様式Ｃ－１９

# 科学研究費助成事業（科学研究費補助金）研究成果報告書

平成　24　年　4　月　17　日現在

機関番号：１２６１２
研究種目：基盤研究（Ｃ）
研究期間：２００９～２０１１
課題番号：２１５０　００８
研究課題名（和文）　外部からの入力との相互作用を考慮したアルゴリズムの設計

研究課題名（英文）　Algorithm Design Considering Interaction with External Inputs

研究代表者
古賀　久志（KOGA HISASHI）
電気通信大学・大学院情報システム学研究科・准教授
研究者番号：４０３６１８３６

研究成果の概要（和文）：複数の計算機が相互作用する環境で発生する２つの問題を取り上げ、効率的に問題を解決するアルゴリズムの設計を行った。まず、インターネットにおいて、複数の TCP コネクション間の転送速度の公平性を改善するアルゴリズムを開発した。このアルゴリズムはルータへの修正を必要とせず、ユーザが持つ計算機の OS を置換するだけで実現できる。次に、複数計算機で構成される分散データベースにおける類似検索に取り組み、既存の単純な実装法より 10%程度応答時間を短縮するアルゴリズムを開発した。

研究成果の概要（英文）：We studied the two problems which arise in the environment where multiple computers interact with one another and designed effective algorithms for the problems. First, we constructed an algorithm which alleviates the throughput unfairness among different TCP connections. To deploy our algorithm, we have only to replace the operating systems in the end-hosts without modifying the routers.　Next, we dealt with the similarity search in the distributed database composed by multiple computers and devised an algorithm which makes the query response shorter by 10% than the known simple method.

交付決定額

（金額単位：円）

| | 直接経費 | 間接経費 | 合　計 |
|---|---|---|---|
| ２００９年度 | 700,000 | 210,000 | 910,000 |
| ２０１０年度 | 700,000 | 210,000 | 910,000 |
| ２０１１年度 | 500,000 | 150,000 | 650,000 |
| 年度 | | | |
| 年度 | | | |
| 総　計 | 1,900,000 | 570,000 | 2,470,000 |

研究分野：総合領域
科研費の分科・細目：情報学・情報学基礎
キーワード：アルゴリズム理論，分散コンピューティング，TCP プロトコル

## １．研究開始当初の背景

通常、アルゴリズム解析では、入力が完全に与えられると仮定する。しかし、現実のシステムでは不完全な入力を用いて問題を解

く状況が多々ある。例えば、1つの問題を複数計算機で解く分散環境では、各計算機には入力の一部のみ与えられ、他の計算機がどれだけ仕事をするかによって、自分への入力が変化する。また、オンライン環境では入力は時間経過と共に徐々に与えられる。

上記のような不完全な入力しか与えられない環境で効率的なアルゴリズムをどう設計するかは情報システム構築の重要なテーマである。研究代表者はオンライン環境で動くアルゴリズム（オンラインアルゴリズム）の解析を専門として来た。しかし、既存解析手法である競合比解析では、入力がオンラインアルゴリズムの振舞いに影響されないと仮定するので、その仮定が成立しない状況では良解を得るのが難しいとの印象を抱いていた。

２．研究の目的

本研究は、アルゴリズムの振舞いが外部からの入力に影響を与える環境で効率の良いアルゴリズムを設計することを目的とする。

３．研究の方法

実応用分野で発生する問題に対して、実用的にうまく動くヒューリスティックスアルゴリズムを開発する。そして、それらのアルゴリズムがどうしてうまく動くかを分析する。具体的には、

（1） 複数の TCP コネクション間のスループット公平性向上

（2） 複数計算機で構成される分散データベースにおける類似検索の高速化

という 2 つの研究課題に取り組む。

４．研究成果

（1）TCP コネクション間のスループット公平性向上

近年、インターネットに流入するトラフィック量が増加し続けており、輻輳制御技術の重要性が増している。 現在のインターネットでは、エンドホストでの TCP プロトコルによる輻輳回避機能によって輻輳制御は実現される。

