Resource allocation under channel uncertainty

C. Manikandan and Srikrishna Bhashyam
Department of Electrical Engineering,
Indian Institute of Technology Madras,
Chennai 600036
mani2004c@gmail.com, skrishna@ee.iitm.ac.in

Rajesh Sundaresan
Department of Electrical Communication Engineering,
Indian Institute of Science,
Bangalore 560012
rajeshs@ece.iisc.ernet.in

***Abstract*— Scheduling policies in multi-queue multi-server systems need to allocate servers based on the channel state information and the queue state information. In the downlink, channel state information available to the scheduler may be imperfect due to feedback delay and estimation errors. Motivated by this, we consider the downlink scheduling problem of allocating servers to multi-queue multi-server systems under channel uncertainty. We propose new policies which allocate the servers based on the predicted channel information. Simulations indicate that our policies have better delay and backlog properties than a policy proposed by Kar, Luo & Sarkar [1].**

I. Introduction

In this paper, we consider resource allocation in multi-queue multi-server systems. We propose new policies for resource allocation when the channel state information available is imperfect. The multi-queue multi-server model can be used for the downlink of packet data systems based on orthogonal frequency division multiplexing (OFDM) or code division multiple access (CDMA). In an OFDM-based system, each subcarrier or a group of subcarriers can be modeled as a server. In a CDMA-based system, each spreading code can be modeled as a server. Therefore, the code allocation and subcarrier allocation problems in CDMA and OFDM are special cases of the problem considered here.

Andrews et al [2] proposed resource allocation policies for multi-queue single server systems, particularly for CDMA packet systems such as 1xEV-DO Rev 0 [3] where all available codes are allocated to a single user in each time slot. Kumaran & Viswanathan [4] and Agarwal et al [5] proposed scheduling of multiple users in each slot for CDMA-based systems where the codes could be considered as multiple servers. Downlink scheduling for such systems is quite well-studied. A sparse sampling of the works in this area, relating to those closely relevant to this paper, is as follows. Kittipiyakul & Javidi [6] proposed an optimal server allocation policy to minimise average delay under time-varying on-off connectivities. In particular, they showed that the maximum-throughput load-balancing (MTLB) policy that achieves maximum instantaneous throughput and simultaneously balances the load across queues is optimal for an on-off channel model. For a general channel, the existence of an MTLB policy and its capability to minimise average delay (when MTLB exists) are still open questions. Kittipiyakul & Javidi [7] proposed and studied a heuristic extension of the MTLB policy for a general channel. Mohanram & Bhashyam [8] considered joint power and server allocation to maximise throughput.

Tassiulas & Ephremides [9] characterised the stability region and proposed a policy that would stabilise the queues, if at all it was possible to stabilise them. Their policy did not depend on knowledge of arrival rates, and roughly speaking routed traffic from the longest queue to the shortest queue among connected links. The connections were either on or off and were independent and identically distributed (iid) from slot to slot. In the presence of channel uncertainty, the probability of a connection was factored into the weights. The states of all links in a slot were made available to the scheduler at the end of the slot. Kittipiyakul & Javidi [7] considered multi-packet transmission per server (all packets from the same queue), but assumed that channel state information for a slot was available to the scheduler prior to the start of the slot. Kar et al [1] considered a practical setting where channel state information is available only once every T slots and channel fading is modeled by a Markov process. They proposed a policy based on virtual queueing, and showed that within such a framework, if there is some policy that will stabilise a set of arrival rates, then so will their policy. The virtual queueing enables easy computation of the best virtual-queue-based schedule.

In this paper, we propose two policies for Markov channels under channel uncertainty. They may be thought of as extensions of the policies in Kittipiyakul & Javidi [7] to the uncertain channel case. One of our proposed policies stabilises the queue states for a set of arrival rates if at all any policy can; this was not addressed by Kittipiyakul & Javidi [7] even for the full information case. Our policies go beyond the virtual queueing framework, and is superior to the one of Kar et al [1] at low rates, as demonstrated by simulations results that compare average backlog and delay across policies.

The rest of the paper is organised as follows. Section II describes the system model and Section III the proposed policies. Section IV makes some remarks about the stability region for the system, Section V presents simulation results and Section VI our conclusions.

