Journal of Computer Science 8 (5): 665-672, 2012
ISSN 1549-3636

A Forager Bee's Intelligence Inspired Dynamic Queue Scheduling for the Internet Traffic

[1]Suresh, Y., [2]S. Arumugam and [3]M.A. Bhagyaveni
[1]Department of Information Technology,
Sona College of Technology, Salem-636005, India
[2]Department of CEO, Nandha Engineering College, Erode, India
[3]Department of ECE, Anna University, Chennai, India

Abstract: Problem statement: In the present evolution of large scale internet communication, per flow control scheme faces scalability issue due to tremendous number of flows. The aggregation based approaches such as differentiated architecture relieve the storage of state of flows in core router. TCP is the dominating protocol that carries majority of the total internet traffic. Recent internet traffic measurement shows most of the TCP flows are short lived. The performance improvement in the internet traffic can be achieved by the advantages of scheduling algorithms to favor short TCP flows first However long TCP flows competing against short TCP flows starve at some point. **Approach:** In this study we propose aggregation based scheduling algorithm namely Guaranteed Dynamic Queue Scheduling (G-DQS) that estimates the available bandwidth of the network using the forager bee's intelligence for providing guaranteed throughput. In addition, G-DQS algorithm is proposed to favor the short TCP flows without penalizing the performance of long flows using dynamic scheduling ratio. **Results:** Simulation of the proposed scheduling method show that mean transmission time of flows and packet loss significantly decrease in comparison with FIFO and RuN2C. **Conclusion:** Proposed forager bee's intelligence inspired scheduling approach achieves the guaranteed throughput in the large scale network.

Key words: TCP flows, aggregated scheduling, large scale networks, Internet Guaranteed Dynamic Queue Scheduling (G-DQS), guaranteed throughput

INTRODUCTION

An explosive growth in business applications using the Internet have resulted in a strong demand for some notion of reliability or quality of service. During periods of congestion or failure, the quality of service of all flows is degraded. As a result, a strong need for service differentiation in the flows to provide guarantees and/or assuring minimum throughput guarantees.

To provide quality of service QoS, the Internet Engineering Task Force (IETF) has proposed the Integrated Services (Int-Serv) model (Braden *et al*., 1994) and the Differentiated Services (Diff-Serv) model (Bernet *et al*., 1999). The Int-Serv model provides per-flow QOS guarantees, but does not scale well with the number of users. The Diff-Serv model on the other hand provides per hop behavior based on aggregates (or classes) at the core routers and hence scales well at the core routers.

The internet carries different types of traffic with the increased use of peer to peer, Web, Telnet, VoIP, FTP applications. These traffics are differentiated as short flows and long flows. Short flows are mainly generated by the delay sensitive applications such as Web, Telnet and VoIP. The long flows are generated in the internet originate from peer to peer applications (Rai *et al*., 2005).

A flow is defined as a group of packets with a common set of attributes such as source address, destination address, source port, destination port. The existing internet uses TCP as a Transport Control protocol and FIFO scheduling in routers. TCP is connection oriented transport layer protocol that provides end to end delivery across the internet (Floyd, 2001). The studies (Guo and Matta, 2002) shows that TCP conveys about 80-90% of traffic over the internet. Internet traffic exhibits that most of the TCP flows are short, while more than 50% of the flows are carried by less than 5% of largest flows (Paxson and Floyd, 1995).

Offering service guarantees to existing and emerging applications in the Internet has been a big challenge to Internet designers. One of the most important mechanisms to provide service guarantees

Corresponding Author: Suresh, Y., Department of Information Technology, Sona College of Technology, Salem-636005, India

(Zhang, 1995) is scheduling. Scheduling determines the order in which the packets from different flows are served. *Packet* scheduling in routers has been an active area of research in the last two decades and most of the attention has focused on Processor Sharing (PS) type of scheduling algorithms.

From queuing theory point of you it has been shown that choosing an appropriate scheduling algorithm significantly improves the performance of the system. The studies (Chen and Heidemann, 2003) shows that short flows should be given highest priority over the long flows. The issues in the design of a scheduling algorithm are (a) Classification of short flows and long flows (Avrachenkov *et al*., 2004) Favoring short flows without penalizing the performance of long flows (c) available bandwidth estimation for achieving the guaranteed throughput.

