A Combined Adaptive Approach for Congestion Control in the Transmission Control Protocol

Plamen Z. Zahariev, Georgi V. Hristov, Teodor B. Iliev
Department of Communication Systems and Technologies, University of Ruse, Ruse, Bulgaria
E-mail: {pzahariev, ghristov, tiliev}@ccnrgroup.com

Abstract. Due to the fundamental end-to-end design principle of the TCP/IP for which the network cannot supply any explicit feedback, today TCP congestion control algorithm implements an additive increase multiplicative decrease (AIMD) algorithm. It is widely recognized that the AIMD mechanism is at the core of the stability of end-to-end congestion control. In this paper we describe a new mechanism. The key concept of the adaptive decrease mechanism is to adapt congestion window reductions to the bandwidth available at the time the congestion is experienced.

Keywords. TCP/IP, TCP behavior, Additive Increase Multiplicative Decrease.

1. Introduction

The stability of the backbone networks requires flow control mechanisms, so that the flows can adapt to the available resources. The mechanism for congestion control is added to TCP in 1980 after several consecutive collapses of the Internet [1].

Congestion control in the networks is performed through adaptive change of the transmission speeds of the sources as a function of the network state [2]. The congestion control has to be performed accordingly to a distribution law, which satisfies the following three main requirements:

- Maximum usage of the network resources and avoidance of network congestion.
- Guarantee for fair distribution of the available resources between the different flows.
- For higher efficiency an additional requirement for the congestion control mechanism can be added. This requirement involves fast redistribution of the resources in environments with varying number of sessions, which use a common environment.

The additive increase and multiplicative decrease mechanism is suitable for environments with distributed congestion control. The separate flows have to increase their throughputs accordingly to an additive law (for example linearly), when no congestion is present and decrease them by a multiplicative coefficient when congestion occurs. The sole requirement, for equilibrium (without congestion) of the system, is that all flows must increase or decrease their throughputs using the same slope. TCP is required to conduct such a method for congestion control [6]. Other protocols for congestion control, such as the ABR protocol in the ATM networks and others, use similar mechanisms [5], [7]. It is well known, from the definitions for control of the TCP/IP protocol stack, which present the network as a black box, that does not provide any additional information for the state of the network, that the congestion control in TCP is performed using the method of the sliding window.

2. The standard AIMD mechanism

Fig.1 presents a block scheme for the AIMD algorithm and an example of a distributed algorithm for an AIMD-based system of m flows. A new feature is that it distinguishes the amount by which the window of the flow has widened during additive increase phase. This amount is symbolized initially by ‘k’, and is increased additively each round with ‘a’ resources. Resources consumed by the flows (i.e. congestion window) are represented by the vector \((w_1, w_2, \ldots, w_m)\), where the i-th element of represents the congestion window of flow i.

Extending the above discussion, we highlight the following observations and we arrive at the conclusions that constitute the foundation of the modification of the standard AIMD algorithm:

- When the flows \(f_1\) and \(f_2\) are in additive increase phase they move parallel to the 45\(^{th}\) axis \((x_1=x_2)\) or fairness line. During that phase, equal amount of system resources \((k)\) is being allocated to the flows;
The initial windows \((x)\) and the amount of system resources \((k)\) that has been fairly allocated during additive increase are affected by multiplicative decrease. The original AIMD scheme calls for adjustments of the current windows - not the initial windows;

- The distance between the cliff line and the efficiency line when the system is in equilibrium depends only on the multiplicative decrease factor. The closer the efficiency line to the cliff, the higher the bandwidth utilization of the algorithm.

**Evaluation of the performance of TCP with the AIMD algorithm**

TCP with AIMD has been studied mainly in terms of throughput or goodput [2], [5], [7]. The approach is to isolate a flow from a system in equilibrium, record its highest window value and calculate the average number of packets/bits that this flow sends per round trip time (RTT), based on the behavior of AIMD.

If the transmitting time of all segments and all acknowledgments in a given window is minimal in comparison to the RTT, then we can give the following definition for round.

**Definition:** The round is period of time between the transmitting of the first segment of the current window and the reception of its acknowledgment. A suggestion is made, that in a given round the loss of one segment leads to the loss of the consecutive segments (correlated losses) as it is with the high-speed IP networks.

If \(k\) segments are transmitted before the congestion occurs in the round with losses, then these segments will generate acknowledgments, which will lead to the increase of the window. This means that \(k\) new segments will be send in the next round, known as the residual round.

Considering a single-flow AIMD system with RTT \(r\) and capacity \(C\) packets per RTT, the average throughput of this flow is:

\[
G = \frac{M}{r} \frac{3}{4} C
\]

where \(M\) is maximum segment size.

In terms of a periodic packet loss rate \(p\), the throughput is equal to:

\[
Thp = \frac{M}{r} \frac{C_i}{p} \sqrt{\frac{3}{2}}
\]

where \(C_i = \sqrt{\frac{3}{2}}\).

Fig. 2 shows the window evolution of a one-flow AIMD system. The efficiency of this system is the average throughput over the theoretical throughput. Therefore, the efficiency of TCP and in general of an AIMD system is 75%.