II. System model

Consider a downlink system of N users with one queue each. Each downlink queue may be thought of as a *class* in the parlance of Tassiulas & Ephremides [9]. There are K servers serving these queues. Transmission is slotted. In each slot, the scheduler decides how the servers are allocated across users. Users then transmit their packets on the assigned servers. The effect of the fading channel is modeled by the physical layer's ability to transmit a certain number of packets from the set $\mathbb{C} = \{0, 1, \cdots, c_{\max}\}$. As each user may see a different channel on each server (as in the OFDM case), the instantaneous channel state at slot t may be modeled as $C(t) \in M^{N \times K}(\mathbb{C})$, a matrix of size $N \times K$ with entries from $\mathbb{C}$. An element of the matrix $C_{nk}(t)$ denotes the number of packets user n can transmit on server k at slot t. For simplicity we assume that $\mathbb{C}$ is the same for all user-server pairs. The process $(C(t) : t = 0, 1, \cdots)$ is assumed to be an ergodic Markov chain.

Let $b(t) = (b_1(t), \cdots, b_N(t))^\dagger$ denote the queue state vector at slot t where $b_n(t)$ is the queue size of user n at slot t. Backlog is defined as the sum of the components of the queue state vector.

Scheduling is done based on available channel and queue state information. Success or failure of a transmission is known only upon explicit feedback from the receiver. Similarly channel state information in frequency-division duplexed systems is known only upon explicit feedback from the receiver. To model these delays, we assume that channel states and the results of transmissions are known only once every T slots, which we call an interval. (The same model was used by Kar et al [1]). More precisely, let the $(l-1)$st interval be made of slots $(l-1)T, (l-1)T+1, \cdots, lT-1$. For scheduling decisions in the lth interval, exact queue states and exact channel states are assumed known for slot $lT - 1$. Decisions for other slots in this interval have to be made at the start of the interval.

In conformance with existing wireless systems, we assume that a server can serve at most one queue in a slot. A queue however may connect to several servers. We may therefore think of a bipartite graph with queues on the left side and servers on the right with connections only between queues and servers, and the degree of any server being at most 1.

The scheduler decides the connections between queues and servers at each slot t in the interval, subject to the degree constraint. Following terminology in [1], we refer to the set of connections meeting the constraints as *polymatching*. The scheduler further decides $R_{nk}(t)$, the number of packets that flows across the connection n to k. If $R_{nk}(t) \le C_{nk}(t)$, the connection capacity, then the packets are received correctly by the receiver. Otherwise the entire transmission fails. Thus the number of received packets is $R_{nk}(t) 1\{R_{nk}(t) \le C_{nk}(t)\}$, where $1\{\cdot\}$ is the indicator function of an event. This loss model is motivated by systems that encounter outage if transmission of data rate is higher than the unknown instantaneous link capacity for the slot, and is different from the optimistic model of Kar et al [1] where $\min\{R_{nk}(t), C_{nk}(t)\}$ is assumed received.

III. Scheduling Algorithms

We now present our policies for allocation under channel uncertainty. For slot $lT + m$ in the lth interval, $0 \le m < T$, define

$$\tilde{C}_{nk}(lT+m) = \max_r r \Pr\{r \le C_{nk}(lT+m) \mid C_{nk}(lT-1)\}, \tag{1}$$

and $R_{nk}(lT + m)$ to be the argument that achieves the maximum. $\tilde{C}_{nk}(lT+m)$ is the maximum expected throughput for user n on server k in slot $(lT + m)$, given the channel information for slot $C_{nk}(lT - 1)$.

Policy 1:

```
1) Assign $w_n \leftarrow b_n(lT - 1)$ for $1 \le n \le N$.
2) Repeat the following for each slot $m$ in the $l$th interval, $0 \le m < T$.
   a) Form the complete bipartite graph where every queue is connected to every server.
   b) Let $X$ denote the set of unallocated servers.
   c) Initialise $X = \{1, 2, ....., K\}$
   d) While[1] $X \ne \phi$
      i) Assign
            $(n^*, k^*) \leftarrow \arg\max_{n,k} w_n \tilde{C}_{nk}(lT + m)$.
      ii) Skip. (Policy 2 is different in this step.)
      iii) Choose the connection $(n^*, k^*)$ for the polymatching and let $R_{n^*k^*}(lT + m)$ packets be transmitted in this slot[2].
      iv) Packets may be retransmitted. Packets are chosen according to the lexicographical order based on the pair $(v, s)$ that is maintained for each packet, where $v$ is the number of times a packet was transmitted and $s$ is the sequence number[3].
      v) Remove the server $k^*$ from the set $X$.
3) Update queue states and channel states based on information from the receivers at the last slot of the interval and reset $v$ to 0 for all the packets at the start of every interval.
```

Remark: Observe that in Step 2.d), because the weights w_n do not change, the search for queue-server connections separates into K independent searches, one for each server, i.e., to each server k, connect the queue

$$n^*(k) \leftarrow \arg\max_n w_n \tilde{C}_{nk}(lT + m).$$

The search simplifies to a small extent, because the search for the maximum is restricted to within a set with at most N elements.

