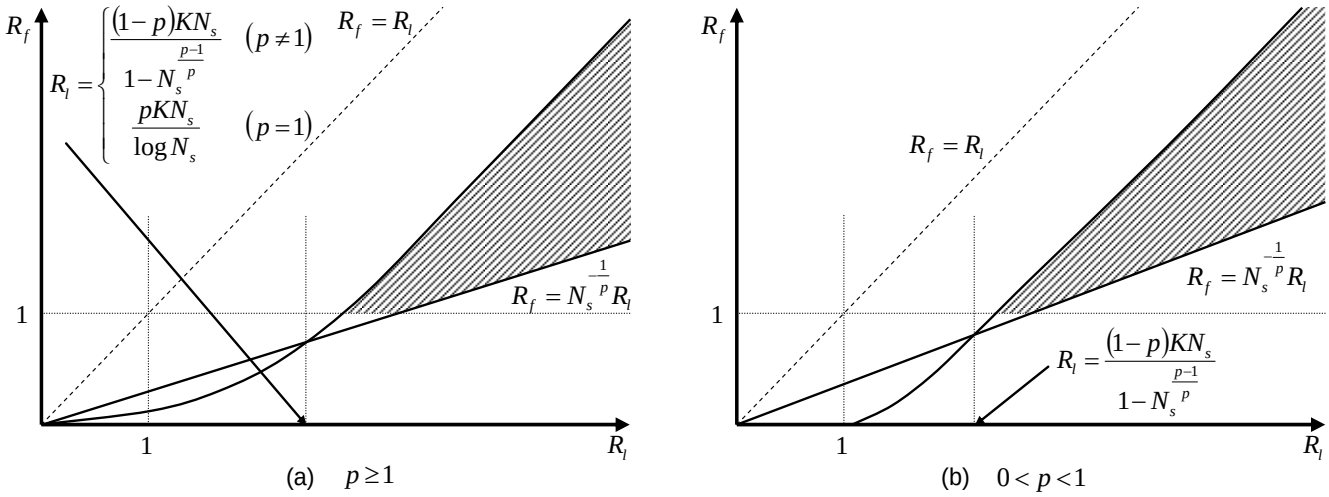

図 3　負荷分散条件とサーバ台数条件の交点となる R_f が 1 未満における二条件の関係

よって，R_f を R_f+1 に変化させたときの記憶域のサイズの変化 $m(R_f, R_l)$-$m(R_f$+1 $R_l)$は，条件(6)より，以下のように 0 以下となる．

$$m(R_f, R_l) - m(R_f + 1, R_l) = e(R_f)\left(\left\lceil \frac{R_l^p}{R_f^p} \right\rceil - N_s\right) \leq 0 \tag{12}$$

次に，R_l を R_l +1 に変化させたときの記憶域のサイズ $m(R_f$, R_l+1)は，次式となる．

$$m(R_f, R_l + 1) = N_s \sum_{r=1}^{R_f - 1} e(r) + \sum_{r=R_f}^{R_l} e(r)\left\lceil \frac{(R_l+1)^p}{r^p} \right\rceil + \sum_{r=R_l+1}^{N_r} e(r) \tag{13}$$

よって，R_l を R_l +1 に変化させたときの記憶域のサイズの変化 $m(R_f, R_l)$-$m(R_f, R_l$+1)は，以下のように 0 以下となる．

$$m(R_f, R_l) - m(R_f, R_l + 1) = \sum_{r=R_f}^{R_l - 1} e(r)\left(\left\lceil \frac{R_l^p}{r^p} \right\rceil - \left\lceil \frac{(R_l+1)^p}{r^p} \right\rceil\right) + e(R_l)\left(1 - \left\lceil \frac{(R_l+1)^p}{R_l^p} \right\rceil\right) \leq 0 \tag{14}$$

以上の議論から，R_f, および, R_l を減少させることにより，$e(r)$の定義によらず，記憶域のサイズを減少させることができることがわかる．よって，式(5)と(7)の等号が成立する条件の交点の近傍で，且つ，斜線の領域となる格子点を R_f と R_l の組合せとすることで，$e(r)$の定義によらず，記憶域のサイズを近似的に最小化することが可能となる．