しかし、TCP プロトコルでは、コネクションによってアグレッシブさが異なり、コネクション間でスループットが不公平になる問題が知られている。ここでアグレッシブさとは「送信レートを高く維持しようとする程度」という意味である。例えば、標準 TCP プロトコルである TCP Reno では、RTT (Round Trip Time) が短いコネクションの方がスループットが大きくなり、RTT 公平性が成立しない。また、高速 TCP プロトコルが標準 TCP プロトコルの帯域を奪う問題も存在する。

本研究では、ルータにおける輻輳制御機能である AQM (Active Queue Management)をエンドホストに移動することで TCP コネクション間のスループット公平性を向上することを提案する。

AQM とはルータにおいて輻輳の兆候を検出し、早期に輻輳通知を行ってエンドホストにトラフィック量を抑えるよう促す技術である。早期に輻輳通知することにより重輻輳を未然に防止する。最も代表的な AQM として、RED がある。RED は平均キュー長 $q$ に従って大きくなる確率によってエンドホストへ輻輳を通知する。ルータにおける AQM では多数のコネクションが処理対象なので、コネクション毎の状態管理は不可能であり、すべてのコネクションに対して等確率で輻輳を通知する。

これに対して、本研究では AQM 機能をエンドホストに移動する。エンドホストにおける AQM では 1 ホストから張られた TCP コネクションのみを処理すればよく、処理対象のコネクション数が非常に少なくなるので、ルータでは実現困難なコネクション毎の状態を維持するコネクションステートフル な AQM を実現できる。従って、各 TCP コネクションのアグレッシブさに応じて、コネクション毎に輻輳通知確率を変えることが可能になる。提案手法は、処理対象となる各 TCP コネクションのアグレッシブさを把握し、輻輳通知確率をアグレッシブなコネクションに対して大きく、アグレッシブでないコネクションに対しては小さくして、TCP コネクション間のスループット公平性向上を試みるエンドホストでの AQM アルゴリズムである。ここで TCP コネクションのアグレッシブさは TCP プロトコルと AQM が同じエンドホスト（の OS）内に実装されるので、容易に取得できる。提案アルゴリズムの動きは以下のようになる。

Step1：TCP の ack パケットを受信。ack が属する TCP コネクションを $i$ とする。
Step2: コネクション $i$ のアグレッシブさと RTT の増加値から定まる確率 $p$ で輻輳通知を行う。

Step2 では RTT の増加によって輻輳通知確率を変化させている。ルータにおける AQM が平均キュー長の伸びに従って輻輳通知確率を大きくする事象を、提案手法ではエンドホストで観測可能な RTT の増加に従って輻輳通知確率を大きくすることでエミュレーションする。

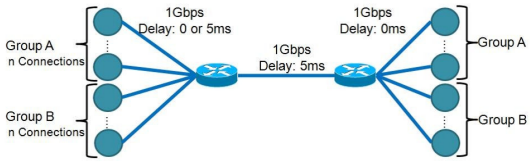


**図 1：ネットワークトポロジ**

ネットワークシミュレータ ns-2 を用いて提案アルゴリズムの評価実験を行った。ネットワークトポロジは図 1 に示すダンベルトポロジであり、2 台のルータ間がボトルネックリンクとなる。この上で、アグレッシブさが異なるグループ（Group A と Group B）の TCP コネクションを張り、Group A のコネクション群の平均スループットと Group B のコネクション群の平均スループットの比(以下、単にスループット比と記述する)で公平性を評価する。スループット比が 1 に近いほど公平性は高い。なお、Group A と Group B でコネクション数は同数 *n* (*n* はパラメータ)であり、1 つのエンドホストから張られるコネクション数はちょうど 1 とした。つまり、図の左サイドと右サイドに *2n* 個ずつ、合計で *4n* 個のエンドホストが存在する。