Policy 2: This policy is the same as Policy 1, except for the following insertion:

- **Step 2.d.ii)**: Update

$$w_{n^*} \leftarrow [w_{n^*} - \tilde{C}_{n^*k^*}(lT + m)]_+,$$

[1] Repeat the algorithm until all the servers are allocated

[2] The queue that has the best queue-size weighted throughput on the server is chosen.

[3] Preference is thus given to packets transmitted the fewest number of times v, and amongst those transmitted the same number of times, to the one with the smallest sequence number s, i.e., the earliest to arrive. The retransmissions may occur on different servers.

where $[x]_+$ is 0 if $x < 0$ and x if $x \geq 0$.

If *all* w_n are zero, then reset $w_n \leftarrow b_n(lT-1)$ for $1 \leq n \leq N$.

The motivation for these changes is that the weights can be adapted to the decisions taken, based on the expected number of packets that will go through. When all weights become zero, then all packets have been transmitted roughly equal times, and we may resume retransmissions with original weights.

The Kar-Luo-Sarkar (KLS) policy [1]: This policy maintains K virtual queues at each queue, one for each server, in addition to the input queue. In all, there are $NK + N$ queues. Arrivals that occur during the intermediate slots of an interval are held in the input queue and allowed to enter the virtual queues only at the start of the interval. This virtual queueing reduces the multi-server problem to K single-server problems. Let Q_{nk} denote the state of that virtual queue of user n associated with server k.

1) Queueing: At the start of lth interval, all input-queued packets of user n will enter this user's queue associated to server k if

$$k = \arg\min_{k'} Q_{nk'}(lT-1).$$

2) Service: Compute for every queue-server pair, the weight given by

$$\hat{C}_{nk}(lT) = \frac{1}{T}\mathbb{E}\left[\sum_{t=lT}^{(l+1)T-1} C_{nk}(t) \middle| C_{nk}(lT-1)\right]$$

3) To server k, assign the virtual queue

$$n^*(k) \leftarrow \arg\max_n \hat{C}_{nk}(lT)Q_{nk}(lT-1).$$

4) $R_{n^*(k)k}(lT+m)$ packets will go through in slot $(lT+m)$ where $R_{nk}(lT+m)$ is the argument that maximises equation (1).

For the KLS policy in [1] no loss model is incorporated. Therefore, average number of packets that go through is assumed to be $\hat{C}_{nk}(lT)$. To use their policy along with our loss model, we use their polymatching and choose the transmission rate such that the expected throughput is maximum for each connection given the polymatching. Thus, for connection (n,k), the rate attempted is $R_{nk}(lT+m)$ which is the same rate attempted in our proposed policies provided they chose this connection. The difference between our proposed policies on the one hand and the KLS policy on the other is that the KLS policy fixes the connections for the entire interval. Our proposed policies adapt them to changing estimated queue-size weights and predicted channel conditions.

IV. Stability region

The definition of stability region is the standard one given by Tassiulas & Ephremides [9, Defn. 1]. The space of queue states may be partitioned into sets $T, R_1, R_2, \cdots$ where T is the set of transient states and $R_j, j = 1, 2, \cdots$ are closed sets of communicating states. A system is stable if the queue state process satisfies the following: (1) it exits T in finite time with probability 1, when initialised in any one of the transient states; and (2) all other states are positive recurrent. The stability region of a policy is the set of arrival rate vectors $\lambda = (\lambda_1, \cdots, \lambda_N)^\dagger$ under which the system remains stable under the policy. The stability region of the system is the set of arrival rate vectors for which there exists a policy under which the system is stable. An optimal policy is one that stabilises the system for any arrival vector in the stability region. The work of Tassiulas & Ephremides [9] established that an optimal policy exists for a wide class of multi-class multi-queue network under perfect information available with one slot delay. The KLS policy [1] is optimal under the virtual queueing framework and with channel uncertainty. Using techniques similar to those of Tassiulas & Ephremides [9] and [1], we can show that our Policy 1 is optimal under channel uncertainty. Further, our policies are not restricted to within the virtual queueing framework, and are therefore likely to fare better for low rates; simulations indicate that our policies outperform the KLS policy at low rates in terms of the backlog performance metric.