**Figure 2. AIMD window evolution under periodic loss**

The authors of [7] prove that in a synchronous system with \(m\) flows that run the AIMD algorithm, the fairness function \(F(x)\) is nondecreasing. This means that in a synchronous system the AIMD achieves max-min or optimal fairness when \(\lim_{t \to \infty} F(x) = 1\).

In an asynchronous system AIMD does not achieve max-min fairness. In a system with \(m\) flows, where \(x_i\) is the throughput of the flow \(i\) and \(r_i\) is its corresponding RTT, the rate/throughput distribution of the flows is given by:

\[
F = \sum_{i} \frac{1}{r_i} \log \frac{x_i}{a_i + bD x_i},
\]

where \(a_i\) and \(bD\) represent the additive increase and multiplicative decrease parameters, respectively.
According to equation (3) and (2) it is possible to calculate the throughput of each flow (Fig.3) when the system has reached its fair state.

![Figure 3. Dependences between the throughput of the system and the RTT r](image)

3. Modification of the AIMD mechanism – MAIMD1

Assume that two flows \((f_1, f_2)\) at time \(t\) enter the system with windows \(x_1\) and \(x_2\) \((x_1 < x_2, x_1 + x_2 < W)\). The flows start consuming resources additively from the system and at time \(t+\delta t\), the system notifies the flows to release resources \((x_1 + x_2 + 2k > W)\). From time \(t\) to time \(t+\delta t\) they consume exactly \(k\) resource units, each. When the system resources are exhausted the flows essentially release resources from the initial windows \(x_1\) and \(x_2\) which were allocated. Those released resources can be called the decrease window. This conclusion is the foundation for an approach for modification of the standard AIMD algorithm (MAIMD1) [3].

Decrease window is that portion of the congestion window that is multiplied by the decrease coefficient \(b_D\). In MAIMD1, \(w\) is the decrease window and \((w+k)\) is the congestion window. In AIMD \((w+k)\) is both the decrease window and the congestion window.

If \(x_1, x_2 \in \mathbb{N}\) denote the initial states of the windows of two flows, then the number of cycles \(n \in \mathbb{N}, n > 0\) completed towards convergence to fairness for both AIMD and MAIMD1 systems is the same.

The pseudo code of the MAIMD1 is given by:

```plaintext
MAIMD1 algorithm:
P2(w,k,dw){
  while (feedback==1)do
    { k:=k+a;
    }
  dw:=1/2w+k;
}
```

```
AIMD1 system:
System \{(x_1,x_2,...,x_m),m,n\}
i:=1
if (i<m) {
  w[i]:=x[i];
  i:=i+1;
}
if (i<n) {
  if (i<m) {w[i]:=P2(w[i]);
    i:=i+1;
  }
  j:=j+1
  return (w_1,w_2,...,w_m)
}
```

**Evaluation of the performance of MAIMD1**

In Fig. 4 we present the window evolution of TCP with MAIMD1 algorithm. On the base of this graphical representation we calculate its average throughput (per step or RTT).

![Figure 4. MAIMD1 window evolution under periodic loss](image)

Assume a single-flow system where \(W\) is the system’s capacity in packets. Let \(w=yW\) be the decrease window for this flow. After the multiplicative decrease, \((y/2)W\) resources will be released and they will be allocated again during additive increase \((k=(y/2)W)\).

So the size of the window is equal to: \(W=w+k=yW+(y/2)W\), which implies that \(y=2/3\). Hence, the decrease window value will be \(2/3W\) and the additive increase space will be \(1/3W\).

Using the same method as in the analysis of AIMD in section 2, but based on the window evolution of Fig. 4, the average throughput (per RTT) of the MAIMD1 flow is:

\[
Thp = \frac{M}{r} \frac{5}{6} W .
\]  

(4)

In terms of a periodic packet loss probability \(p\) the average throughput of this flow is (Fig.5):

\[
Thp = \frac{M}{r} \frac{C_2}{\sqrt{p}} ,
\]

(5)

where \(C_2 = \sqrt{5/2} \).
4. Further improvement of the modified algorithm

MAIMD1 increases the boundaries of variation of the congestion window \([2/3]W, W\) and increases the efficiency of the TCP flows by 1/3. However, this is not the maximum usable capacity of the AIMD algorithm.

Placing side by side the multiplicative decrease functions of AIMD and MAIMD1 we notice that MAIMD1 augments its window by a well-known factor 1/2:

\[
\begin{align*}
\text{AIMD} & : w \leftarrow 1/2(w + k), \\
\text{MAIMD1} & : w \leftarrow 1/2(w) + k.
\end{align*}
\]

This improves its fairness and efficiency and suggests that augmenting the windows after multiplicative decrease, by a well-known increase factor, leads to enhanced efficiency and faster convergence to fairness. Therefore, by augmenting the window value of the algorithm by a well known factor, we get a whole category of algorithms that have high efficiency and converge to fairness in exactly the same number of cycles.

A natural question of practical importance is how far we can adjust the window upwards. We need to determine the appropriate value, which will not violate the constraints of congestion avoidance nor the conditions of equilibrium.

