Shahram JAMALI[1], Akbar EFTEKHARI[2]

University of Mohaghegh Ardabili, Ardabil, Iran (1), Islamic Azad University-Sarab Branch, Sarab, Iran (2)

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 α and β are one of the most important weaknesses of TCP Vegas. Ideally, α and β should be function of network conditions. For this purpose, this paper presents a PSO-based modified Vegas algorithm, which adjusts its parameters i.e. α and β to present good performance compared to Vegas. The simulation results show that the performance of the proposed algorithm is much better than TCP Vegas.*

***Streszczenie.** W biologii występuje wiele samoorganizujących się procesów. Takie procesy mogą być studiowane jako metoda rozwiązywania wielu różnych problemów naukowych. Artykuł przedstawia ulepszenie 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 α i β. W warunkach idealnych α i β powinny być funkcją warunków sieci. W tym celu w pracy przedstawia się zmodyfikowany algorytm Vegas z możliwością ustawiania parametrów α i β. Symulacje wykazały, że nowy algorytm ma lepsze właściwości niż TCP Vegas. (**PSO-Vegas: algorytm Vegas z poprawioną PSO**)*

Keywords: Communication Network, Congestion Control, Vegas and PSO
Słowa kluczowe: sieci komunikacyjne, kontrola zatorów, Vegas

Introduction

Nature-inspired computation relies on examining, analyzing and sometimes modelling of the fundamental principles of natural computation, and how these principles can be adopted and applied to solve problems of computer science and other engineering fields. Nature-inspired computation is widely used in many fields, such as optimal design, optimal control and creative design. Recently, nature-inspired methods and algorithms are widely used in computer networking research area. For example network security, pervasive computing, sensor networks [1] and Internet congestion control [2, 3] are fields that have experienced biologically-inspired solutions. In this paper we are going to employ one of the well-known bio-inspired techniques, namely, Particle Swarm Optimization (PSO) [4, 5, 6] to improve TCP Vegas algorithm [7]. Wide range of research has been carried out on the TCP congestion avoidance mechanism in order to enhance the performance of communication networks. The Vegas algorithm, presented by Brakmo et al. as an alternative to TCP Reno [8], has higher performance in compare with other congestion control algorithms [9, 10, 11]. By the way, Vegas seems to be suitable algorithm for emerging network environments such as MANET [12, 13] and high bandwidth-delay product [14].The Vegas algorithm tries to adjust number of queued packets in an acceptable level i.e. between α and β. In this algorithm α and β are set to 1 and 3, respectively. Unfortunately, these static values in the Vegas algorithm cannot lead to desired performance level. We propose a modified Vegas, called PSO-Vegas, in which α and β are tuned dynamically with respect to the network conditions. PSO-Vegas is a source algorithm that estimates network conditions by using measured RTTs and then uses these values to direct the network to a high-performance state by appropriately adjustment of α and β.

TCP Vegas

TCP-Vegas is a delay-based transport protocol, which adjusts its congestion window size according to measured RTTs and the gap $\triangle$ between the expected and estimated sending rates. Three thresholds i.e. α, β and γ are defined in Vegas algorithm. Senders of Vegas compare $\triangle$ with γ in slow start phase and with α and β in congestion avoidance phase to determine window adjustments. The estimation of the gap is done once per RTT period. Vegas updates $BaseRTT$ by the minimum of all measured round trip times and computes the expected rate as $Expected = w/BaseRTT$, where w denotes window size. If RTT_a denotes the average measured RTT, Vegas calculates the actual rate as $Actual = w/RTT_a$. Then the gap between expected and estimated sending rates is computed as $\triangle = (Expected-Actual)*BaseRTT$.