Aiming these issues we propose a Forager bee's intelligence Guaranteed Dynamic Queue Scheduling algorithm (G-DQS) which classifies internet flows into a short flows and long flows. The G-DQS estimates the available end to end bandwidth across the link by adapting the forager bee's intelligence into the monitor component of architecture and uses the Dynamic packet Scheduling ratio to schedule the short and long flows for achieving the guaranteed throughput.

MATERIALS AND METHODS

Two queue threshold based approaches has been proposed (Wierman and Harchol-Balter, 2003) that gives highest service priority to the short flow. TCP flows are differentiated as short and long flows using a threshold value and short flows are en queued in one queue and remaining long flows are en queued in the second queue. Service priority is given to the first queue in First In First Out (FIFO) discipline and the second queue are only served if the first queue is empty. This approach reduces the mean transfer time however leads to starvation of long flows.

Bandwidth Adaptive Stratified Round Robin (BASRR) packet scheduling algorithm has been proposed in this study for enhancing quality of service of real-time multimedia applications. Embedded Network Processors (NP) has recently emerged with flexibility and speed to reduce the stress of the router by effectively processing the packets. The main objective of this study was to implement the proposed packet scheduling algorithm in a Network Processor (NP) based router for enhancing quality of service of real-time multimedia applications

In RuN2C (Avrachenkov *et al*., 2004), using TCP sequence number the packets are put in first queue or second queue. Packets from the second queue were not served unless the first queue was empty. Limitation of this protocol is TCP sequence number should start from a set of possible initial numbers and would lead to security problems such as IP address spoofing and session hijacking.

In LAS (Rai *et al*., 2004), the next packet to be served is one belonging to the flow that has received the least amount of service. By this definition, LAS will serve packets from a newly arriving flow until that flow has received an amount of service equal to the amount of least service received by flow in the system before its arrival. The long lived TCP flows competing against short TCP flows shows starvation in LAS. LAS reduces the loss rate for the short flows and approximately doubles the loss rate of long flows as compared with the loss rate under FIFO. Similarly, a Dynamic Packet Scheduling algorithm (Suresh *et al*., 2011) was proposed to treat the short and long flows in fair manner during the scheduling and avoids the starvation of long flows.

Another protocol Context Aware Transport/Network Internet Protocol (CATNIP) (Williamson and Wu, 2002) requires application layer information, the web document size to provide explicit context information to the TCP and IP protocol. While this approach violates the traditional layered Internet protocol architecture, it enables informed decision-making; both at network endpoints and at network routers, regarding flow control, congestion control and packet discard decisions.

Cprobe (Carter and Crovel, 1996) estimated the available bandwidth based on the dispersion of long packet trains at the receiver. A similar approach was taken in pipechar (Jin *et al*., 2001). The underlying assumption in these works is that the dispersion of long packet trains is inversely proportional to the available bandwidth. The dispersion of long packet trains does not measure available bandwidth in a path; instead, it measures a different throughput metric that is referred to as Asymptotic Dispersion Rate (ADR).

Another technique, called TOPP, for measuring available bandwidth was proposed in (Melander *et al*., 2000). TOPP uses sequences of packet pairs sent to the path at increasing rates. From the relation between the input and output rates of different packet pairs, one can estimate the available bandwidth and the capacity of the tight link in the path.

From the survey, it is revealed that accurately estimating the best profitable bandwidth value is very important for the scheduler to achieve the guaranteed throughput in the large scale network. The bandwidth estimation carried out by the scheduler is considered to

be a typical search optimization i.e., finding the most profitable bandwidth value among the available.

Evolutionary and Meta-heuristic algorithms (Afshar *et al*., 2007) have been extensively used as search and optimization tools in various problem domains. The broad applicability, ease of use and global perspective of meta-heuristic algorithms (Ghoul *et al*., 2007) may be considered as the primary reason for their extensive application and success as search and optimization tools in various problem domains. Among them, Genetic Algorithms (Jalilzadeh *et al*., 2009) have been extensively employed as search and optimization methods in various problem domains, including science, commerce and engineering (Ahrari *et al*., 2009). Genetic Algorithms are search and optimization procedures that are motivated by the principle of natural genetics and natural selection. Fundamental ideas of genetics are borrowed and used artificially to construct search algorithms that are robust and require minimal problem information. Over the last decade, modeling the behavior of social insects, such as ants (Ismail and Loh, 2009) and bees, for the purpose of search and problem solving has been the context of the emerging area of swarm intelligence.

