Scheduling in Multi-Channel Wireless Networks: Rate Function Optimality in the Small-Buffer Regime

Shreeshankar Bodas, Sanjay Shakkottai, Lei Ying, R. Srikant

***Abstract*—The problem of designing scheduling algorithms for a multi-channel (e.g., OFDM-based) wireless downlink network is considered. The classic MaxWeight algorithm, although throughput-optimal, results in a very poor per-user delay performance in such systems. Hence, an alternate class of algorithms called iLQF (iterated Longest Queues First) is proposed for overcoming this issue. The iLQF-class algorithms are analyzed in a number of different system configurations. A particular algorithm in this class, called iLQF with PullUp, is shown to be rate function optimal for the problem in an appropriate large deviations setting, and is shown to result in a strictly positive value of the rate function for a number of modifications to the basic system model. Thus, the proposed algorithm yields provable performance guarantees. The analytic results are confirmed through simulations.**

***Index Terms*—Delay optimality, large deviations, perfect matchings, random bipartite graphs, scheduling algorithms, small buffer**

I. Introduction

With the ever-growing numbers and capabilities of mobile communication devices, the service providers are required to support a variety of delay-sensitive and possibly high data-rate applications such as VoIP, GPS and streaming video. The fundamental network resources (time/frequency/codes) that these applications demand are, however, limited and must be shared. Therefore, designing efficient scheduling algorithms is an important problem in the study of wireless networks. Good scheduling algorithms are often critical in determining the performance of a communication system. In this paper, we look at the problem of designing scheduling algorithms for the downlink of a wireless network.

The 4G-systems such as WiMax [2] and LTE [3] employ an Orthogonal Frequency Division Multiplexing(OFDM)-based wireless downlink. In these systems, the bandwidth available at the base-station is partitioned into several tens to hundreds of orthogonal channels (frequency bands). Each channel can be potentially allocated to different users over time, but only one user per timeslot, which is typically of the duration of a few milliseconds. The channels support user- and time-dependent data rates. The scheduling decision required at the base-station is an allocation of the channels to the users in each timeslot. In this paper, we study this service allocation problem in detail.

The scenario described in the previous paragraph can be modeled as a multi-queue queuing system with as many servers as the number of channels (frequency bands). The queues correspond to the per-user queues at the base station. The queues temporarily store the incoming data packets destined for the respective users. A given queue can be served by any number of servers (i.e., a particular user can be allocated any number of frequency bands), but a given server can serve only one queue in a given timeslot. When allocated, a server can serve a time-dependent number of packets from the corresponding queue. A server allocation policy is a rule for allocating the servers to the queues over time, obeying the scheduling constraints.

A possible server allocation policy for this system is the classic MaxWeight rule [4]. In out setting, the MaxWeight rule allocates a server to that queue that results in the maximum value of the product of the queue-length and the corresponding channel rate. It is well-known that this server allocation rule is throughput-optimal for the system, i.e., it makes all the queues stable (positive recurrent) under a given vector of arrival rates, if there is *any* other scheduling rule that can do so, under very mild assumptions on the arrival and channel processes. This scheduling rule has received considerable attention [5], and has been analyzed in a number of regimes. Researchers have established several of its performance properties in the large-queue [6], [7], [8], [9] or heavily loaded [10], [11], [12] regimes. However, in a multi-carrier regime with large bandwidth (a scenario that is typically anticipated in 4G-systems), one is interested in developing algorithms that ensure small queues at the base-station. The small-queue performance of the MaxWeight algorithm is not clear. For example, consider a system with 100 channels, each of which can serve one packet per timeslot. Suppose that there are 3 users in the system, with user 1 having 100 packets in its queue and the other two users having 99 packets each. It is easy to show that the MaxWeight algorithm allocates all the available channel resources to user 1. This would result in user 1's queue length

This research was partially supported by NSF grants CNS-0721380, CNS-0964391, CNS-1017549, CNS-1161868, CNS-1264012 and CNS-1262329.