In slow start phase, the congestion window is smaller than slow start threshold, called W_{th}. When receiving a new ACK, if $\triangle$ is less than γ, the sender increases w by one. If not, Vegas decreases the window size by a specific fraction p, sets W_{th} to be the reset value W_{th}, and then switches to the congestion avoidance phase. Slow start is initiated at the beginning or after a timeout event and ends when the window is larger than W_{th}. Vegas applies timeout mechanism by a coarse grain timer, which is checked once per 500 ms. Congestion window update in slow start phase can be explained as in (1).

$$(1)\qquad w = \begin{cases} w + 1 & if \;\; \Delta \prec \gamma \\ w \times (1 - p) & if \;\; \Delta \ge \gamma \end{cases}$$

When sender side of Vegas is in congestion avoidance phase and receives a new ACK, it increases the window by $1/w$, if $\triangle$ is less than α and decrements it by $1/w$ if $\triangle$ is larger than β, and keeps it unchanged when $\triangle$ falls between α and β. Details are shown in (2).

$$(2)\qquad w = \begin{cases} w + \dfrac{1}{w} & if \;\; \Delta \le \alpha \\ w & if \;\; \alpha \prec \Delta \prec \beta \\ w - \dfrac{1}{w} & if \;\; \Delta \ge \beta \end{cases}$$

In addition to coarse grain timeout mechanism, Vegas performs retransmission with another fine grain timer via time stamp included in packets, which helps Vegas to retransmit faster than other TCP variations with only coarse grain timer. Also, some works propose a special AQM to support Vegas source algorithm [15]. More details about Vegas can be found in [7, 9].

Particle Swarm optimization algorithm

Particle swarm optimization algorithm, which is adapted for optimizing difficult functions, is capable of imitating the ability of human societies to process knowledge [4, 5, 7]. It has roots in artificial life (such as fish schooling, bird flocking and swarming) and evolutionary computation. Its key concept is that potential solutions are accelerated toward better or more optimum solutions. Its model can be implemented as a simple computer procedure and is computationally inexpensive in terms of both memory requirements and time complexity. It lies somewhere in between evolutionary computing and the genetic algorithms. As in evolutionary computation paradigms, the concept of fitness is used and candidate solutions to the problem are termed particles or sometimes individuals, each of which adjusts its flying based on the flying experiences of both itself and its companion.

In PSO algorithm, a swarm consists of m particles, in which, each particle is treated as a point in a N-dimensional space which adjusts its flying according to its own flying experience as well as the flying experience of other particles. The position and velocity of particle i at iteration k can be respectively expressed as

$$(3)\qquad X_i(k) = [X_{i1}(k), X_{i2}(k), ..., X_{iN}(k)]$$

$$(4)\qquad V_i(k) = [V_{i1}(k), V_{i2}(k), ..., V_{iN}(k)]$$

Particle i tries to track the best solution that has achieved so far by that particle. This value is called local best $Lbest_i$. Another best value that is followed by the PSO is the best value obtained so far by any particle in the neighbourhood of that particle. This value is called global best $Gbest$. The fundamental concept of PSO is placed in accelerating each particle toward its local best and the global best locations. The fix of responses between the individual and group values leads to a diversity of response. As it is reported in [16], this optimization technique can be used to solve many of the same kinds of problems as GA, and does not suffer from some of GA's difficulties. It has also been shown to be robust in solving problem featuring non-linearity, non-differentiability and high-dimensionality. Because of such abilities PSO has been used in many engineering research fields [17, 18]. The velocity and position of particle i at iteration $k+1$ can be calculated according the following equations:

$$(5)\qquad \begin{aligned} V_i(k+1) = {} & wV_i(k) \\ & + c_1 r_1 \big(Lbest_i(k) - X_i(k)\big) \\ & + c_2 r_2 \big(Gbest(k) - X_i(k)\big) \end{aligned}$$

$$(6)\qquad X_i(k+1) = X_i(k) + V_i(k+1)$$

where: w – inertia weight, c_1, c_2 – constants which determine the influence of the local best position $Lbest_i(k)$ and the global best position $Gbest(k)$, r_1 and r_2 – random numbers uniformly distributed within [0,1].