Honey-bee (Karaboga and Akay, 2009) is among the most closely studied social insects. The intelligent behaviors of bee swarm such as bees foraging, bees mating, have inspired the researchers to develop new algorithms. In a recent work, Blum and Merkle (2008) developed an optimization algorithm based on the honeybee marriage process. Honey-bee mating is considered as a typical swarm-based approach to optimization, in which the search algorithm is inspired by the process of marriage in real honey-bee.

Another important behavior in Bees colony is Foraging concept. Foraging behaviour of Artificial Bee System (Bonabeau *et al*., 1999) is relatively new member of swarm intelligence. It tries to model natural behavior of real honey bee in food foraging. Honey bee uses several dancing methods to exchange information about location and profitability of food source. This food searching bee's behavior is a good candidate for developing new intelligent search algorithms. In the bee's foraging behavior, the best profitable food source can be found using the collective intelligence of forager bees. In similar way, the best profitable bandwidth value can be found from large solution space by mimicking the foraging behavior of honey bees.

In proposed G-DQS architecture, bee's foraging intelligence is adapted into the monitor component to find the best profitable bandwidth value for calculating the admissible flow to ensure the guaranteed throughput. Further, the proposed architecture uses the G-DQS algorithm to de queue the packets from the SFQ and LFQ based on the novel dynamic queue scheduling ratio. The proposed G-DQS is based on periodic rate-controlled streams, rather than window-controlled transmissions, allowing us to compare a certain rate with the available bandwidth more reliably and schedules the packets in two queues.

Proposed G-DQS architecture: Proposed scheduling algorithm G-DQS is suitable across varying traffic flows. The algorithm uses Dynamic Scheduling Ratio Q(r) for the efficient scheduling. Here our classification of short and long flow is as follows: Short flows are those with the flow size less than the threshold th and otherwise long flows. Flow size is the total number of packets or bytes of the flow i. We divide queue into two groups: SFQ and LFQ. If a flow i and its packets to be scheduled are less than the threshold th then the flow i is inserted in SFQ and if a flow i and its packets to be scheduled is not less than the threshold th then the flow i is inserted in LFQ. On the arrival of a packet if the buffer if full, a selected packet will be dropped using buffer stealin g (Zhang, 1995). Unlike Short Flow Highest Priority Scheduling algorithms, we use Dynamic Scheduling Ratio Q(r) to schedule the packets in the two queues. Q(r) decides the number of packets to be scheduled in each queue.

The Architecture components are explained as follows.

Classifier: On arrival of a packet p from flow i the classifier uses the threshold value th to each packet and dispatches it to the proper queue.

Monitor: The Monitor collect statistics about number of packets of all the flows in each queue and estimate the available bandwidth by adapting the forager bee's intelligence and report it to the Controller periodically.

Controller: Controller is the soul of G-DQS. It calculates dynamic scheduling ratio Q(r) for the each round based on the packets (packets) in two queue and commands scheduler to schedule the number of packets in each queue. It consists of a State Variable Counter DCI. The counter is initialized with the total number of flows in SFQ. Whenever the scheduler schedules packets from SFQ, the Q(r) value is decremented from DCi. The controller makes decision according to the information that other components report.

Scheduler: The scheduler decides the service order of packets in two queues according to the Controller's command.

Adapting forager bee's intelligence for finding the profitable bandwidth measure:

Input: All the flows in the SFQ and LFQ and it is denoted by Flow List (FL) = {FL1, FL_2,.., FL_n }.

Output: Profitable Bandwidth measure that provides the best throughput.

Algorithm:

- Initialize the flow list by choosing the random policy on the current flows in the SFQ and LFQ.
- Onlooker Agent (OA) in the Monitor initiates the Forager Agents (FA_n) corresponding to the number of flows in the FL.
- Each Forager Agent on its assigned flow does the following
- Each FA_i receives the flow data size (d_k) as reinforcement from the OA
- Each FA_i applies the following fitness function to find profitable bandwidth measure on its assigned flow

FA_i sends its d_k to destination host and receives the acknowledgement at time point t_k. The initial bandwidth measure can be computed using the following Eq. 1:

$$IBW_{FAi}(d_k) = \frac{d_k}{(t_k - t_{k-1})} \quad (1)$$

FA_i applies the smoothing filter called Exponential Weighting Moving Average (EWMA) on the initial bandwidth measure ($IBW_{FAi}(d_k)$) to find the final bandwidth measure ($FBM_{FAi}(d_k)$) using the following Eq. 2:

$$FBM_{FAi}(d_k) = IBW_{FAi}(d_{k-1}) * \beta + IBW_{FAi}(d_k) * (1 - \beta) \quad (2)$$

where, β value is chosen based on the current status in the network. It is computed as shown in the Eq. 3 and 4:

If $IBW_{FAi}(d_k) \geq FBM_{FAi}(d_{k-1})$ then:

$$\beta = \frac{FBM_{FA_i}(d_{k-1})}{IBM_{FA_i}(d_k)} \quad (3)$$

Else if $IBW_{FAi}(d_k) \leq FBM_{FAi}(d_{k-1})$ then:

$$\beta = \frac{IBM_{FA_i}(d_k)}{FBM_{FA_i}(d_{k-1})} \quad (4)$$

After each Forager Agent (FA_i) executed the step (3), it sends the computed $FBM_{Fai}(d_k)$ to the Onlooker Agent (OA). After receiving the profitable bandwidth measure from each agent (FA_i), it selects the best profitable final bandwidth measure.

According to the above algorithm, selected profitable bandwidth measure of OA is used by the Monitor component to find the admissible flow Fmax as follows.

Calculation of Fmax: Conceptually, G-DQS attempts to bound the instantaneous throughput of flows on an edge-to-edge basis to less than BWg. Fmax represents the maximum number of active flows which is to be scheduled on the edge-to-edge path without sacrificing the QOS.

To derive Fmax, an edge router first computes the predicted Flow Completion Time (FCT) and the throughput for a flow with no packet loss. Thus, the FCT of a short flow with size S_f can be computed using the Eq. 5:

$$FCT = C + DTx$$
$$FCT = 1.5 \times RTT + \log_2\left[\frac{sf}{MSS}\right] \times RTT \quad (5)$$

where, C is the time for connection establishment with the three-way handshake, DTx represents the data transmission time, MSS is the maximum segment size and RTT is the estimated end-to-end round trip time. Besides the computation of Eq. 1, the FCT for short-lived flows can also be gathered at the edge router through passive monitoring.

With S_f and FCT, the throughput T_f can be computed using the Eq. 6:

$$T_f = \frac{s_f \times (MSS + H)}{FCT \times MSS} \quad (6)$$

where, H is the estimated header size. Thus, the maximum number of active fast admitted flows on an edge to-edge path, Fmax, can be computed using the Eq. 7:

$$F_{max} = \frac{BW_g}{T_f} = \frac{BW_g \times FCT\,XMSS}{s_f \times (MSS + H)} \quad (7)$$

G-DQS algorithm: This algorithm is implemented by the scheduler

Begin: Differentiate TCP flows as short flows and long flows using threshold th and insert in two queues SFQ and LFQ respectively.

S:

- Calculate number of flows in SFQ = $\sum_{i=1}^{n} B_{SFQ}(\tau)$ and number of flows in
- LFQ = $\sum_{i=1}^{n} B_{LFQ}(\tau)$
- Initialize a State Variable Counter DC_i (r) = SFQ = $\sum_{i=1}^{n} B_{SFQ}(\tau)$
- Calculate Dynamic Packet Scheduling Ratio Q (r) $Q(R) = \frac{\sum_{i=1}^{n} B_{SFQ}(\tau) + \sum_{i=1}^{n} B_{LFQ}(\tau)}{\sum_{i=1}^{n} B_{LFQ}(\tau)}$
- Estimate the available bandwidth BWg on the edge to edge path using Eq. 2
- Using BWg calculate maximum number of flows to be on the path Fmax using Eq. 5

If Fmax >Q(r) and

- If $\sum_{i=1}^{n} B_{LFQ}(\tau)$ =0 then flows scheduled in the queue LFQ = Q(r) else flows scheduled in the queue SFQ = *Q(r)* and flows scheduled in the queue LFQ =1
- Perform DC_i (r) = DC_i (r) – Q(r)
- The main observation is: DC_i (r–1) –Q(r) = DC_i (r)
- If DC_i (r)> Q(r) then return to D: else return to S: for the calculation of Q(r) and *BWg* for the next round

If Fmax<Q(r) and:

