2つのHTTP-GETパケットによるWebサーバの負荷推定

Estimation of Web Server Load by Two HTTP-GET Packets

前田 亨, 豊泉 洋*

概 要

Webサーバにおいて特に準備をすることなく、リモートからその負荷を観測、評価する方法を提案する。2つのHTTPパケットをサーバとその1ホップ前のルータに送信し、その応答を受信するまでのそれぞれの往復の時間の差と、その統計的な分散を利用し、サーバでリクエストが処理されるのを待つ時間を抽出する。また、この方法を実装したプログラムを用いた実験結果も示す。

1 はじめに

インターネットは既に社会の仕組みの一部に取り入れられ、それなしでは仕事に支障が出るという人も多いと思われる。そこで最近では、重要なサーバを堅牢なインフラをもつIDC(Internet Data Center)にハウジングして集中管理することが多い。そのような自組織の設備の中にないサーバの状態を把握するには、そのマシンにあらかじめ何らかの仕組みを組み込んでおくか、リモートから既存の仕組みを用いてサーバの状態を反映するパラメータを測定する必要がある。この論文では、そのパラメータとしてサーバプロセスの負荷に注目する。

サーバの負荷を観測する方法は数種類存在する。その中でも、そのサーバに特別な用意をしていなくても実現できる方法として、pingの応答時間を利用したものが実際に使われている。しかし、最近ではpingへの応答を行わないサーバも多い。また、pingへの応答はOSが行うので、アプリケーションプロセス自体の負荷を観測できない、応答時間が経路での遅延を含むため、サーバの負荷を直接表しているとは言えないといった問題がある。後者の問題を解決するために、2つのパケットを使って観測する方法[1, 2]が考案され、さらにその後、アプリケーションプロセスを対象とする拡張がなされた[3]。しかしこの方法は、純粋に負荷を測定しているとは言えなかった。そこでこの論文では、この点を改良するために2つのパケットを、サーバとその1ホップ前のルータに送信した場合のそれぞれの往復の時間差と、統計的な分散を使う方法を提案する。

なお、サーバではなくネットワークの負荷などを観測するためのツール等は、Springによって多数紹介されている[4]。

*T. Maeda and H. Toyoizumi are with Performance Evaluation Laboratory, University of Aizu, Aizu-Wakamatsushi, Japan 965-8580. E-mail: m5061234@u-aizu.ac.jp, toyo@u-aizu.ac.jp

2 ネットワークのモデル

この論文では、図1のようなネットワークを想定している。

観測対象となるサーバ(以下サーバと呼ぶ)と、観測を行う遠隔マシン(以下オブザーバと呼ぶ)の間にはインターネットなどのIPネットワークがある。サーバ、オブザーバ間の経路にはn個のルータがある。

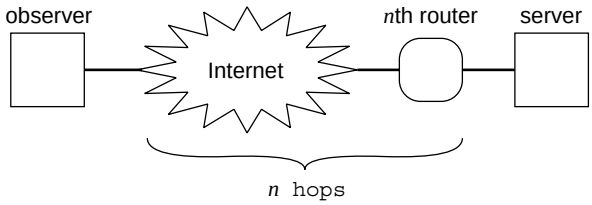

図1: 想定しているネットワーク

サーバとしては、最も一般的ということで、今回はWebサーバを選択した。実際には、適当なリクエストに対して同一の処理を行い、毎回同じようなレスポンスを返すプロトコルに沿ったサーバプロセスであれば、この方法を適用することができる。ここで言うリクエストとはHTTPならばGETなど、SMTPならばHELOなどのことを指す。リクエストを工夫することにより、データベースと連携しているようなシステムで、そのシステム全体を診断対象とするなどというような応用も可能である。