Proposed model: PSO-Vegas

As we saw, in the Vegas algorithm, α and β play important roles in the system performance. The Vegas algorithm tries to adjust number of queued packets between α and β. In original Vegas α and β have static values of 1 and 3 respectively. It can be found that these static values in the Vegas algorithm can't lead to desired level of performance. Therefore, we propose a modified Vegas algorithm, in which, α and β are tuned dynamically with respect to the network conditions such as RTT, bottleneck bandwidth, etc. In this way, we use PSO algorithm to find optimum points of α and β for any network conditions.

As the first step of our design process, an objective function should be defined to link the design requirements with the optimization algorithm. Any congestion control 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].

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

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:

$$(8)\qquad \begin{aligned} & Minimize\quad RTT \\ & \quad Subject\ to:\ Bottleneck\ Utilizatio n > 0.5 \\ & \qquad\qquad \Rightarrow Number\ of\ Queued\ Packets > 1 \end{aligned}$$

Note that number of queued packets is estimated by using $BaseRTT$ and measured RTTs. As we know $\triangle$ 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, α and β will be calculated according to the following equations:

$$(9)\qquad \begin{aligned} V_\alpha(k+1) = {} & wV_\alpha(k) \\ & + c_1 r_1 \big(Gbest_\alpha - \alpha(k)\big) \\ & + c_2 r_2 \big(Lbest_\alpha - \alpha(k)\big) \end{aligned}$$

$$(10)\qquad \alpha(k+1) = \alpha(k) + V_\alpha(k+1)$$

$$(11)\qquad \begin{aligned} V_\beta(k+1) = {} & wV_\beta(k) \\ & + c_1 r_1 \big(Gbest_\beta - \beta(k)\big) \\ & + c_2 r_2 \big(Lbest_\beta - \beta(k)\big) \end{aligned}$$

$$(12)\qquad \beta(k+1) = \beta(k) + V_\beta(k+1)$$

where: ($Lbest_\alpha$, $Lbest_\beta$) – local best position, ($Gbest_\alpha$, $Gbest_\beta$) – global best previous position.

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 α and β to improve Vegas algorithm.

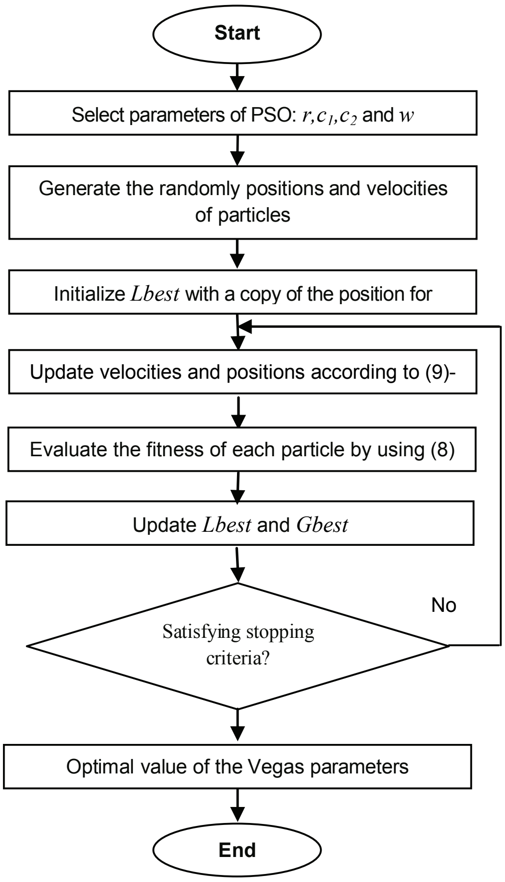
Start
Select parameters of PSO: r, c_1, c_2 and w
Generate the randomly positions and velocities of particles
Initialize $Lbest$ with a copy of the position for
Update velocities and positions according to (9)-
Evaluate the fitness of each particle by using (8)
Update $Lbest$ and $Gbest$
Satisfying stopping criteria?
No
Optimal value of the Vegas parameters
End

Fig. 1. Flowchart of PSO- Vegas algorithm in operation

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.