- If $\sum_{i=1}^{n} B_{LFQ}(\tau)$ = 0 then flows scheduled in the queue LFQ = Q(r) else flows scheduled in the queue SFQ = Fmax
- Perform DC_i (r) = DC_i (r) -Fmax
- The main observation is DC_i (r-1)-fmax
- If DC_i (r)> Q(r) then return to D: else return to S: for the calculation of Q(r) and BWg for the next round

End

A state variable counter DC_i (r) is initialized with a value of total number of flows (packets) in the short flow queue SFQ. The dynamic scheduling ratio Q(r) is calculated from the number of flows in SFQ and LFQ. Maximum flows to be available on edge to edge path are calculated using BW_g (k) and Fmax. If Q(r)<Fmax then the flows in the SFQ are scheduled using the calculated value of Q(r) The Q(r) value of flows are scheduled in SFQ and one flow is scheduled in LFQ. This approach is continued until total number of flows in the queue SFQ becomes zero and then LFQ is completely scheduled. The condition DC_i (r) <Q(r) is checked during whenever the flows are scheduled in SFQ. If the condition is not satisfied then the new value of Q(r) and Fmax will be calculated for the next round.

If Q(r)>Fmax then the flows in the SFQ are scheduled using the calculated value of Fmax.The condition Q(r)>Fmax indicates that the number of flows is to be scheduled is more than the number of flows to be available on the edge to edge path. This is needed to provide a guaranteed throughput because when more flows are scheduled than Fmax the packet loss occurs and requires the retransmission of flows. This reduces the mean transmission time and throughput .Hence number of flows are scheduled in SFQ is Fmax. During this period flows are scheduled only from SFQ and not in LFQ. The G-DQS gives priority to short flows when the available bandwidth is minimum compared to the flows to be scheduled. The condition DC_i (r) <Fmax is checked whenever the packets are scheduled in SFQ. If the condition is not satisfied then the new value of Q(r) Fmax will be calculated for the next round.

This dynamic changing behavior of Q(r) is the main difference between our approach and other threshold approach. The Q(r) always changes according to the total number of flows in the two queues. This adaptive ratio Q(r) significantly improves the performance than the constant packet scheduling ratio used in QSPS (Paxson and Floyd, 1995) algorithm. The results shows that short TCP flows are treated without penalizing the performance of long TCP flows.

RESULTS

The practical networks normally will have many bottleneck links interconnecting the router. Hence, the proposed G-DQS is tested for their performance in the network topology with multiple bottleneck links. This model is shown in Fig. 1. In this configuration TCP sources traversing three bottleneck links and terminating at R3. The routers also shares the cross traffic. The bottleneck link capacities are 50Mbps, 30ms and other sources are connected with 10Mbps, 10ms. Here we refer to the packets that belonging to one TCP connection as a flow.

For studying the performance of Guaranteed-Dynamic Queue Scheduling (G-DQS), we test the following relations with the other protocols like First in First out (FIFO) and RuN2C.

Figure 2 Depicts that G-DQS reduces the mean transmission time of flows by treating long flows fairly. The transmission time of a flow is time interval starting when a first packet leaves a server and ending when a last packet of flow is reduced by the corresponding client.

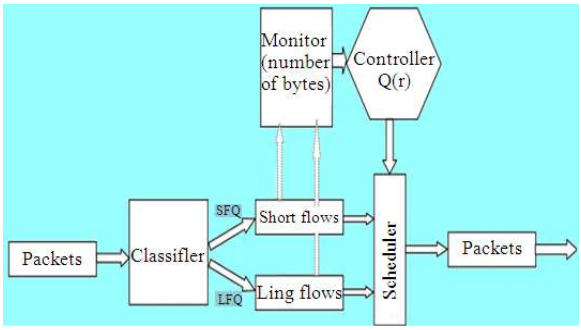

Fig. 1: G-DQS Architecture

It is evident from the figure that G-DQS approach reduces the mean transmission time compared to the simple FIFO and short flow highest priority scheduling. The transmission time of RuN2C almost same up to the threshold and G-DQS approach improves the performance for long flows. Figure 3 shows the mean transmission time of short flows. It indicates that the mean transmission time of flows with flow size less than the threshold under G-DQS is almost the same as that under RuN2C. But G-DQS shows better performance for short flows larger than the threshold. From Fig 4 we observe both G-DQS and RuN2C significantly reduce the mean transmission time of short flows compared with FIFO. As RuN2C follows strict short flow priority scheduling, the mean transmission time of short flows is minimum under G-DQS. Figure 5 Depicts comparison of Constant Q and Dynamic Q(r) in terms of the mean transmission time of number of flows using various constant Packet Scheduling Ratio values. When Q =1 increases the mean transmission time of large flows because the packets in two queues are served with the equal priority.