3 待ち時間の分散推定

この方法では、まずオブザーバからサーバおよびその1ホップ前のルータにほぼ同時にHTTPパケットを

送り、それらのパケットへの応答が戻ってくるまでの往復の時間 (RTT) を計測し、それらの差を求める。

リクエストがサーバで処理を待たされる時間は、サーバの負荷を表していると考えられるので、それを取り出すために RTT の差の統計的な分散を求める。RTT の差を使うことにより、ネットワーク内での遅延の揺らぎの影響を少なくできる。さらに分散を使うことにより、固定的な値を無視することができるので、サーバでの待ち時間の分散のみが得られる。

3.1 2 つの RTT の差を求める

図 2 は 2 つのリクエストとそれぞれへの応答の時間的な流れを表している。

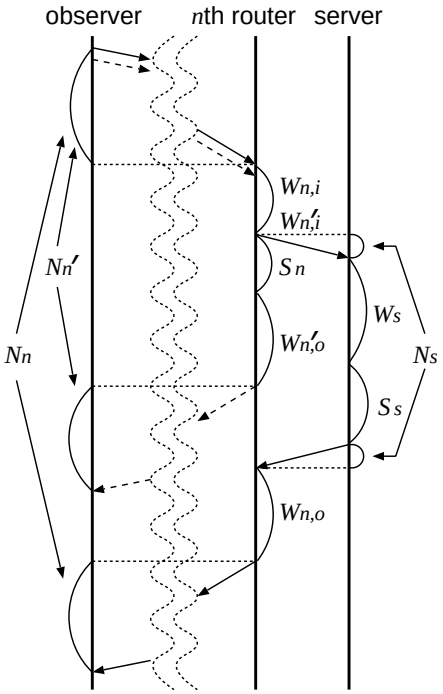

図 2: 2 つのパケットの流れ

(1) サーバへ送った場合の RTT を R_s とすると、R_s は次のように分解される。

$$R_s = N_n + W_{n,i} + N_s + W_s + S_s + W_{n,o} \quad (1)$$

N_n: n 番目のルータ (1 ホップ前のルータ) までの往復時間。ネットワーク遅延と、$n-1$ 個のルータでの待ち時間の累積からなる

$W_{n,i}$: n 番目のルータで、サーバに転送されるまでに待たされる時間

N_s: n 番目のルータとサーバ間の往復のネットワーク遅延

W_s: リクエストがサーバで処理されるのを待つ時間

S_s: サーバがリクエストの処理にかかる時間

$W_{n,o}$: 応答パケットが n 番目のルータで転送されるのを待つ時間

普通のサイトでは n 番目のルータ、サーバ間の帯域が十分に大きいと思われる。その時には N_s はほぼ一定の値になると思われるので、そのように仮定する。S_s もリクエストの内容が同じであれば、いつも一定の値になると仮定する。

(2) n 番目のルータに送信した場合の RTT を R_n とする。R_n は次のようになる。

$$R_n = N'_n + W'_{n,i} + S_n + W'_{n,o} \quad (2)$$

N'_n: N_n と同じだが、異なるパケットなので区別する

$W'_{n,i}$: $W_{n,i}$ と同じ。N'_n 同様区別する

S_n: n 番目のルータがパケットを処理するのにかかる時間

$W'_{n,o}$: $W_{n,o}$ と同じ。N'_n 同様区別する

S_n は同じパケットに対しては、いつも一定の値になると思われる。

(3) W_s をうまく取り出すために、まず R_s を観測するためのパケットと、R_n を観測するためのパケットを送信する時間をなるべく近づける。同じ時間帯ならば、ネットワークの状態はそれほど異なっていないと予想されるので

$$N_n \simeq N'_n, \quad W_{n,i} \simeq W'_{n,i}, \quad W_{n,o} \simeq W'_{n,o} \quad (3)$$