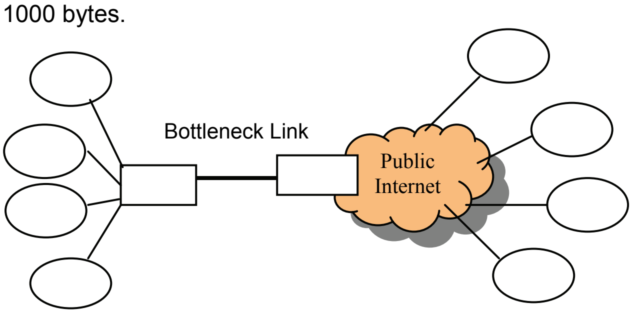

Fig. 2. Network Model

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.

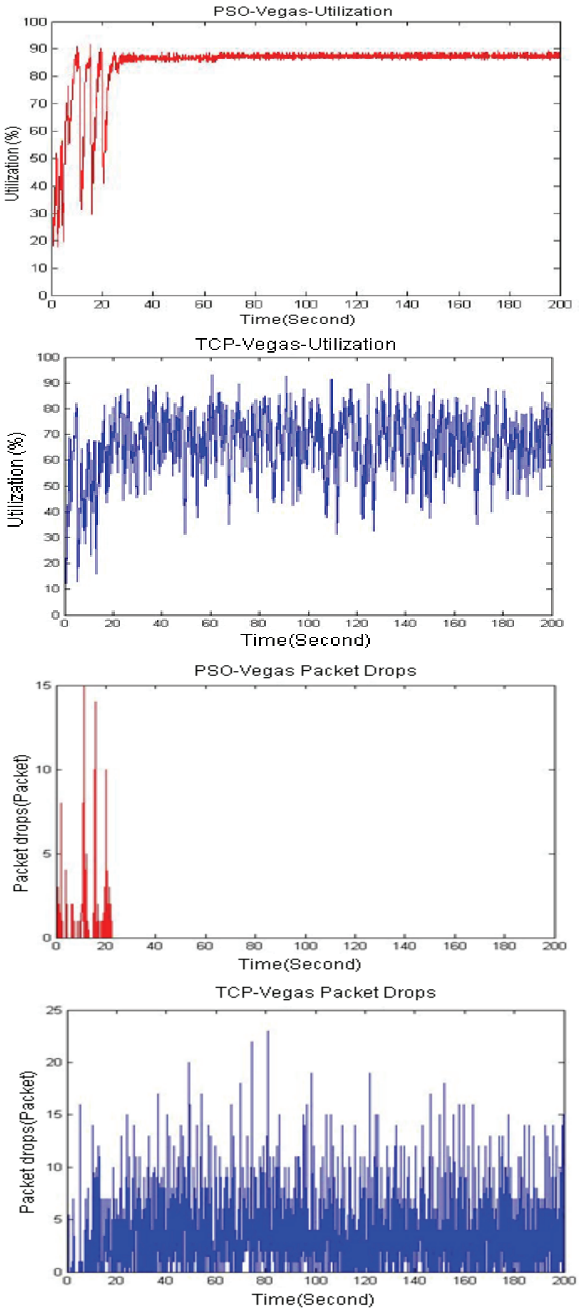

Fig.3. PSO-Vegas keeps higher utilization and drops fewer packets in compare with Vegas

Table 1. Comparison of PSO-Vegas and Vegas in a typical network

----------------------	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.