S. Bodas is with Qualcomm, Inc. This research was carried out when he was a graduate student at The University of Texas at Austin, and a postdoctoral associate at Massachusetts Institute of Technology. E-mail: shreeshankar.bodas@utexas.edu

S. Shakkottai is with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712, USA. E-mail: shakkott@austin.utexas.edu

L. Ying is an Associate Professor at the School of Electrical, Computer and Energy Engineering at Arizona State University, Tempe, AZ 85287, USA. E-mail: lei.ying.2@asu.edu

R. Srikant is with the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA. E-mail: rsrikant@illinois.edu

A shorter version of this paper appeared in the proceedings of the ACM SIGMETRICS/Performance Conference, June 2009 [1].

decreasing to zero, but the other two queue lengths are still large. Thus, it intuitively seems better to share the channel resources among all users in order to reduce the peak queue length at the end of the slot.

A key insight emerging from this paper is that, for small-buffer multi-channel systems, scheduling needs to be iterative in each timeslot – as resources (channels) get allocated to the users, the effect of this allocation (i.e., that the queue-lengths for these users would decrease) needs to be factored in while making allocation decisions for the remaining channels. We develop a class of iterative algorithms (namely, iLQF – iterated Longest Queues First) for scheduling in large multi-carrier systems, that manifests this point. While we have recently shown that iLQF is throughput-optimal [13], the focus of this paper is on rate-function optimality. In particular, while many algorithms (including the classical MaxWeight rule) are throughput-optimal, rate-function optimality (corresponds to fastest decay rate of queue-length tail probability) is much harder to achieve. In this paper, we show that for a queuing system with a symmetric Bernoulli arrival process, the iLQF-class algorithms (with certain additional properties) are rate-function optimal in the many-channels regime (the rate-function is defined in Section IV). Roughly speaking, this means that for a system with a large number of channels (such as a multi-channel OFDM system), the proposed algorithms "minimize" the probability of the maximum queue length (across users) exceeding *any nonnegative* queue-length threshold b, and where *this threshold* b *does not scale with system size*. We also consider the case of asymmetric arrival rates (where the arrival rates to the users are not identical to each other), as discussed in Section IX.

The following is a summary of our main contributions in this paper:

- We consider the small-buffer overflow problem in a multi-user, multi-carrier system, in a large deviations setting (see Section IV for details). We establish an upper bound on the rate function for the small buffer overflow event (defined in Section IV) under *any* scheduling algorithm (Section V).
- We present a class of algorithms called iLQF (iterated Longest Queues First) for scheduling in these systems (Section VII). We show that under certain technical conditions, the algorithms in this class meet the upper bound on the rate function, and are therefore *rate function optimal* for the problem under consideration.
- We present a particular rate-function-optimal algorithm in this class, called iLQF with PullUp (Section VIII).
- We show that the proposed iLQF with PullUp algorithm is robust to changes in the system model by showing that the algorithm results in a strictly positive value of the rate function under a number of variations of the basic system model (Section IX).

Section X presents simulation results comparing the proposed iLQF with PullUp algorithm with the classic MaxWeight algorithm. We conclude with a summary of the paper and future work in Section XI.

II. Related Work

Multi-user scheduling in wireless networks has received a lot of interest over the past few years [14], [4], [5], [15], [16], [17], [18]. Recent progress in studying the performance of scheduling algorithms includes the characterizations of the queue-performance in heavy-traffic limits [10], [11], [12], and computations of the tail probability of queue-lengths using the large-deviations analysis [6], [7], [8], [9]. Order-optimality in the number of flows under the MaxWeight algorithm has been explored in [19]. While these results provide very useful insights into the QoS of scheduling algorithms, theoretically, a majority of the prior results are valid only when the queue-lengths increase to infinity, i.e., in a large-queue regime.

Recently, a model similar to the one in this paper has been analyzed in [20], where the authors use scheduling algorithms based on graph matchings (similar in spirit to the iLQF class of algorithms in this paper) and show delay-optimality in the case of two users, and provide heuristics when more users are present. To the best of our knowledge, the first paper to consider the small-buffer overflow problem in a large deviations setting was [1], where we presented a subset of the results in this paper. That paper focused mainly on the symmetric arrivals case, whereas this paper, in addition, analyzes a number of generalizations of the basic system model that were not considered in [1].