本実験では AIMD（Additive Increase Multiplicative Decrease）型の TCP コネクションに対してアグレッシブさを決定するパラメータを変化させて、アグレッシブさが異なるグループを構成した。AIMD は TCP の輻輳回避モードで送信レートを制御する基本方式である。輻輳を検知しない場合は加算的に輻輳ウィンドウ cwnd を増加して送信レートを上げ、輻輳検知時には乗算的に 1 度に大きく cwnd を下げて輻輳の解消を試みる。AIMD における cwnd の更新式を式(1)に示す。α, β は cwnd を増加，減少させるためのパラメータである．TCP Reno では α=1, β=0.5 である。α が大きいほど，また β が小さいほど，TCP コネクションはよりアグレッシブになる。

$$\begin{aligned} cwnd &= cwnd+\alpha/cwnd \quad (\text{ack 受信時}) \\ &\quad (1-\beta)cwnd \quad (\text{輻輳時}) \end{aligned} \qquad (1)$$

Group A と Group B とで(実験 1) α の値が異なる場合、(実験 2)β の値が異なる場合、(実験 3) リンク遅延が異なる場合の 3 パターンで実験を行った。実験条件を以下に示す。

実験 1: グループ A の α=1、グループ B の α=5 に設定した。β 及び往復リンク遅延は全コネクションで共通で、β=0.5、(往復リンク遅延)=10ms である。

実験 2: グループ A の β=0.5、グループ B の β=0.1 に設定した。全コネクションで共通に α=1、(往復リンク遅延)=10ms である。

実験 3: (グループ A の往復リンク遅延)=20ms、(グループ B の往復リンク遅延)=10ms に設定した。全コネクションで α=1、β=0.5 である。

とくに提案手法とルータで従来手法の ECN ベースの RED を走らせた場合とを比較する。RED のパラメータは ns-2 のデフォルト値を用いた。

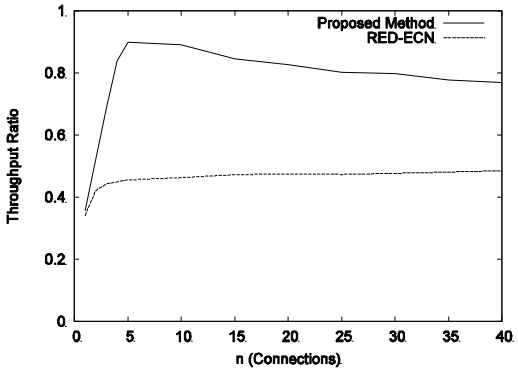


**図 2：α が異なるコネクション間のスループット比**

実験 1 ではすべての *n* に対して，提案手法が RED より高い公平性を達成した（図 2）。RED のスループット比は常に 50%以下である。逆に提案手法の公平性は *n* の値によって変動する。スループット比は *n*=1 の時は 35%にとどまるが、*n*=5 まで大きくなると 90%まで上昇し高いスループット公平性が達成される。さらに *n* が増えるとスループット公平性が下がり、*n*=40 でスループット比が 78%になった。

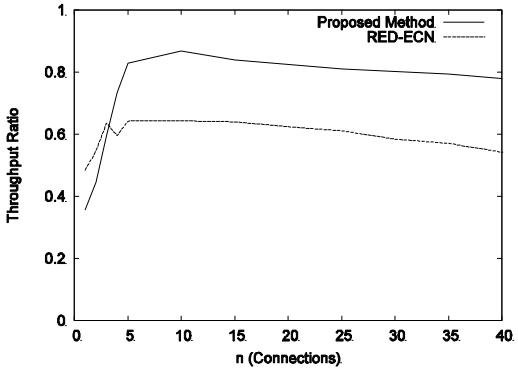


**図 3：β が異なるコネクション間のスループット比**

実験 2 では提案手法のスループット比は実験 1 と同様である（図 3）。しかし、RED の公平性が実験 1 より良いので、*n* が小さい時、提案手法の公平性が RED を下回った。*n*>=4