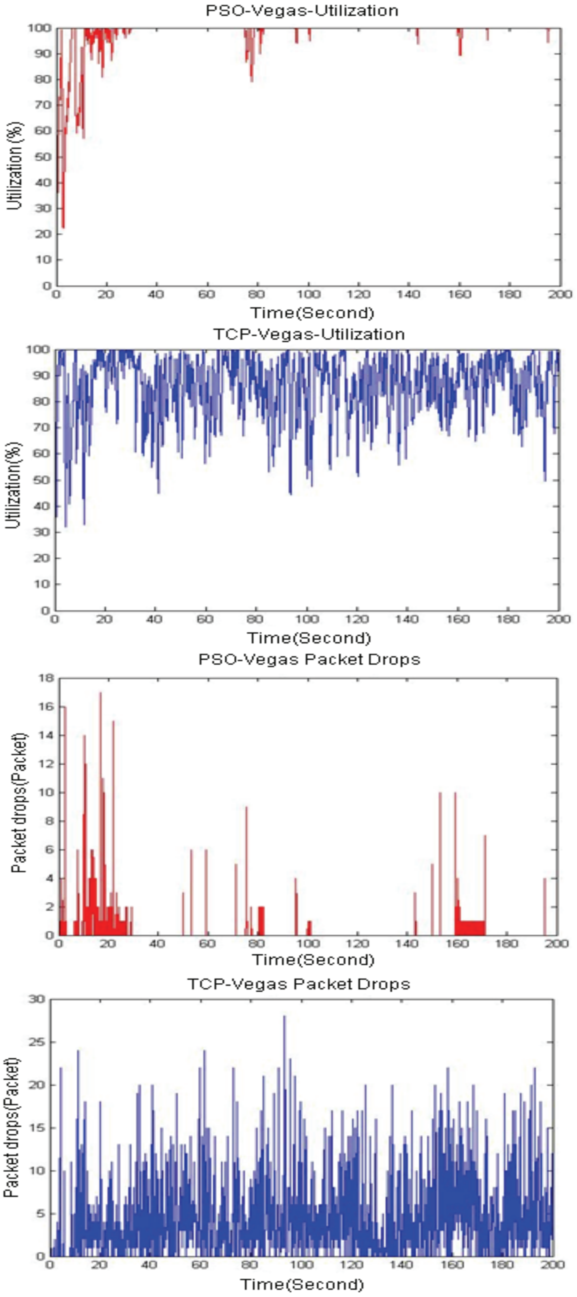

Fig. 4. PSO-Vegas adapts well to sudden changes in the network conditions

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

----------------------	Dropped Packets Count	Utilization
TCP Vegas	8611	85.29
PSO-Vegas	530	96.93

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.