次に，**図 3** は，式(9)の R_f の値が 1 未満の場合であり，同図の(a)，および，(b)は，それぞれ，$p \geqq 1$，および，$0<p<1$ の場合である．同図の斜線の領域は，式(5)と(7)の条件，$R_f \geqq 1$，および，$R_l \geqq 1$ の条件を満足する領域である．この場合も，**図 2** の場合の議論と同様に，$e(r)$の定義によらず，R_f と R_l が小さいほど，記憶域のサイズを近似的に最小化することが可能となる．

以上の議論から，式(5)と $R_f \geqq 1$ の等号が成立する条件の交点の近傍で，且つ，斜線の領域となる格子点を R_f と R_l の組合せとすることで，$e(r)$の定義によらず，記憶域のサイズを近似的に最小化することが可能となる．

5. 評価

5.1 評価項目および条件

本論文の提案方法の目的は，これまで議論してきたように，下記の三つである．

(1)　各サーバに必要な記憶域を削減すること
(2)　サーバ間の負荷の偏りを抑えること
(3)　サーバ間の記憶域サイズの偏りを抑えること

上記の目的の(1)については，下記の二つの負荷分散方法と比較する．

(A)　ラウンドロビンによる負荷分散
(B)　単純なラウンドロビンとコンシステントハッシングによる負荷分散方法(以下，二分割法と呼ぶ)

上記の(B)二分割法とは，提案方法のホットゾーンとノーマルゾーンをラウンドロビンによる負荷分散とし，コールドゾーンをコンシステントハッシングによる負荷分散とする方法として比較を行う．記憶域の削減効果については，章 5.2 で議論する．また，上記の目的の(2)については，章 3 で述べた，全サーバ台数に対する，ノーマルゾーンの分散度の総和の倍率をあらわす K の値を一定の値とすることで，サーバ間の負荷の偏りが抑えられることを示す．負荷の偏りを抑制する効果については，章 5.3 で議論する．

更に，上記の目的(3)の，各サーバに必要な記憶域のサイズの偏りについての評価結果については，章 5.4 で議論する．

評価のための条件としては，処理要求に含まれるキーについて，十分に大きな値として，1.5×10^7 を用いる．

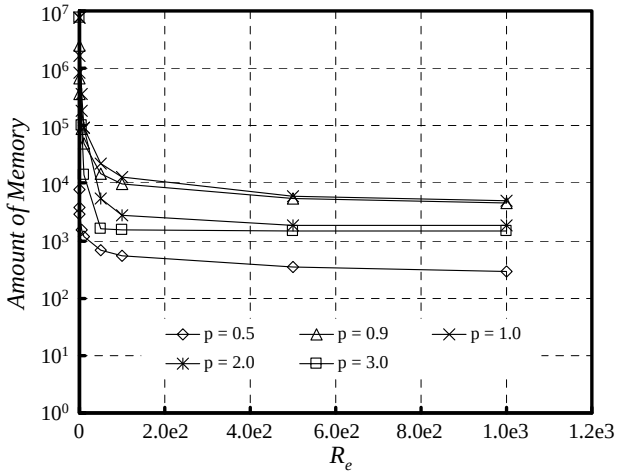

図 4　サーバに必要な記憶域のサイズ

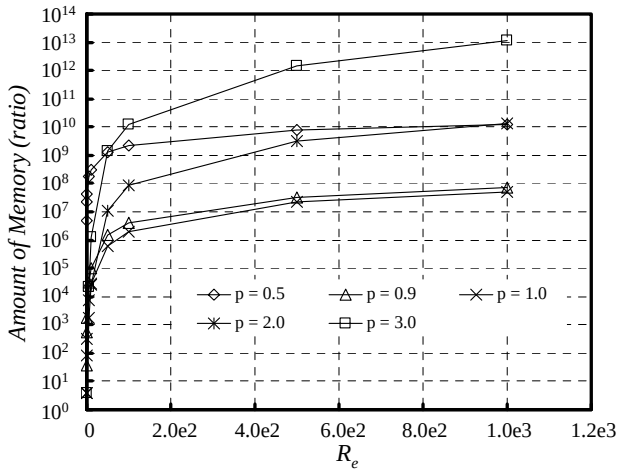

図 5　ラウンドロビンによる負荷分散を用いた場合のサーバに必要な記憶域のサイズの倍率

章 4 では，各キーを処理するためのデータサイズが，キーの要求頻度の変化に対して任意の変化をする場合においても，既存提案[20]のゾーン分割方法が利用可能であることを示した．すなわち，キーのランクの変化に対して，キーを処理するために必要なデータのサイズの変化が任意の場合にもゾーン分割方法が利用可能であることを意味している．本評価では，提案方法を章 1 で述べたネットワーク制御に用いることを想定し，キーの要求頻度が高いほど，比例してデータのサイズが大きい条件を考える．そこで，ランク r のデータサイズ $e(r)$ を，次のように定義する．