では、提案手法の公平性は RED を上回る。

実験 3 ではすべての $n$ に対して提案手法が RED より高い公平性を示した（図４）。実験 1, 2 と異なり本実験では、$n$ が大きいほど、提案手法のスループット比が 1 に近づく。

上記より提案手法は実験 2 で n<=3 の時以外は、RED より TCP コネクション間の公平性を向上できた。

提案手法はルータを修正しないので、インターネットで展開するのが容易であるという利点も有する。

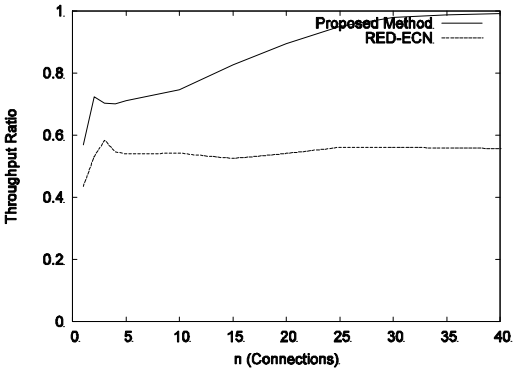


**図 4: 往復遅延が異なるコネクション間のスループット比**

（２）分散データベースの類似検索高速化

複数計算機で構成された分散データベースにおける類似検索の応答時間短縮に取り組んだ。類似検索アルゴリズムとしては Locality-Sensitive Hashing (LSH) を取り上げた。

LSHはハッシュベースの手法であり、近似的に最近接点を求めることで高次元データに対しても計算量を抑える確率的アルゴリズムである。LSHにおけるハッシュ関数は、高次元空間を空間軸に直交するc個(cはパラメータ)のランダムに選択された超平面でセルに分割することで実現され、それぞれのセルがハッシュバケツ（以下バケツ）となる。ハッシュの構成法より、LSHでは空間に近接した類似データ同士が高確率で同じハッシュ値となる。確率的な近接点の見落としを防止するためにLSHではこのようなハッシュ関数を複数個用意する。以下ではハッシュ関数の個数を$l$で表し、$l$個のハッシュ関数を$g1,g2,\cdots,gl$と記述する。LSHは、検索時にはクエリと同じハッシュ値を持つバケツ内のデータを最近接点の候補とし、類似度計算をこの候補に対してのみ行うことでオーバーヘッドを削減する。LSHは高速である反面、ハッシュテーブルが複数個必要なため空間計算量が大きい。従って、1台の計算機で大規模なLSHを実現するとハッシュテーブルがメモリから溢れて著しく性能が劣化する。そのため、大規模なデータに対しては、LSHを複数計算機に分散して実現する技術が必要になる。

本研究では、LSHの$l$ 個のハッシュテーブルを $n$ 個の計算機（以下ノード）に分散する状況を対象とする。この環境で単純にLSHを実現する手法としては、1個のノードに均等に $l/n$ 個ずつハッシュテーブルを格納する手法がまず考えられる。しかし、この方法ではクエリ処理時に $l$ 個のハッシュテーブルにアクセスするために、$l>n$ であれば全ノードへのリモートアクセスが必要となる。従って、この方式では通信がボトルネックとなる分散環境においてクエリへの応答時間が長くなる。以下ではこの方式を単純法と呼ぶ。