Practical requirement is to avoid an increase which will cause immediate congestion.

From equation (7) we can recall that the window decrease can be given by

\[
w \leftarrow w / 2 + k.
\]

Hence, a hard boundary for the increase extra value is half the maximum decrease window.

The primary task is to determine the amount of resources that could be added to the congestion windows after multiplicative decrease. According to the observation above, this needs to be less than half the maximum decrease window in the system. Otherwise equilibrium will not be reached.

Let \(w_j\) be the decrease window of an MAIMD1 flow at the beginning of cycle \(j\), and assume that after \(k_j\) steps congestion occurs in the network. In the next cycle: \(w_{j+1} \leftarrow w_j / 2 + k_j\). In equilibrium \(k_j = w_j / 2\) and \(k_j\) is a common knowledge in the system. Adding resources from \(k_j\) into \(w_{j+1}\) does not preserve the congestion avoidance property of the algorithm. If the system is in equilibrium and any flow leave the system, the system will still be in equilibrium but \(k_j > w_j / 2\). If we add resources from \(k_j\) to \(w_{j+1}\) the sum of the windows in the system might exceed the system threshold \((\sum w_j + k_j) \geq X\).

The following equations are true for the system:

\[
\begin{align*}
\text{AIMD} & : w_j \leftarrow w_{j-1} / 2 + k_{j-1}, \\
\text{MAIMD2} & : w_{j+1} \leftarrow w_j - 1/2(w_{j-1}) + k_{j-1} / 2,
\end{align*}
\]

Evaluation of the performance of MAIMD2

The minimum value of the congestion window of MAIMD2 is \(dw + k_{j-1} / 2\). By estimating \(dw\) and \(k_{j-1}\) we compute the average throughput per RTT of the algorithm.

If we assume a single-flow system: after \(k\) additive increase steps the congestion window \((w+k)\) will reach the maximum value \(W_{max}\). From equation (10) we can determine that:

\[
w + k - dw = k + k_{j-1} / 2,
\]
where \( dw \) is an array, in which we record the values of \( w \), and the record for the current moment is \( 2/3W \).

From Fig.4 it is evident that the MAIMD1 algorithm has a decrease window of \( 2/3W \) and increase space of \( 1/3W \). The decrease window has the same value as in MAIMD2 (eq.(11)).

Under the same conditions, let \( q_j \) be the number of additive increase steps of the last cycle and the increase space can be determined by the following equation:

\[
\phi = W - dw = 1/3W , \quad (12)
\]

The real additive increase during the \( j+1 \) cycle would be \( k_{j+1} = \phi - k_{j-1} / 2 \), and hence we receive that \( k_{j-1} / 2 = -1/3\phi = -1/9W \). This means that

\[
dw + k_{j-1} / 2 = 7/9W .
\]

If we use the analysis from section 3, but with the window adjustments form equations 11 and 12, the mean throughput of a TCP flow, that implements the MAIMD2 algorithm can be given using the equation (Fig.6):

\[
Thp = M \frac{8}{9} W , \quad (13)
\]

5. A proposition for combined adaptive approach for congestion control

Based on the analysis of the standard AIMD algorithm and the two modified algorithms, we propose a combined algorithm for adaptation of the system for more efficient usage of the available resources. The adaptive algorithm is applied to a modified version of the TCP protocol - ModTCP, which has integrated module for bandwidth estimation [4]. This bandwidth estimation module uses modified and improved version of a low-pass filter for cancellation of the phenomenon called acknowledgment compression. The proposed algorithm guarantees the passing of the system trough three possible states. The block scheme of the proposed algorithm is presented in Fig. 7.

![Figure 7. Block scheme of the combined adaptive algorithm](image)

The passing of the system from one state to another is determined by the variation of the value for the available bandwidth \( b \) received by the bandwidth estimation module for the current moment \( t \) and the next moment \( t+1 \).

From the defined conditions and the structure of the algorithm it is easy to see that at a given moment of time the system can be in only one of the given states. Hence the proposed combined algorithm does not complicate the realization of the processes for control of the size of the window, because each step is conducted in separated algorithm with relatively small complexity.

6. Simulation and results

In order to evaluate the combined adaptive scheme we use the simulation scenario shown on Fig. 8.

![Figure 8. Simulation scheme of the evaluation scenario](image)
The simulation scheme consists of bottleneck duplex line with capacity of 10 Mbit/s. The line is shared by TCP sources, which use different versions of the protocol, and a UDP ON/OFF source, which transmits with a constant rate of 1 Mbit/s in the ON periods, which last for 100 s. The reverse channel is used by N TCP sinks, which cause congestion and acknowledgment compression. The size of all segments is 1400 bytes, and the acknowledgments size is 40 bytes. The RTT length is set to 200 ms.

8. Acknowledgements

This work is a part of the projects: BG051PO001/07/3.3-02/8-“MEQSIS”, funded by the scheme “Support of the development of PhD students, postdoctoral, post-graduate and young scientists” from the program “Development of human resources” of the “European social fund” and BY-TH-105/2005 of Bulgarian Science Fund at Ministry of Education and Science.

9. References