$$e(r)=\begin{cases}\dfrac{M-1}{R_b^p-1}\left(\dfrac{R_b^p}{r^p}-1\right)+1 & (1\le r<R_b)\\ 1 & (R_b\le r\le N_r)\end{cases} \qquad (15)$$

ここで，R_b，および，M は，それぞれ，キーを処理するために必要なデータのサイズが，そのランク以上で，最小量の一定値となるランク，および，ランク R_b のデータのサイズを 1 としたときの，ランク 1 のデータのサイズである．ランク R_b 以降で $e(r)$を一定としているのは，キーの要求頻度が一定以下に小さくなったとしても，そのキーを処理するために,最低限必要なキーのデータ量が存在することで,$e(r)$に下限が存在すると考えられるためである．上式は，要求頻度 $f(r)=A/r^p$ を用いて，$e(r)=pf(r)+q$ の形となり，要求頻度に比例するとともに，最小値が 1 であり，最大値が M となる．ここで，p および q は，定数である．

各ランクのデータのサイズを考えた場合, 要求頻度は, Zipf 分布によってランクの違いにより,劇的に変化する一方で,要求頻度に比例したサイズのデータが必要としても，現実的には，要求頻度の違いほどのデータサイズが必要な状況は考えにくいため，本評価では，最大値 M としては，1000 を考えるものとする．

5.2 記憶域の削減

上記(1)の評価結果として，まず，**図 4** を用いて提案方法を用いた場合に必要な記憶域のサイズについて説明する．

図 4 の横軸は，サーバ単体の性能が，下記の式の要求頻度を処理できる能力として表されるときの R_e を表す．

$$\frac{A}{R_e^p} \qquad (16)$$

ここで，A は，式(2)で用いているものと同じであり，上式は, ランク R_e の要求頻度と同じ処理要求を処理できる能力がサーバ単体の性能であることを表している．また，縦軸は，サーバ当たりに必要な記憶域のサイズであり，処理要求に含まれるキーに対する処理に必要な記憶域のサイズを単位(すなわち，この記憶域のサイズを 1)としている．ここで，**図 4** の評価では，章 3 で導入した K の値として，1.0×10^2 を用い，章 5.1 で導入した R_b の値として，150000(全キー数の 1%)を用いた．また，サーバ数 N_s は，全ての要求頻度の総和を，式(16)のサーバ単体の性能で割った値として計算した．本図で示した記憶域のサイズを，比較対象(A)のラウンドロビンによる負荷分散で必要となる記憶域のサイズと比較を行った結果について，**図 5** に示す．**図 5** の横軸は，**図 4** と同じであり，縦軸は，提案方法に必要な記憶域のサイズに対するラウンドロビンによる負荷分散に必要な記憶域のサイズの倍率を表している． ここで，**図 5** の評価でも，K と R_b の値としては，**図 4** の評価と同じ値を用いた．

図 5 の評価結果から，ラウンドロビンを用いた負荷分散に必要な記憶域のサイズと比較して，提案方法は，特にランクの小さい(要求頻度の高い)キーが，単一のサーバで処理できない範囲において，記憶域のサイズを大幅に改善できることがわかる．

第二に，上記の比較対象(B)との比較評価として，二分割法との，必要な記憶域のサイズの違いを**図 6** を用いて説明する．**図 6** 横軸は，**図 4** のものと同じであり，縦軸は，提案方法に必要な記憶域のサイズに対する二分割法に必要な記憶域のサイズの倍率を表している．ここで，**図 6** の評価