これに対して我々のアルゴリズムではクエリ処理時にアクセスされる$l$個のバケツをなるべく同じノード上に配置する。これにより、1回のリモートアクセスで複数のバケツにアクセスできるのでリモートアクセス回数を減らせる。LSHにおける$i$番目のハッシュテーブルのハッシュ値 $v$ のバケツを$(i, v)$と記述すると、クエリ$q$に対する最近接点探索では

$$(1, g1(q)), (2,g2(q)),\cdots,(l,gl(q))$$

の $l$ 個のバケツがアクセスされる。従って、これらのバケツ群を同じノード上に配置すればよい。提案手法では、同一のデータを含むバケツが同じハッシュ値をとるようなバケツに対するハッシュ関数 BH を構築し、BH のハッシュ値に基づいてバケツをノードへ割り当てることでこのアイデアを実現する。同一のデータ $x$ を含むバケツ群は異なるハッシュテーブル上に存在する。しかし、これらのバケツは全部 $x$ を包含し、ユークリッド空間上で重なっており近接している。従って、それぞれの重心も互いに近い。例えば、図５では(a)が g1 による空間分割、(b)が g2 による空間分割を示す。$x$ を内包するセルがバケツ$(1,g1(x))$、$(2,g2(x))$を表す。$(1,g1(x))$と$(2,g2(x))$は共に $x$ を含むので空間的に近接し重心も近い。この性質から重心が近いバケツが同じハッシュ値となるハッシュ関数を作れば、目的の BH を構成できる。これは、空間的に近い点が同じハッシュ値となるハッシュ関数であり、まさに LSH に他ならない。そこで、空間分割を行う超平面数を $cb(< c)$とした LSH 関数を BH とする。超平面数を減少させることで、BH のセルはハッシュ関数 $gi$ $(1<i<l)$ のセルより大きくなり、図５(c)のように同じデータを含むバケツ群の重心座標が同一の BH のハッシュ値を取ることが期待できる。提案手法では同一ノードに複数のハッシュ

テーブルのバケツが混合して存在する。この性質から提案手法を MIXED-LSH と呼ぶ。

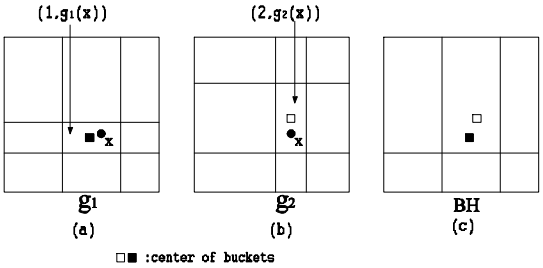


**図 5：異なるハッシュテーブル上の同じ点を含むバケツ**

実機環境でMIXED-LSHと単純法の応答時間の比較実験を行った。実験には画像データベース Caltech256 の画像を色ヒストグラムに変換することで得られる24次元の実データを使用した。まず、各画素のRGB値からR(Red)、 G(Green)、 B(Blue)の3つの色頻度ヒストグラムを構築する。そして、それぞれの頻度ヒストグラムの横軸を32ずつ8個のビンに分割して8次元ベクトルを得る。最後にR、G、Bに対応する8次元ベクトルを連結して8x3=24次元ベクトルを求める。Caltech256は30607枚の画像からなり、30607個の24次元ベクトルが得られる。これから任意に選んだ25000個のデータをデータベースに登録後、残りの5607個のデータから2000個をクエリとして選択し、k近傍探索を行い性能評価を行った。実験ではk=5とした。

実機環境は5台のサーバノード(Linux，CPU: Core2Duo，2.93GHz，メモリ: 2GB)と1台の検索クライアント(Linux，CPU: Intel i7 2.8GHz，メモリ: 2GB)で構成される。サーバノード・クライアント間通信はTCPソケットで実現され、応答時間を短縮するために、TCP_NODELAYソケットオプションを指定した。サーバノード・クライアント間のネットワークは別のPC上で動作するLinuxカーネル標準のnetemネットワークエミュレータで実現した。とくに5台のサーバノードとクライアント間の往復通信遅延は、地域レベルの広域網を想定して4ms、8ms、12ms、16ms、20msに設定した。また、サーバノード・クライアント間のボトルネック物理帯域は1Gb/sである。