と仮定する。この仮定の元では

$$R_s - R_n \simeq N_s + W_s + S_s - S_n \quad (4)$$

となる。

3.2 待ち時間の分散を求める

$R_s - R_n$ を多数回観測することにより、その分散 $Var\,[R_s - R_n]$ を求めることができる。ところで、N_s、S_s、S_n は W_s と独立であり、ほぼ一定なので

$$Var\,[R_s - R_n] \quad \simeq \quad Var\,[W_s] \quad (5)$$

4　負荷の評価方法

(4) 式より、α は次の式を満たす諸々の定数と変数の和であるとする。

$$R_s - R_n = W_s + \alpha \tag{6}$$

$R_s - R_n$ の平均 $E[R_s - R_n]$ は、$R_s - R_n$ を多数回観測すると得られる。また、もし W_s が指数分布に従っているとすると、$E[W_s]$ は $\sqrt{Var[W_s]}$ となる [5]。つまり、α は次のように推定することができる。

$$\alpha = E[R_s - R_n] - E[W_s] \tag{7}$$

ある時点で観測された $R_s - R_n$ から α を引くことによりその時の W_s を見積もる。

W_s は指数分布に従うとしているので、$\lambda = 1/E[W_s]$ とすると

$$P\{W_s > t\} = P\{R_s - R_n - \alpha > t\} = e^{-\lambda t} \tag{8}$$

これを用いると、ある確率 θ をしきい値として

$$e^{-\lambda W_s} < \theta \tag{9}$$

となる W_s が、連続 5 回観測されたら警告を発するなどという応用が考えられる。

5　実際の流れ

今回は送信する 2 つのパケットの内容を HTTP の GET リクエスト [6] にした。片方のパケットは n 番目のルータで TTL が 0 になるようにしてあり、そのルータが ICMP Time Exceeded を返してくる [7] ことを利用して R_n を得た。1 回の観測は次の流れに沿って行われる。

1. オブザーバとサーバの間でコネクションを確立する (3-Way ハンドシェーク [8] を完了させる)
2. 2 つのパケットを送信する
3. n 番目のルータからとサーバからの返信を受け取る
4. コネクションを切断する

6　実用に際しての課題

この方法を用いた場合に問題となると思われる点を考えてみる。

まず、ICMP パケットが経路のルータでどう扱われているかによって、R_n の意味が変わる可能性があることが挙げられる。これは n 番目のルータが送り返してくるパケットが ICMP であることによる。例えば、ICMP とサーバアプリケーションのプロトコルのパケットを、優先制御などで区別しているルータが経路の途中にあった場合には、N_n と N'_n が大きく異なる値となる可能性がある。

次に、S_s と S_n を、観測した時によらない定数として扱っている点が挙げられる。いずれも、その時点でサーバまたは n 番目のルータで行っている処理によっては大きく変動する可能性がある。S_s に関しては、その変動がサーバの負荷によるものとして、W_s が変化したと考えると問題にはならない。しかし、S_n は n 番目のルータがルーティングプロトコルの処理などで CPU 使用率が高い状態では変動する可能性があり、S_s のように他の変数がその変化を肩代わりすることもできない。

さらに (3) 式が、どの程度信頼できるかも問題になる。$W_{n,i} \simeq W'_{n,i}$ は、2 つのパケットを送信した時刻が近ければ概ね成り立つと思われるが、他の 2 つの関係は、2 つのパケットが返信される時刻が離れている可能性があることから、成り立っていないかもしれない。

最後に、実装時に考慮すべき点としての問題点を挙げる。負荷を評価する際に、2 つのパケットの時間差から W_s を取り出すために用いる α は (7) 式により求める。この式で用いる 2 つの平均を、どの範囲のデータから得るべきかが問題となる。どのくらいの間隔で観測を行うかにもよるが、普通は直前の数十程度のデータを用いると思われる。その数は平均として意味のある値を算出できるほどには大きくなければならず、かつその時点でのネットワークの状態を適切に表現している値を導くことのできるものでなくてはならない。時間的により近いデータに重みを与えるという方法も有効かもしれない。

7　実験

前述の方法を実装したプログラムを用いて、実験を行った。オブザーバは C で書かれた観測用のプログラムを実行する。サーバとしては Web サーバの Apache[9] を用いたが、この Apache はリクエストが実際にどのくらい待たされたのかを記録できるようにハックしてある。