V. Simulation Results

Consider a system with $N = 6$ users and $K = 4$ servers. The channel state, in terms of the number of transmissible packets in a slot, is modeled as a Markov chain. This Markov chain is composed of smaller independently evolving and identical Markov chains on four states, one for each user-server pair. So $\mathbb{C} = \{0, 1, 2, 3\}$. The transition probabilities are given by the matrix

$$\begin{bmatrix} 0.98 & 0.0067 & 0.0067 & 0.0067 \\ 0.0067 & 0.98 & 0.0067 & 0.0067 \\ 0.0067 & 0.0067 & 0.98 & 0.0067 \\ 0.0067 & 0.0067 & 0.0067 & 0.98 \end{bmatrix}$$

and the initial distribution is the stationary distribution on each user-server pair. Arrivals to queues are truncated Poisson with a maximum of 100 arrivals. Other parameters depend on whether the scenario is *symmetric* or *asymmetric*. In the symmetric case, all users have the same mean number per slot λ. In the asymmetric case, users 1, 3, and 5 have mean arrivals per slot of λ and users 2, 4, and 6 have mean arrivals per slot of $\lambda/2$. The abscissa in all plots is the mean total arrivals per slot, summed over all users. Backlog and delay are used as metrics for comparison. Backlog is measured only at interval boundaries. Simulations assume a finite queue size of 1000. Packets arriving at a full queue are dropped, and therefore do not contribute to either backlog or delay.

A. *Average backlog comparison for a fixed T*

Average backlog in packets/time-slot/user is calculated and plotted for all described policies against total arrival rate in Figures 1, 2, 3, and 4. T is set to 8. The symmetric case is plotted in Figures 1, 2, and asymmetric case in Figures 3 and 4. From the plots, we infer that all the policies have similar performance at high rates. However, our proposed policies outperform the KLS policy at low rates as can be seen from the magnifications in Figures 2 and 4. Simulation results for other values of T demonstrate the same qualitative behaviour.

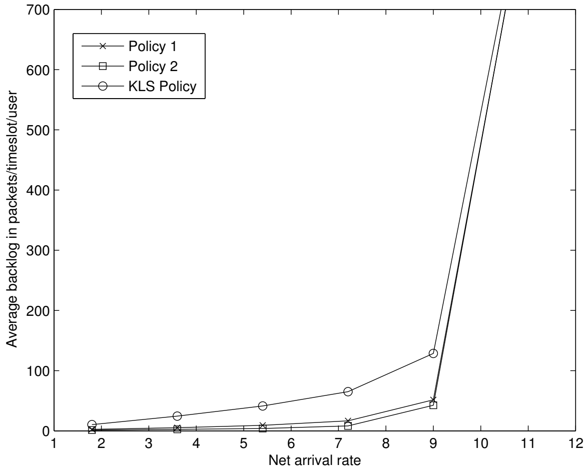

Fig. 1. Average backlog comparison for T=8 for symmetric arrivals at high traffic

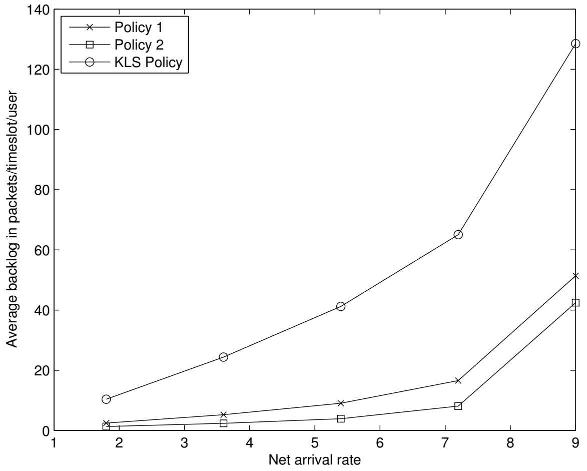

Fig. 2. Average backlog comparison for T=8 for symmetric arrivals at low traffic

B. Delay comparison for a fixed T

Figures 5 and 6 plot the delay of all transmitted packets for the symmetric and asymmetric arrival cases, respectively. For each policy, the figures contain a best and worst case value for the delay distribution at each value of delay. We observe that the proposed policies give significantly better delay performance than the KLS policy for both symmetric and asymmetric arrival cases. Results are shown for net arrival rate of 6 and 4.5 for symmetric and asymmetric arrivals respectively. Similar results were obtained for other rates in the range 2 to 9.