When Q value is increased to 5 the mean transmission time reduces up to the threshold value th and it is increases during large TCP flows. But the Dynamic Q(r) decreases the mean transmission time during large TCP flows. The mean transmission time is almost same up to the threshold when Q = 5 and in G-DQS. This shows G-DQS algorithm treats short flows fairly without penalizing the performance of long TCP flows.

Figure 6 shows the number of packets dropped for the various flow sizes. It is evident from the figure that short flows of size less than 40 packets not experiences packet loss in G-DQS, whereas for the similar size of flows FIFO experience packet loss. The packet loss is less in G-DQS compared to RuN2C for the large flows because of adaptive nature of the scheduling ratio used in G-DQS.

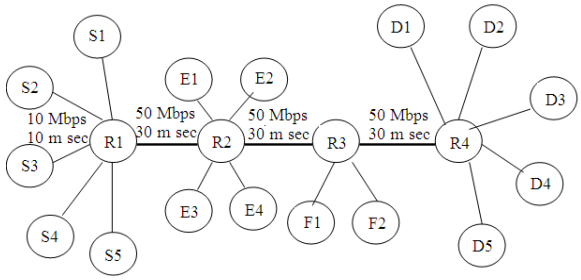

Fig. 2: Network topology with multiple bottleneck links

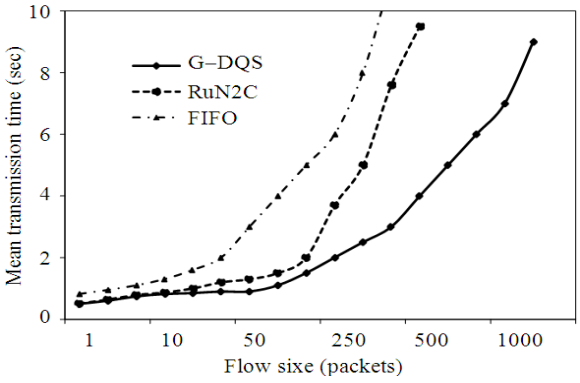

Fig. 3: Number of flow Vs mean transmission time

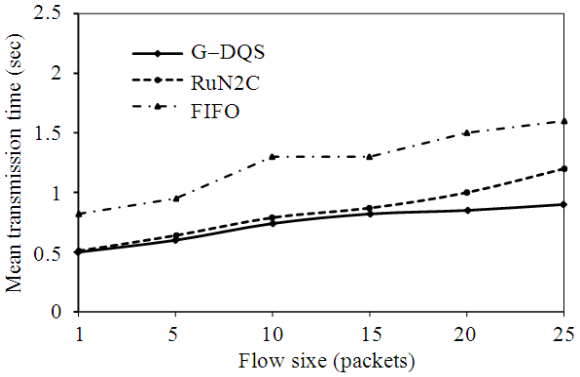

Fig. 4: Number of flow Vs mean transmission time

Figure 7 shows throughput of flows by the number of received packets per seconds (counting every 10 sec). We can see that the throughput of FIFO decreases suddenly during simulation period between 100 and 150 sec. This is because The FIFO and Ru2Nc approach schedules the packets not considering the large flows but in G-DQS packets are scheduled from two queues. Hence large flows are also getting service in addition to the short flows. This reduces the packet loss in large flows.

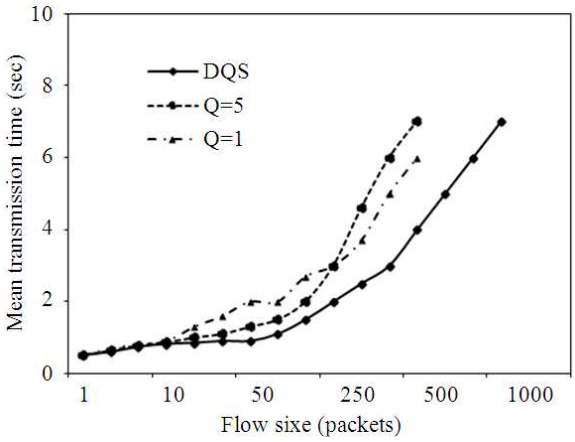

Fig. 5: Comparison of constant Q and dynamic Q(r)