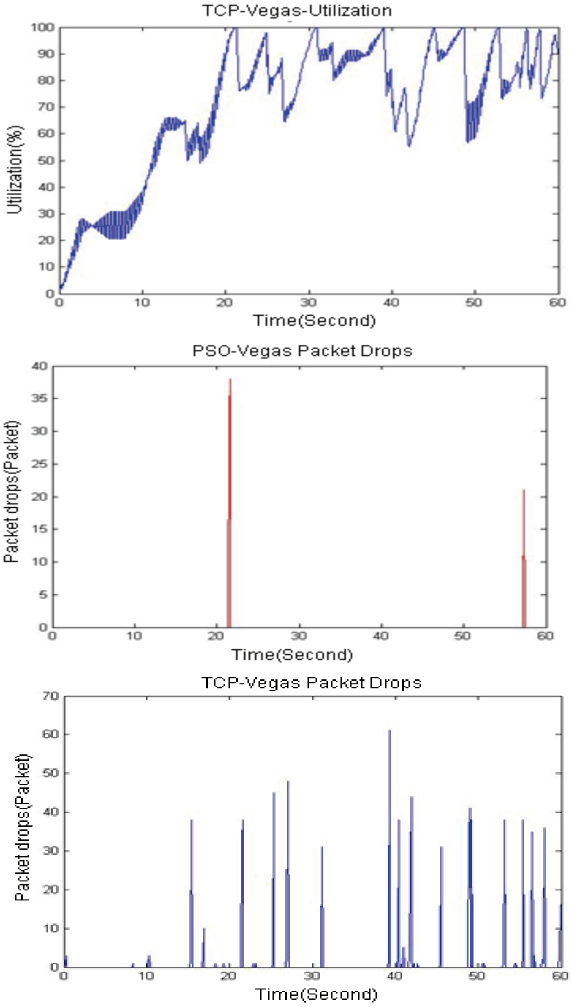

Fig. 5. PSO-Vegas keeps higher utilization and drops fewer packet in HBD environments

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 term 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

----------------------	Dropped Packets Count	Utilization
TCP Vegas	822	70
PSO-Vegas	93	75.64

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 α and β 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

[1] F. Dressler, Ö. B. Akan, A survey on bio-inspired networking, Computer Networks, 2010.

[2] M. Analoui, S. Jamali, Bio-Inspired Congestion Control: Conceptual Framework, Algorithm and Discussion, Advances in Biologically Inspired Information Systems: Models,Methods, and Tools, Studies in Computational Intelligence (SCI), Springer, 2007.

[3] M. Analoui, S. Jamali, Congestion control in the internet: inspiration from balanced food chains in the nature, Journal of Network and Systems Management, 2008.

[4] R. Eberhart, J. Kennedy, A New Optimizer Using Particles Swarm Theory, International Symposium on Micro Machine and Human Science , IEEE Service Center, 1995.

[5] J. Kennedy, R. Eberhart, Particle Swarm Optimization, IEEE International Conference on Neural Networks, 1995.

[6] Y. Shi, R. Eberhart, ,Parameter Selection in Particle Swarm Optimization, Conference on Evolutionary Programming, 1998.

[7] L.S. Brakmo, L.L. Peterson, TCP Vegas: end to end congestion avoidance on a global Internet, IEEE Journal of Selected Areas in Communication, 1995.

[8] V. Jacobson, Congestion avoidance and control, ACM SIGCOMM, 1988.

[9] T. Bonald, Comparison of TCP Reno and TCP Vegas via fluid approximation, Technical Report, 1998.

[10] J. Mo, R.J. La, V. Anantharam, J.C. Walrand, Analysis and comparison of TCP Reno and Vegas, INFOCOM, 1999.

[11] S.H. Low, L.L. Peterson, L. Wang, Understanding Vegas: a duality model, Journal of ACM, 2002.

[12] D. Kim, H. Bae, C.K. Toh, Improving TCP-Vegas performance over MANET routing protocols, IEEE Transactions on Vehicular Technology, 2007.

[13] L. Ding, X. Wang, Y. Xu, W. Zhang, Improve throughput of TCP-Vegas in multihop ad hoc networks, Computer Communications, 2008

[14] Y. Chan, C. Lin, Quick Vegas: Improving performance of TCP Vegas for high bandwidth-delay product networks, IEICE Transactions on Communications, 2008.

[15] N. Bigdeli, M. Haeri**,** AQM controller design for networks supporting TCP Vegas: A control theoretical approach, ISA Transactions, 2008.

[16] S. H. Zahiria, S. A. Seyedin, Swarm intelligence based classifiers, Journal of Franklin Institute, 2007.

[17] H.I. Ali, S.B. Mohd Noor, S.M., Pso-based robust H∞ controller design using cascade compensation network, Journal of IEICE Electronics Express, 2010.

[18] R.N. Ray, D. Chatterjee, S.W.Goswami, A PSO-based optimal switching technique for voltage harmonic reduction of multilevel inverter, Journal of Expert Systems with Applications, 2010.

[19] D. A. Menasce, A. F. Almeida, Capacity Planning for Web Services, Metric, Models, and Methods, Prentice-Hall, 2002.

[20] Ns-2. Network Simulator, http://www.isi.edu/nsnam/ns.

[21] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking, 1993.

Authors

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