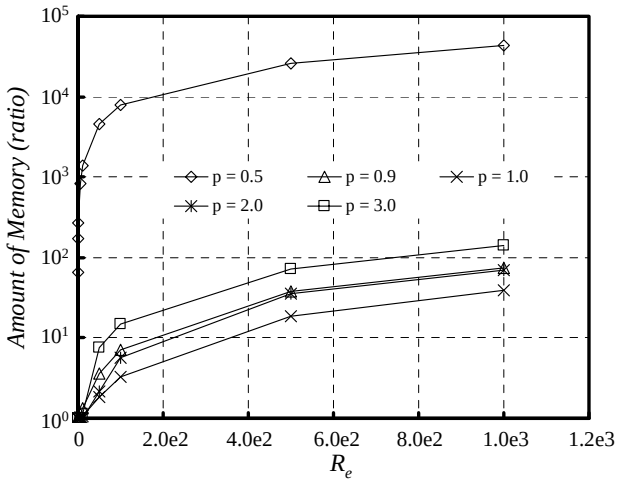

図 6　二分割法による負荷分散を用いた場合のサーバに必要な記憶域のサイズの倍率

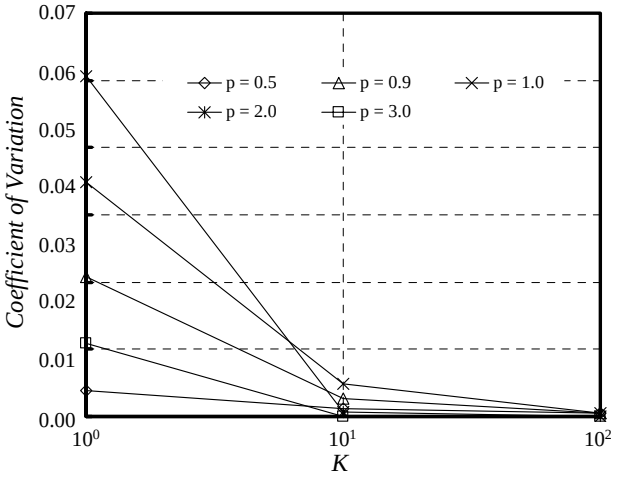

図 7　サーバ負荷の偏り(変動係数)の *K* による変化

でも，*K* と R_b の値としては，**図 4** の評価と同じ値を用いた．同図の評価結果から，特にランクの小さい(要求頻度の高い)キーが，単一のサーバで処理できない範囲において，提案方法に必要な記憶域のサイズは，二分割法と比較しても数倍以上の改善が得られることがわかる．

5.3 負荷の平均化

次に，章 5.1 で述べた評価目的(2)の評価結果として，サーバ間の負荷の偏りについての，*K* の変化による影響を，**図 7** を用いて議論する．**図 7** の評価では，R_e と R_b の値としては，10.0 と 150000 を用いた．**図 7** の横軸は，*K* であり，縦軸は，サーバ負荷の標準偏差をサーバ負荷の平均値で割った値である変動係数である．同図からわかるように，*K* の値が増加するに従い，変動係数が大幅に減少し，サーバ負荷の偏りが減少していることがわかる．特に，$K=10^2$ においては，負荷の変動係数が，1%以下と十分に小さい値を実現できていることがわかる．

以上の議論から，サーバ負荷の偏りを抑えながら，既存の負荷分散方法と比較して，大幅に必要な記憶域のサイズを

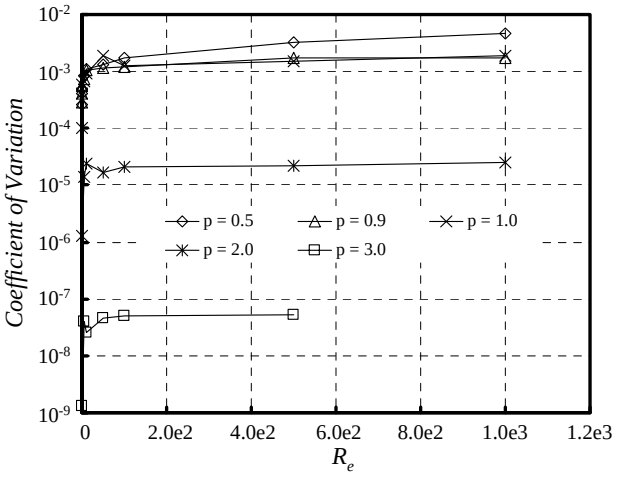

図 8　サーバ負荷の偏り(変動係数)