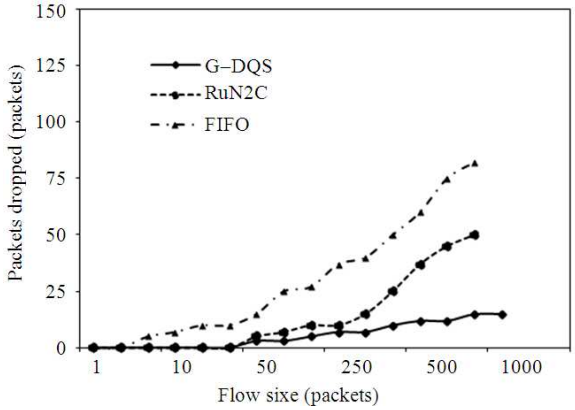

Fig. 6: Number of flows Vs packets dropped

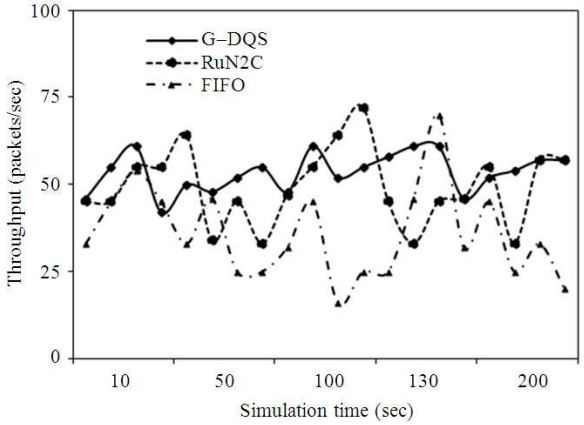

Fig. 7: Throughput Vs simulation time

The long flows are penalized and starved. It also indicates the throughput of FIFO and RuN2C is not constant during the complete simulation period. The throughput of proposed G-DQS is almost constant and it provides the guaranteed throughput.

DISCUSSION

The G-DQS avoids the starvation problem as shown in RuN2C because G-DQS schedules the packet both in short flows as well as long flows based on the bandwidth measure. This measure is estimated using swarm intelligence inspired honeybee's foraging behavior which provides the optimum profitable bandwidth compared to the other existing techniques. The proposed method reduces the mean transmission time and also packet loss compared with the other techniques as it uses the forager intelligence. It also shows that throughput does not change rapidly throughout the simulation time and it is almost constant. The proposed scheme is an aggregated flow scheduling and hence it can be adopted for large scale network.

CONCLUSION

Scheduling has been known for several years and attention has been given to use scheduling for the packet switched networks. In this study we presented a Forager bee's intelligence inspired Guaranteed-Dynamic Queue Scheduling approach namely G-DQS to improve performance of short flows without penalizing long flows much. Unlike other scheduling approaches, the TCP flows are scheduled with a Dynamic Packet Scheduling Ratio Q (r) and with accurate bandwidth estimation using forager bee's intelligence. This approach decreases the mean transmission time and packet loss of flows compared with the other protocols like FIFO and RuN2C scheduling. This algorithm can be deployed in edge router without complexity.

REFERENCES

Afshar, A., O.B. Haddad, M.A. Mariño and B.J. Adams, 2007. Honey-Bee Mating Optimization (HBMO) algorithm for optimal reservoir operation. J. Franklin Inst., 344: 452-462. DOI: 10.1016/j.jfranklin.2006.06.001

Ahrari, A., M. Shariat-Panahi and A.A. Atai, 2009. GEM: A novel evolutionary optimization method with improved neighborhood search. Applied Math. Comput., 210: 376-386. DOI: 10.1016/j.amc.2009.01.009

Avrachenkov, K., U. Ayesta, P. Brown and E. Nyberg, 2004. Differentiation between short and long TCP flows: Predictability of the response time. Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, Mar. 7-11, IEEE Xplore Press, pp: 762-773. DOI: 10.1109/INFCOM.2004.1356965

Bernet, C.Y., J. Binder, S. Blake, M. Carlson and B. Carpenter *et al*., 1999. A Framework for Differentiated Services. Cornell University, Internet Draft.

Blum, C. and D. Merkle, 2008. Swarm Intelligence: Introduction and Applications. 1st Edn., Springer, Berlin, ISBN-10: 3540740880, pp: 281.