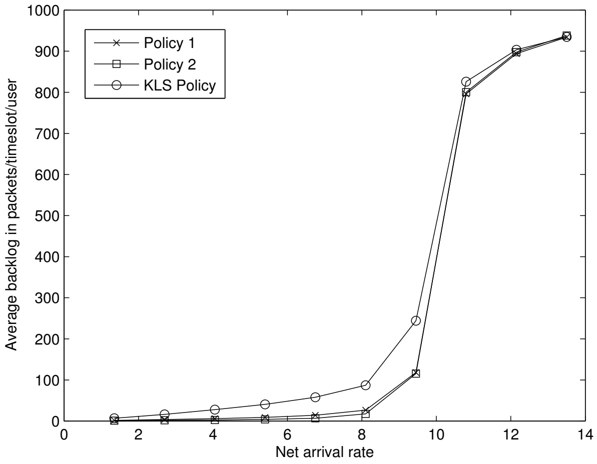

Fig. 3. Average backlog comparison for T=8 for asymmetric arrivals at high traffic

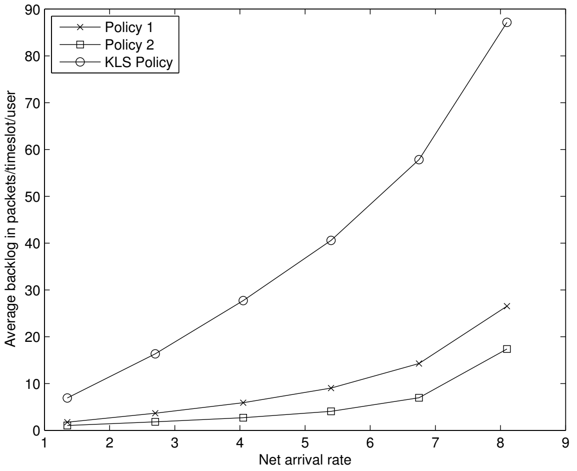

Fig. 4. Average backlog comparison for T=8 for asymmetric arrivals at low traffic

C. Average backlog comparison vs T for a given algorithm

Figures 7 and 8 plot backlog of Policy 2 and the KLS policy, respectively, across T. The plots are for symmetric arrivals. The behaviour in case of asymmetric arrivals is qualitatively the same. It is interesting to note that the stabilisable sum rate remains roughly the same for all the considered T. Its exact dependence on T remains to be studied. The transition behaviour from a stable system to unstable queues is different across T; the performance degrades quite clearly with increasing T in both figures.

VI. Conclusions

We proposed downlink scheduling policies for multi-queue multi-server systems under channel uncertainty. Our policies provide better throughput at low rates than the virtual-queueing-based KLS policy. Joint power control and schedul-

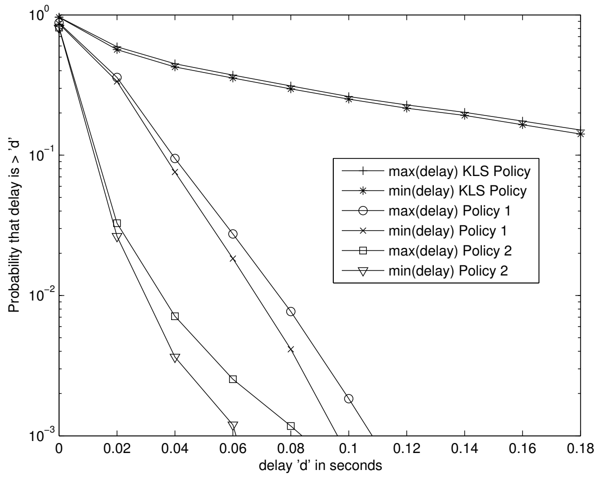

Fig. 5. Delay comparison for symmetric arrivals, net arrival rate=6

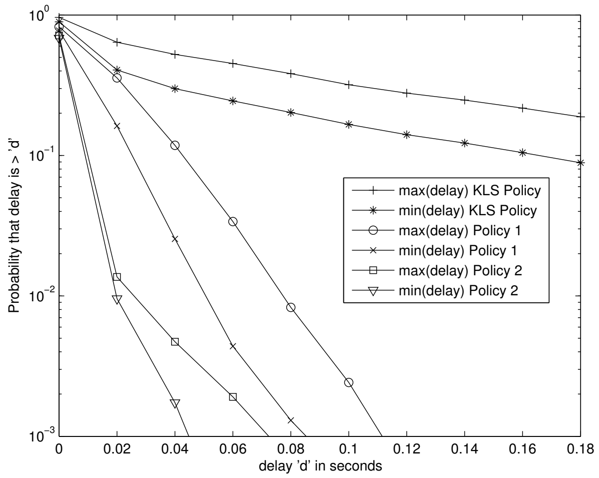

Fig. 6. Delay comparison for asymmetric arrivals, net arrival rate=4.5

ing, under channel uncertainty, is currently being investigated.

