PSO-Vegas: PSO-based enhanced Vegas

Abstract. Biology is hosting many self-organizing processes. These processes can be studied by researchers to employ their principles as an inspirational metaphor to offer new solutions for different scientific problems. We follow such an inspiration here, to improve TCP Vegas algorithm. It has been confirmed that TCP Vegas has higher performance in compare with TCP Reno. However, TCP Vegas has several problems that affect its performance in congestion avoidance phase. Fixed values for $\alpha$ and $\beta$ are one of the most important weaknesses of TCP Vegas. Ideally, $\alpha$ and $\beta$ should be function of network conditions. For this purpose, this paper presents a PSO-based modified Vegas algorithm, which adjusts its parameters $\alpha$ and $\beta$ to present good performance compared to Vegas. The simulation results show that the performance of the proposed algorithm is much better than TCP Vegas.

Keywords: Communication Network, Congestion Control, Vegas and PSO

Stresszczenie. W biologii występuje wiele samoorganizujących się procesów. Takie procesy mogą być studiowane jako metoda rozwiązywania wielu różnych problemów naukowych.artykuł przedstawia ulepszanie algorytmu TCP Vegas. Zostało potwierdzone, że TCP Vegas ma lepsze parametry niż TCP Reno. Jednak samo TCP Vegas ma też słabości – jedną z nich jest stała wartość $\alpha$ i $\beta$. W warunkach idealnych $\alpha$ i $\beta$ powinny być funkcją warunków sieci. W tym celu w pracy przedstawia się zmodyfikowany algorytm Vegas z możliwością ustawiania parametrów $\alpha$ i $\beta$. Symulacje wykazały, że nowy algorytm ma lepsze właściwości niż TCP Vegas. (PSO-Vegas: algorytm Vegas z poprawioną PSO)

Keywords: Komunikacja sieciowa, Kontrola przepływu, Vegas i PSO

Stwórzono. W biologii występują wiele samoorganizujących się procesów. Takie procesy mogą być studiowane jako metoda rozwiązywania wielu różnych problemów naukowych.artykuł przedstawia ulepszanie algorytmu TCP Vegas. Zostało potwierdzone, że TCP Vegas ma lepsze parametry niż TCP Reno. Jednak samo TCP Vegas ma też słabości – jedną z nich jest stała wartość $\alpha$ i $\beta$. W warunkach idealnych $\alpha$ i $\beta$ powinny być funkcją warunków sieci. W tym celu w pracy przedstawia się zmodyfikowany algorytm Vegas z możliwością ustawiania parametrów $\alpha$ i $\beta$. Symulacje wykazały, że nowy algorytm ma lepsze właściwości niż TCP Vegas. (PSO-Vegas: algorytm Vegas z poprawioną PSO)
The Vegas algorithm tries to adjust number of queued packets between important roles in the system performance. The Vegas algorithm is aimed to keep the RTT in its lowest level, while it considers constrains such as high utilization in bottleneck link. If the network is modelled as an M/M/1 system from the source point of view, the relation of link utilization and number of queued packets can be described as equation (7), in which $U$ is the bottleneck link utilization [19].

$$\text{(7) Number of Queued Packets} = \frac{U}{1-U}$$

For example if we consider 0.5 as the lowest level of acceptable bottleneck utilization, then according to (7), there should be at least one queued packet in the bottleneck router. Hence, considering all the above factors, the objective function is defined as follows:

$$\text{Minimize RTT}$$

$$\text{Subject to : Bottleneck Utilization > 0.5}$$

$$\Rightarrow \text{Number of Queued Packets > 1}$$

Note that number of queued packets is estimated by using BaseRTT and measured RTTs. As we know $\Delta$ that is computed by Vegas is an estimation of number of queued packets.

By using this objective function, we can employ PSO technique to design a novel source algorithm, in which, parameters are set by considering the link utilization, the queue length and connections RTT. For this purpose, $\alpha$ and $\beta$ will be calculated according to the following equations:

$$V_{a}(k+1) = wV_{a}(k)$$

$$+ c_{1}\cdot r_{1} \cdot (\text{Lbest}_{a}(k) - X_{i}(k))$$

$$+ c_{2}\cdot r_{2} \cdot (\text{Gbest}_{a}(k) - X_{i}(k))$$

$$\text{(10) }$$

$$V_{b}(k+1) = wV_{b}(k)$$

$$+ c_{1}\cdot r_{1} \cdot (\text{Gbest}_{b} - \beta(k))$$

$$+ c_{2}\cdot r_{2} \cdot (\text{Lbest}_{b} - \beta(k))$$

$$\text{(11) }$$

$$\beta(k + 1) = \beta(k) + V_{b}(k+1)$$


These best positions are selected according to objective function of (8). The parameters $c_{1}$ and $c_{2}$ determine the relative pull of Lbest and Gbest and the parameters $r_{1}$ and $r_{2}$ lead to stochastically varying these pulls. These parameters should be selected sensitively for efficient performance of PSO-Vegas. The constants $c_{1}$ and $c_{2}$ represent the weighting of the stochastic acceleration terms that pull each particle toward Lbest and Gbest positions. Low values allow particles to roam far from the target regions before being tugged back. On the other hand, high values result in abrupt movement toward, or past, target regions. Hence, the acceleration constants are set to be 0.5. Suitable selection of inertia weight, $w$, provides a balance between global and local explorations, thus requiring less iteration on average to find a sufficiently optimal solution. We set $w$ to be 0.6. Fig. 1 shows how PSO-Vegas tunes $\alpha$ and $\beta$ to improve Vegas algorithm.
Packet-level simulation