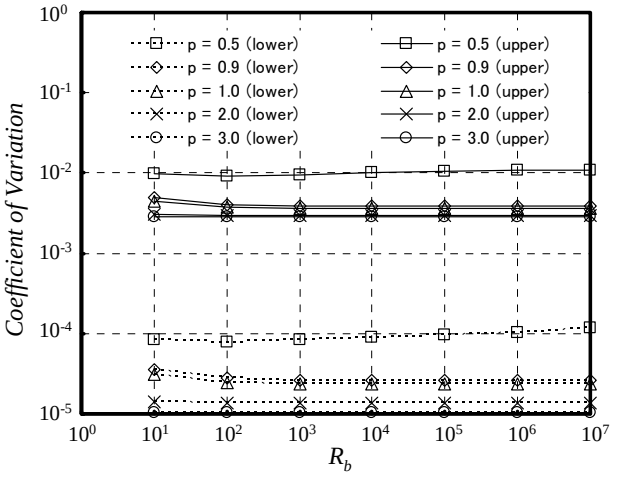

図 9　サーバに必要な記憶域サイズの偏り(変動係数)について無限母集団を仮定した場合の上限および下限

削減可能であることがわかる．

図 8 は，**図 7** と同様のサーバ負荷の標準偏差をサーバ負荷の平均値で割った値である変動係数が，R_e を変化させたときに，どのように変化するかを示している．ここで，*K* と R_b の値としては，**図 4** の評価と同じ値を用いた．本図からわかるように，全ての領域において，十分にサーバ負荷の偏りを十分に小さい値に抑えることができることがわかる．

5.4 サーバ間の記憶域サイズの偏り

各サーバに必要な記憶域のサイズについて，まず，ランク *r* のキーのデータ量 *e*(*r*)が，一定の場合について考える．**図 1** において，負荷の平均化をするために，図の中のサーバ割り当てを表す四角形を各サーバに均一に配置できる条件を，章 3 で求めた．このサーバ割り当てを表す四角形は，負荷の割り当てであると同時に，各サーバに四角形の数に比例したデータサイズが割り当てられることを意味している．よって，*e*(*r*)が一定の場合には，各サーバに必要な記

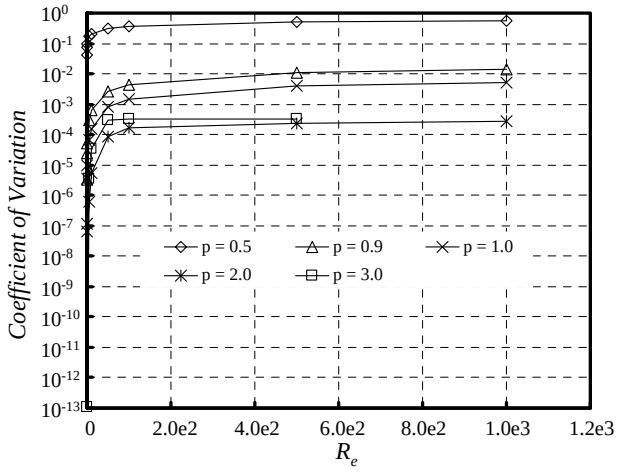

図 10　サーバに必要な記憶域サイズの偏り(変動係数)