References

[1] K. Kar, X. Luo, S. Sarkar, "Throughput-optimal scheduling in multichannel access point networks under infrequent channel measurements," *Proceedings of IEEE Infocomm 2007*, May 2007.

[2] M. Andrews, K. Kumaran, K. Ramanan, A. L. Stolyar, R. Vijayakumar, P. Whiting, "Providing quality of service over a shared wireless link," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 150-154, Feb 2001.

[3] "TIA/EIA/IS-856 cdma2000 High Rate Packet Data Air Interface Specification," Telecommunications Industry Association, (www.3gpp2.org)

[4] K. Kumaran, H. Viswanathan, "Joint power and bandwidth allocation in downlink transmission," *IEEE Transactions on Wireless Communications*, vol. 4, no. 3, pp. 1008-1016, May 2005.

[5] R. Agarwal, V. Subramanian, R. Berry, "Joint scheduling and resource allocation in CDMA systems," 2^{nd} *workshop on modeling and optimization in mobile, adhoc and wireless networks (WiOPT '04)*, Cambridge, UK, Mar 2004.

[6] S. Kittipiyakul, T. Javidi, "Delay-optimal server allocation in multiqueue multi-server systems with time-varying connectivities," Submitted to *IEEE Transactions on Information Theory*

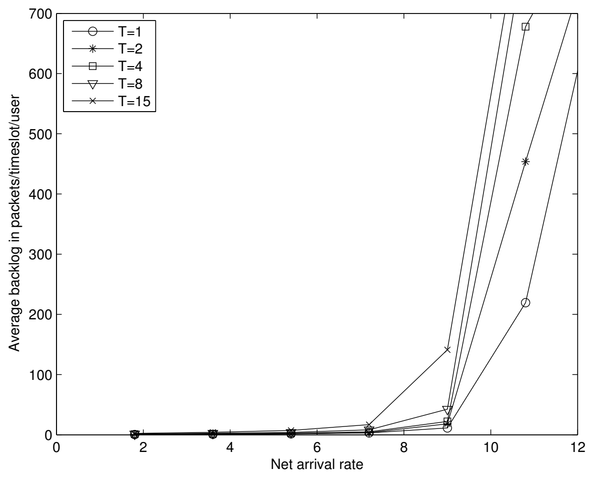

Fig. 7. Average backlog of policy 2 vs T for symmetric arrivals

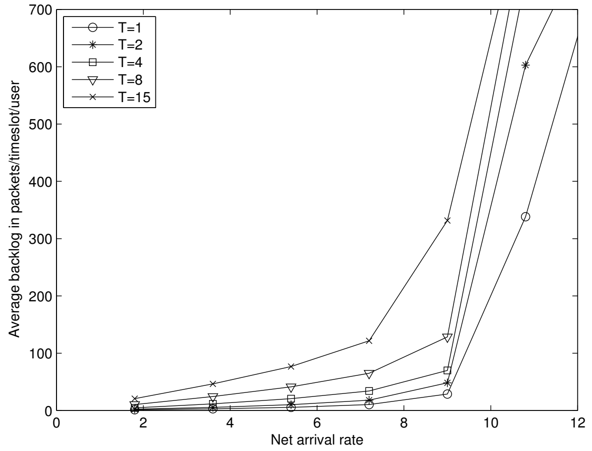

Fig. 8. Average backlog of KLS policy vs T for symmetric arrivals

[7] S. Kittipiyakul, T. Javidi, "Resource allocation in OFDMA with timevarying channel and bursty arrivals," *IEEE Communication letters*, vol. 11, no. 9, September 2007.

[8] C. Mohanram, S. Bhashyam, "Joint subcarrier and power allocation in channel-aware queue-aware scheduling for multiuser OFDM," *IEEE Transactions on Wireless Communications*, vol. 6, no. 9, September 2007.

[9] L. Tassiulas, A. Ephremides, "Stability properties of constrained queuing systems and scheduling for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, 1992, pp. 1936-1949.