To implement the proposed model, we use the packet-level simulator ns-2 [20], and modify the TCP Vegas module to implement PSO-Vegas algorithm. We present a group of simulation results to show the validity of our analysis. We demonstrate through extensive simulations that PSO-Vegas outperform Vegas both in typical and high bandwidth-delay environments. For this purpose, we apply it to network of Fig. 2 and consider a ten-connection network that has a single bottleneck link. All routers use RED algorithm [21] to manage their queues. We assume that all flows are long-lived, have the same end-to-end propagation delay and always are active. The bottleneck bandwidth, RTT, and the number of flows vary based on the objective of the experiment. The buffer size is set to the delay-bandwidth product. The data packet size is assumed 1000 bytes.

Scenario 1: Typical Setup

In this scenario we consider a typical network in which bottleneck capacity is 16 Mbps and is shared by 10 sources. All flows have same RTT of 55 ms and are active from \( t=0 \) till \( t=200 \) second. The simulation results of this congestion control system are shown in Fig. 3. In order to reference to the results of these figures, we note that:

**Utilization:** According to the Fig. 3 after the start-up transient, utilization of bottleneck link for PSO-Vegas remains always around 90% that is good enough. As has been reported in table 1, average utilization of PSO-Vegas is higher than Vegas about 18%.

**Stability and speed of convergence:** As we can see in Fig. 3 utilization of PSO-Vegas has decreasing oscillation level and track stable behaviour in compare with Vegas. Note that convergence is an important feature for any congestion control scheme.

| Table 1. Comparison of PSO-Vegas and Vegas in a typical network |
|-------------------|------------------|------------|
| Source            | Dropped Packets Count | Utilization |
| TCP Vegas         | 7038              | 66.26      |
| PSO-Vegas         | 203               | 84.27      |
Scenario 2: Sudden changes in traffic demand

In order to address the adaptability of the proposed algorithm when sudden changes in traffic demands take place, we consider another scenario on Fig. 2. We simulate this network with the single bottleneck link capacity of 30 Mbps, shared by 25 sources. Only 10 sources are active at time 0, thereafter 15 more sources become active with random intervals, until all 25 sources are active. All sources have the same end-to-end propagation delay of 42 ms. Fig. 4 shows the dynamics of our protocol during 200 seconds simulation time.

Fig. 4. PSO-Vegas adapts well to sudden changes in the network conditions

In reference to the results of Fig. 4, we note that while Vegas’ utilization oscillates seriously in response to changes in number of flows, PSO-Vegas keeps bottleneck utilization stably near the optimum value. Also, as can be found in table 2, average behaviour of PSO-Vegas is remarkably better than Vegas. Its drop rate is only 7% of Vegas’ and utilizes bottleneck link around 10% more than Vegas.

Table 2. Comparison of PSO-Vegas and Vegas in a network with varying traffic

<table>
<thead>
<tr>
<th></th>
<th>Dropped Packets Count</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP Vegas</td>
<td>8611</td>
<td>85.29</td>
</tr>
<tr>
<td>PSO-Vegas</td>
<td>530</td>
<td>96.93</td>
</tr>
</tbody>
</table>

Scenario 3: High speed Network

As we know, high-speed networks with high bandwidth-delay (HBD) product present a unique environment where currently TCP may have a major challenge to its performance. In this scenario we consider a high bandwidth-delay product network, in which, bottleneck capacity is 100 Mbps and flows’ RTT is 40 ms. Fig.5 and table 3 show the simulation results for this scenario.

Fig. 5. PSO-Vegas keeps higher utilization and drops fewer packets in HBD environments

Simulation results demonstrate that as capacity increases, bottleneck utilization decreases significantly for both Vegas and PSO-Vegas, but PSO-Vegas presents better performance in compare with TCP Vegas in terms of bottleneck utilization and number of dropped packets. This feature makes PSO-Vegas suitable for HBD environments.

Table 3. Comparison of PSO-Vegas and Vegas in a HBD network

<table>
<thead>
<tr>
<th></th>
<th>Dropped Packets Count</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP Vegas</td>
<td>822</td>
<td>70</td>
</tr>
<tr>
<td>PSO-Vegas</td>
<td>93</td>
<td>75.64</td>
</tr>
</tbody>
</table>

Conclusion

In this paper we proposed PSO-Vegas algorithm as a bio-inspired congestion control algorithm. Toward this
design, the following steps were considered: (1) formulating the congestion control problem as an optimization problem that tries to direct the network to its optimum point. (2) Choosing PSO technique as solver of the optimization problem. The main feature of this algorithm is that it sets $\alpha$ and $\beta$ parameters based on the network dynamic conditions by employing PSO technique. (3) Implementing of the model in ns-2 environment. The simulation results indicate that number of dropped packets and bottleneck utilization of PSO-Vegas are much more better than Vegas algorithm.

REFERENCES


Authors
Shahram Jamali, University of Mohaghegh Ardabili, Ardabil, Iran, E-mail: jamali@iust.ac.ir; Akbar Effekhary, Islamic Azad University-Sarab Branch, Sarab, Iran, E-mail: Akbar_effekhary@yahoo.com.