図 6 に 2000 個のクエリ処理をした時の応答時間を示す。ハッシュ関数の数 $l$ を 5 から 20 まで 5 ずつ変更して実験を行った。MIXED-LSH は単純法より応答時間を $l$=5 の時 17%、$l$=20 の時 9.3%短縮できた。図 7 に本実験において発生したリモートアクセス回数を示す。図 6 、図 7 より、リモートアクセス回数と応答時間には正の相関があり、MIXED-LSH がリモートアクセス回数を削減することにより応答時間を短縮できたと言える。

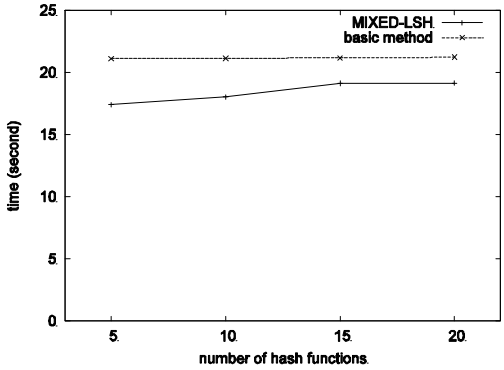


**図 6 ：応答時間**

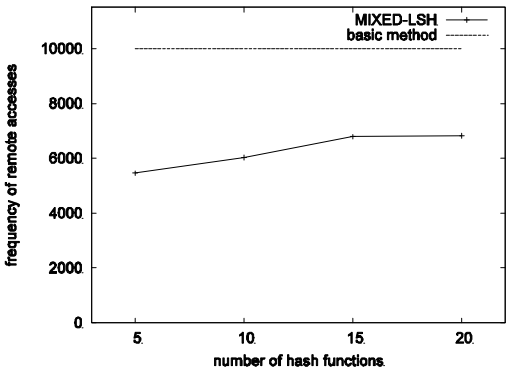


**図 7 ：リモートアクセス**

MIXED-LSHが応答時間を短縮できる理由を述べる。簡単のために$l$>$n$の場合とする。$T$maxをクライアントとサーバノード間の最大の往復遅延時間とする。単純法ではクライアントは全サーバノードにアクセスするので、応答時間は$T$maxを下回ることはない。一方MIXED-LSHは通信遅延が最大となるサーバノードにアクセスしない場合に応答時間が$T$maxより短くなる。

従来のP2Pの分散システムにおける検索はファイル名とファイル生成時間などを検索キーとして用いて、同一ファイル検索を目的とする技術が多い。これに対し、提案手法はコンテンツに基づいた類似ファイルの高速検索を実現するものであり、例えば似た画像・音楽ファイルを検索する基盤技術となりえる。

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計３件）

① H. Koga, M.Oguri and T. Watanabe, MIXED-LSH: “Reduction of Remote Accesses in Distributed Locality Sensitive Hashing based on L1 Distance". Proc. 26th IEEE International Conference on Advanced Information Networking and Applications (AINA2012), IEEE Computer Society, pp.175-182, 2012. 査読有

② 古賀久志、渡辺俊典, "分散環境における L1 距離ベース Locality-Sensitive Hashing の通信回数削減手法とその実装評価", 信学技報 DE2011-40, pp.1-6. 2011. 査読無

③ 石津圭太、古賀 久志、渡辺 俊典, "エンドホストでの AQM エミュレーションによる TCP コネクション間のスループット公平性改善", 信学技法 IN2010-204, pp. 359-364, 2011. 査読無

〔学会発表〕（計１件）

① 小栗正之、古賀久志、渡辺俊典、分散環境での L1 距離ベース Locality-Sensitive Hashing のリモートアクセスの回数削減, 第3回データ工学と情報マネジメントに関するフォーラム(DEIM), 2011/3/5 修善寺.

６．研究組織

(1)研究代表者

古賀　久志（KOGA HISASHI）

電気通信大学・大学院情報システム学研究科・准教授

研究者番号：４０３６１８３６