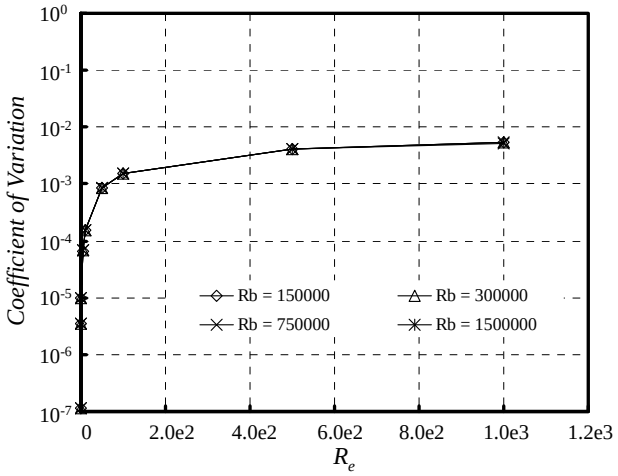

図 11　サーバに必要な記憶域サイズの偏り(変動係数)の R_b による違い

憶域のサイズについても平均化が実現できる．

次に，$e(r)$が，式(15)で表される値をとるときの各サーバに必要な記憶域のサイズについて議論する．章 5.1 で説明したように，$e(r)$は，要求頻度ほどの大きな変化はないと想定している．そのため，Zipf 分布の負荷分散に必要となる，本論文で提案しているような，特別な機能がなくても十分な数のランクをそれぞれのサーバが受け持つことで平均化できることが期待できる．そこで，$e(r)$の分布について R_b を変化させたときの変動係数(=標準偏差を平均値で割った値)の上限および下限を，**図 9** に示す．今，**図 9** のような変動係数を持った分布において，n 個の標本を選び，その平均値の分布を考えると，無限母集団とみなせる場合には，中心極限定理 [21]により，その変動係数は，**図 9** の $n^{-0.5}$ 倍となる．実際には，キーの母集団は有限であり，更に同一のキーを複数のサーバで持つ数も有限であることから，中心極限定理を用いた議論は正確ではないが，類推として利用すると，**図 9** より，変動係数は，最大でも 1%付近であることから，例えば 100 個のランクを受け持つサーバを考えた場合でも 0.1%付近の変動係数となり，必要な記憶域のサイズについても十分に平均化されることが期待される．

このことをシミュレーションにより明らかにするために，各サーバの必要記憶域サイズの偏りを変動係数として求めたものを**図 10** に示す．本図からわかるように，$p = 0.5$ 以外においては，変動係数が 1.0×10^{-2} 付近以下となっており，記憶域サイズの偏りは非常に小さい値となっていることがわかる．**図 10** において，$p = 0.5$ の場合においては，数十%の変動係数であり，サーバによって必要な記憶域のサイズが，相対的には大きく異なっている．しかし，これは，**図 4** からわかるように，$p = 0.5$ の場合，必要な記憶域サイズの絶対的な値が小さい為であり，利用上，大きな問題とはならないと考えられる．

図 11 は，各サーバの必要記憶域サイズの偏りについて，R_b による変化を表している．R_b の値としては，全体のキー数の 1%から 10%まで変化させたときの値を示している．本図から，R_b によらず，各サーバの必要記憶域サイズの偏り(変動係数)は，十分に小さい値となっていることがわかる．

以上の議論から，提案手法は，負荷分散の実現，既存手法と比較したときの記憶域サイズの大幅な削減，および，サーバ間の必要記憶域サイズの偏りの低減を十分に達成できていることがわかる．

6. まとめ

本論文では，最初に，これまで提案してきた要求頻度に大きな偏りをもつサービスの提供に利用可能な負荷分散方法の適用範囲として，処理要求に応答するために必要なデータのサイズが平均的に同一であるという既提案の条件から，処理要求に応答するために必要なデータのサイズの変化が，処理要求の頻度の変化に対して，任意の変化を行うという条件に拡張を行い，これまで提案してきた方法が適用可能であることを示した．次に，処理要求に応答するために必要なデータのサイズが，処理要求の頻度に比例する条件下で，記憶域削減効果，負荷分散，サーバ間の記憶域サイズの偏りの観点から評価を行った．その結果，記憶域削減効果について，提案方法に対し，ラウンドロビンによる負荷分散，および，単純なラウンドロビンとコンシステントハッシングの組合せによる負荷分散方法と比較し，大幅に，必要な記憶域のサイズを削減可能であることを示した．また，負荷分散およびサーバ間の記憶域サイズの偏りについても，実用上十分な範囲を実現できることを定量的に示した．

付録

章 3.2 の式(8)で表される R_l については，p の範囲について，$p > 1$，$p = 1$，および，$0 < p < 1$ の範囲のそれぞれにつき，R_l を N_s の関数として考え，R_l の最小値を，$N_s \geqq 2$ と $K \geqq 1$