7.1　1 ホップだけを挟んだ場合

まず、間に 1 ホップだけを挟んだ場合で実験を行った。

構築したネットワークは図 3 のようなもので、オブザーバとサーバがルータを 1 つだけ挟んで接続されて

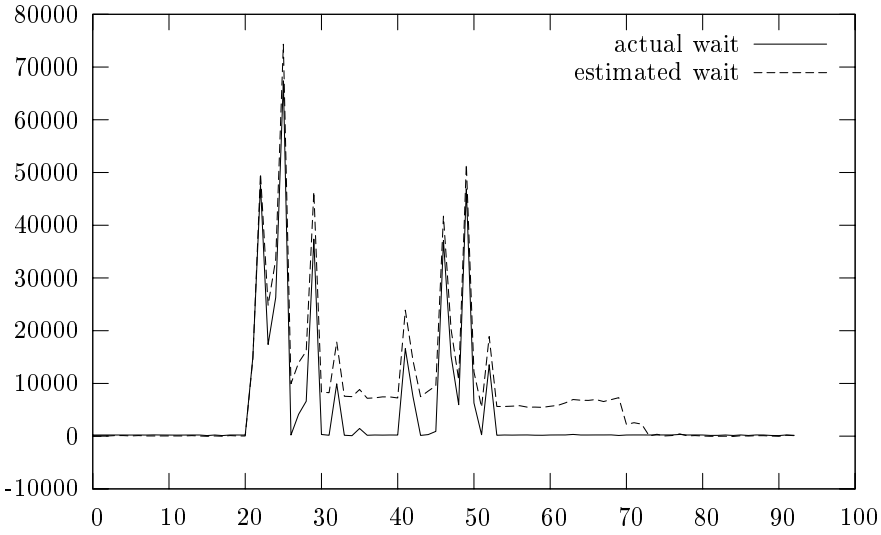

図 4: 1 ホップだけを挟んだ場合

いる。このネットワークは閉じたネットワークであり、インターネットなどの公共のネットワークにはつながっていない。

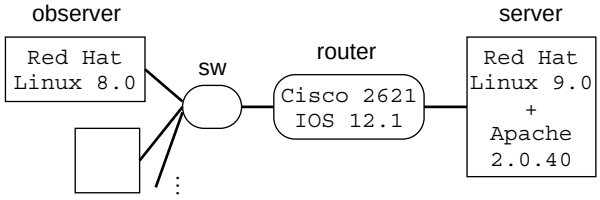

図 3: 実験に使ったネットワーク

この実験ではオブザーバは 3 秒おきに観測を行った。最初の 20 回で得られた $R_s - R_n$ をもとに $Var\,[W_s]$ を求め、以降の観測では直前 20 個の $R_s - R_n$ から毎回 $Var\,[W_s]$ を再計算した。同様に (7) 式の α は、計算に用いる $R_s - R_n$ の範囲を、その観測の直前 20 回にしている。

図 4 が実験の結果を表している。縦軸はマイクロ秒での見積もられた待ち時間を示している。

実際に待たされた時間に見積もられた時間が連動していることが読み取れる。長い待ちを観測した後で、しばらく実際よりも見積もりが大きくなってしまうのは、α を求めるためのデータにその長い待ちが含まれた結果、α がそれに引きずられて大きな負の値になってしまうことによる。

この結果から、単純なネットワークにおいては、この方法での負荷の観測が可能であろうということがわかった。

7.2 インターネットを挟んだ場合

次にインターネットを介してオブザーバとサーバが通信している場合の実験を行った。機器の構成と観測の条件は、オブザーバと n 番目のルータ間にインターネットが入った以外に変わりはない。実験を行った時点ではオブザーバ-サーバ間には n 番目のルータを含めて 17 ホップあった。