Bonabeau, E., M. Dorigo and G. Theraulaz, 1999. Swarm Intelligence: From Natural to Artificial Systems. 1st Edn., Oxford University Press, New York, ISBN-10: 0195131584, pp: 307.

Braden, B., D. Clark and S. Shenker, 1994. Integrated service in the internet architecture: An overview. Center for Technology, Policy and Industrial Development, MIT Libraries.

Carter, R.L. and M.E. Crovell, 1996. Measuring bottleneck link speed in packet-switched networks. Perfor. Evaluat., 27-28: 297-318. DOI: 10.1016/S0166-5316(96)90032-2

Chen, X. and J. Heidemann, 2003. Preferential treatment for short flows to reduce web latency. Comput. Netw., 41: 779-794. DOI: 10.1016/S1389-1286(02)00439-5

Floyd, S., 2001. A report on recent developments in TCP congestion control. IEEE Commun. Mag., 39: 84-90. DOI: 10.1109/35.917508

Ghoul, R.H., A. Benjelloul, S. Kechida and H. Tebbikh, 2007. A scheduling algorithm based on petri nets and simulated annealing. Am. J. Applied Sci., 4: 269-273. DOI: 10.3844/ajassp.2007.269.273

Guo, L. and I. Matta, 2002. Differentiated control of web traffic: A numerical analysis. SPIE Int. Soc. Optical Eng., 184-194.

Ismail, Z. and S.L. Loh, 2009. Ant colony optimization for solving solid waste collection scheduling problems. J. Math. Stat., 5: 199-205. DOI: 10.3844/jmssp.2009.199.205

Jalilzadeh, S., H. Shayeghi, M. Mahdavi and H. Hadadian, 2009. A GA based transmission network expansion planning considering voltage level, network losses and number of bundle lines. Am. J. Applied Sci., 6: 987-994. DOI: 10.3844/ajassp.2009.987.994

Jin, G., G. Yang, B.R. Crowley and D.A. Agarwal, 2001. Network Characterization Service (NCS). Proceedings of the 10th IEEE Symposium on High Performance Distributed Computing, Aug. 07-09, IEEE Xplore Press, San Francisco, CA, USA., pp: 289-299. DOI: 10.1109/HPDC.2001.945197

Karaboga, D. and B. Akay, 2009. A survey: Algorithms simulating bee swarm intelligence. Artif. Intell. Rev., 31: 61-85. DOI: 10.1007/s10462-009-9127-4

Melander, B., M. Bjorkman and P. Gunningberg, 2000. A new end-to-end probing and analysis method for estimating bandwidth bottlenecks. Proceedings of the IEEE Global Telecommunications Conference, Nov. 27-Dec. 01, IEEE Xplore Press, San Francisco, CA, USA., pp: 415-420. DOI: 10.1109/GLOCOM.2000.892039

Paxson, V. and S. Floyd, 1995. Wide area traffic: The failure of Poisson modeling. IEEE/ACM Trans. Netw., 3: 226-244. DOI: 10.1109/90.392383

Rai, I.A., E.W. Biersack and G. Urvoy-Keller, 2005. Size-based scheduling to improve the performance of short TCP flows. IEEE Netw., 19: 12-17. DOI: 10.1109/MNET.2005.1383435

Rai, I.A., G. Urvoy-Keller, M. Vernon and E.W. Biersack, 2004. Performance models for LAS-based disciplines in a packet switched network. Proceedings of the SIGMETRICS/Performance, Jun. 12-16, ACM, New York, USA., pp: 1-12.

Suresh, Y., S. Arumugam and M.A. Bhagyaveni, 2011. An Efficient Dynamic Queue Scheduling (DQS) on classified TCP flows for the internet traffic. Eur. J. Sci. Res., 55: 481-486.

Wierman, A. and M. Harchol-Balter, 2003. Classifying Scheduling policies with respect to unfairness in an M/GI/1. Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, Jun. 10-14, ACM, San Diego, CA, USA., pp: 238-249. DOI: 10.1145/781027.781057

Williamson, C. and Q. Wu, 2002. A case for context-aware TCP/IP. ACM Performance Evaluat. Rev., 29: 11-23. DOI: 10.1145/512840.512843

Zhang, H., 1995. Service disciplines for guaranteed performance service in packet-switching networks. Proc. IEEE, 83: 1374-1396. DOI: 10.1109/5.469298