の条件をもとにして，計算することで，容易に $R_l \geqq 2$ であることを導くことができる．

参考文献

1) P.A. Bernstein, V. Hadzilacos, and N. Goodman, Concurrency Control and Recovery in Database Systems. Addison-Wesley, 1987.
2) A. Helal, A. Heddaya, and B. Bhargava, Replication Techniques in Distributed Systems. Kluwer Academic Publishers, 1996.
3) R. Ladin, B. Liskov, and S. Ghemawat, “Providing High Availability Using Lazy Replication,” ACM Trans. Computer Systems, vol. 10, no. 4, pp. 360-391, 1992.
4) C. Pu and A. Leff, “Replica Control in Distributed Systems: An Asynchronous Approach,” Proc. ACM SIGMOD ’91, pp. 377-386, May 1991.
5) J.J. Fischer and A. Michael, “Sacrificing Serializability to Attain High Availability of Data in an Unreliable Network,” Proc. First ACM Symp. Principles of Database Systems, pp. 70-75, May 1982.
6) P. Cox and B.D. Noble, “Fast Reconciliations in Fluid Replication,” Proc. of International Conference on Distributed Computing Systems (ICDCS), pp. 449-458, 2001.
7) Z. Wang, S. K. Das, M. Kumar, and H. Shen, “Update Propagation through Replica Chain in Decentralized and Unstructured P2P Systems,” Proc. Int’l Conf. Peer-to-Peer Computing (P2P ’04), pp. 64-71 (2004).
8) F. M. Cuenca-Acuna, R. P. Martin, and T. D. Nguyen, “Autonomous Replication for High Availability in Unstructured P2P Systems,” Proc. of the 22nd IEEE International Symposium on Reliable Distributed Systems (SRDS), pp. 99-108 (2003).
9) V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, and P. Keleher, “Adaptive Replication in Peer-to-Peer Systems,” Proc. of the 24th IEEE International Conference on Distributed Computing Systems (ICDCS), pp. 360-369 (2004).
10) T. Yamashita, “Distributed View Divergence Control of Data Freshness in Replicated Database Systems,” IEEE Trans. on Knowledge and Data Engineering, vol. 21, no. 10, pp. 1403-1417 (2009).
11) T. Yamashita, “Dynamic Replica Control Based on Fairly Assigned Variation of Data with Weak Consistency for Loosely Coupled Distributed Systems,” Proc. of the 22nd IEEE International Conference on Distributed Computing Systems (ICDCS), pp. 280-289 (2002).
12) T. Yamashita and S. Ono, “View Divergence Control of Replicated Data Using Update Delay Estimation,” Proc. of the 18th IEEE Symposium on Reliable Distributed Systems (SRDS), pp. 102-111 (1999).
13) I. Foster and C. Kesselman eds., The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann (2003).
14) D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, “Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web,” Proc. of the twenty-ninth annual ACM symposium on Theory of computing, pp. 654-663 (1997).
15) G. K. Zipf, Human Behavior and the Principle of Least Effort, Addison-Wesley (1949).
16) L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: evidence and implications,” Proc of IEEE INFOCOM, pp. 126-134 (1999).
17) H. Kurita, N. Takada, T. Minami, T. Yamashita, and Y. Agawa, “Load Balancing of Requests with Keys Whose Frequencies Exhibit a Big-Head and Long-Tail Distribution,” Proc of the 9th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT 2012), SL-2-1 in Selected Session 2 (Distributed Systems) (2012).
18) 栗田弘之，高田直樹，南拓也，山下高生，阿川雄資，“要求頻度の偏りを考慮したスケーラビリティと負荷の平準化を両立する負荷分散手法の検討”，信学技報 IN2012-1-IN2012-11, vol. 112, no. 4, pp. 7-12 (IN2012-2) (2012).
19) 栗田弘之，高田直樹，南拓也，山下高生，阿川雄資，“要求頻度の偏りを考慮したコンシステントハッシュ法による負荷分散”，信学会総合大会論文集, no. 2, B-7-44, p. 205 (2012).
20) 山下高生，栗田弘之，高田直樹，南拓也，太田賢治，“Zipf 分布型の処理要求に適したスケールアウト手法における負荷分散と記憶域近似的最小化”，信学技報 IN2012-154-IN2012-215, vol. 112, no. 464, pp. 137-142 (IN2012-177) (2013).
21) 松原他，“統計学入門”，東京大学出版会 (1991).