図 5 がその結果を表している。2ヶ所大きく見積もりが外れてしまい、また、それら以降が引きずられて大きめの値を算出してしまっているが、実際の待ち時間に沿って見積もられた待ち時間が遷移していることがわかる。

同じ構成でもう一度実験を行った結果を図 6 に示す。RTT として表されているのは、リクエストを送信してからサーバが返信してきたレスポンスを受け取るまでの単純な往復時間で、これを見ると RTT 中の余計な時間を打ち消して、実際の待ち時間にかなり近い値を見積もれていることがわかる。

7.3 サーバの前が複雑なネットワークの場合

最後に、サーバとその 1 ホップ前のルータとの間に複雑なレイヤー 2 のネットワークがある場合の実験結果を図 7 に示す。これは B フレッツ [10] をアクセス回線に用いて、ブリッジ接続にてサーバを接続した場合の結果で、意味のある値がまったく見積もられていない。

サーバと 1 ホップ前のルータ間にあるネットワーク (この場合は地域 IP 網) が複雑なため、時間的な揺らぎが大きくなったことにより、セクション 6 で述べた (3) 式の仮定が大幅に崩れたことがこの原因だと考えられる。また、1 ホップ前のルータ (地域 IP 網からパケットを受け取る ISP のルータ) は、非常に多数のユーザ

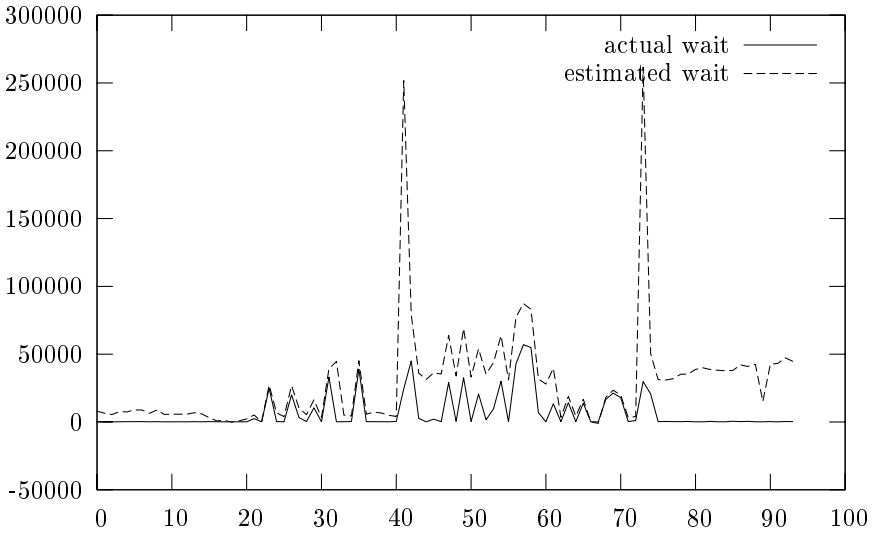

図 5: インターネットを挟んだ場合 1

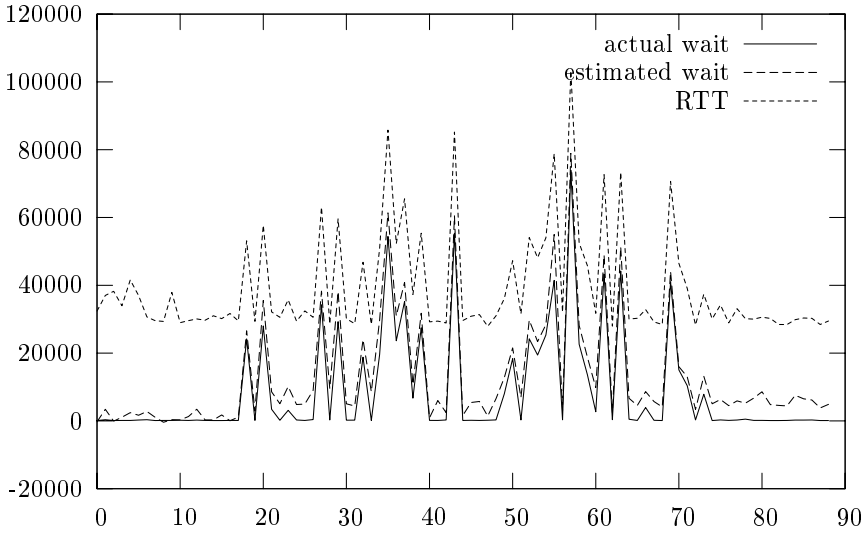

図 6: インターネットを挟んだ場合 2

を収容していると思われるので、そこでもかなり揺らぎが発生しているかもしれない。

8 まとめ

本論文では、2つのパケットを観測対象となるサーバとその1ホップ前のルータに送信し、それぞれからの応答の往復の時間を測定することでサーバの負荷を見積もる方法を提案した。この方法はアプリケーションプロセスのレベルで負荷を予想することができ、多くのプロトコルに対応する。

数例の実験により、実際に有用であることが示された。ただし、1ホップ前のルータまでのアクセスには注意をする必要がある。また、実験で用いたアルゴリズムには、大きな値にその後の値が引きずられてしまうという欠点があったので、それを改善することが今後の課題である。

参考文献

[1] N. Mizutani, “Computer System Performance Observation through the Computer Network,” master’s thesis of the University of Aizu, 2002

[2] N. Mizutani, “Detection of Hacking by Observing Traffic on the Network,” Proceedings of the 2001 Communications Society Conference of IEICE, pp. 17-18

[3] D. Azami, “Detection of the distributed denial of service attack,” graduation thesis of the University of Aizu, 2002

[4] N. Spring, “Reverse Engineering the Internet,” `http://www.cs.washington.edu/homes/nspring/papers/generals.pdf`

[5] L. Kleinrock, “Queueing Systems Volume I: Theory,” John Wiley & Sons, Inc.

[6] “Hypertext Transfer Protocol - HTTP/1.1,” RFC 2616

[7] “Internet Control Message Protocol,” RFC 792

[8] “Transmission Control Protocol,” RFC 793

[9] The Apache Software Foundation, `http://www.apache.org/`

[10] Flets Top Page, `http://flets.com/opt/`

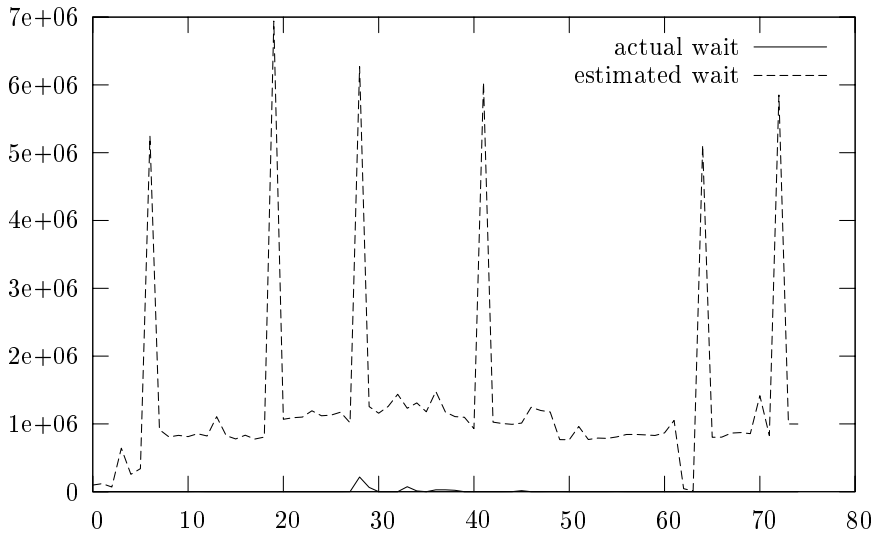

図 7: サーバの前が複雑なネットワークの場合