III. Motivation

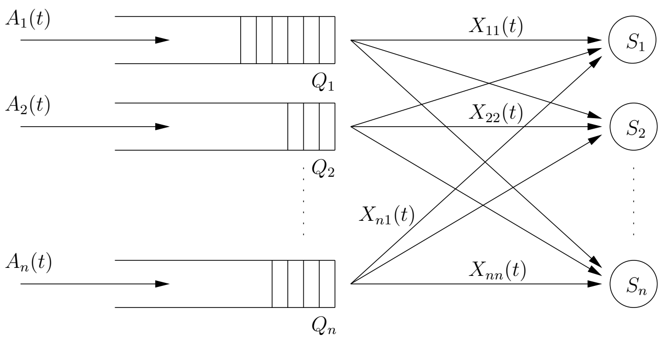

Fig. 1. System Model

We consider a discrete time queuing system with n queues and n servers as in Figure 1. Table I summarizes the notation used throughout this paper.

This system model can be used to study an OFDM downlink (such as WiMax) where each channel (sub-band), consisting of a fixed number of sub-carriers, is a server in Figure 1. There are a fixed number of mobile users, each represented by a queue that corresponds to the backlogged data at the base-station that is destined to the corresponding mobile user. The scheduler operates once every timeslot. During each timeslot, a channel can be assigned to one and at most one user (queue). The state of the channel ($X_{ij}(t)$) to a specific user depends on the location of the user.

Some typical rates (for a 20 MHz WiMax-like system) are as follows: the air-interface is based on OFDM with 50 channels (sub-bands), each of which consists of 25 sub-carriers. Each

Q_i	=	The entity, queue number i
S_i	=	The entity, server number i
$\mathcal{Q}$	=	$\{Q_1, Q_2, \ldots, Q_n\}$
$\mathcal{S}$	=	$\{S_1, S_2, \ldots, S_n\}$
$A_i(t)$	=	The number of packet arrivals to Q_i at the beginning of timeslot t
$X_{ij}(t)$	=	The number of packets in Q_i that can potentially be served by S_j, in timeslot t
$Q_i(t)$	=	The length of Q_i at the end of timeslot t
$Q_i^{(k)}(t)$	=	The length of Q_i after $k \geq 1$ rounds of service in timeslot t
$Q_i^{(0)}(t)$	=	$Q_i(t-1) + A_i(t)$, i.e. the length of Q_i after immediately after arrivals, in timeslot t
a^+	=	$\max(a, 0)$
$\Re^m$	=	The m-dimensional euclidean space
$H(x\|y)$	=	$x \log \frac{x}{y} + (1-x) \log \frac{1-x}{1-y}$

TABLE I
NOTATION

channel can support 400 kbps and the scheduler operates once every 5 milliseconds. Thus, each good (ON) channel offers 2 kb per timeslot.

Now the challenge is to develop a high-performance scheduling algorithm for this system. At first glance, by treating each server as a separate downlink server, the problem is not very different from the scheduling for a traditional downlink network. We can then use the following max-weight scheduling algorithm, which is throughput-optimal.

Definition 1 (MaxWeight scheduling, [4]). *In timeslot t, for $1 \leq j \leq n$, allocate server S_j to serve queue Q_i^* such that*

$$Q_i^* \in \arg\max_{Q_i} X_{ij}(t) Q_i(t),$$

breaking ties arbitrarily. ◇

While the max-weight scheduling is throughput-optimal, it results in large per-user delays (due to large queues at the base-station). For example, consider $Q_1(t) = 100, Q_2(t) = Q_3(t) = 95, Q_4(t) = Q_5(t) \ldots = Q_{100}(t) = 10$. Then, under the MaxWeight rule, all the servers S_j such that $X_{1j}(t) = 1$ are allocated to serve Q_1. Assume that $X_{ij}(t) = 1$ with probability 0.9, and $X_{ij}(t)$ are mutually independent. Then, roughly 90% of the servers (channels) are allocated to user '1', and the remaining 10% to users 2 and 3, which results in long queues for these two users at the end of the timeslot.

In fact, it can be argued that the MaxWeight algorithm "drives up" all the queue lengths to large enough values to ensure the maximum scheduling flexibility. The reason the MaxWeight algorithm is not the right choice for a scheduling algorithm for this system is that it potentially allocates all the available servers to serve *the* longest queue, essentially treating a slightly smaller queue as if it were empty. For a system with a large number of servers, this allocation policy leads to draining the longest queue(s) much more than is warranted by good load-balancing. It also leads the system getting "trapped" in a state where a significant fraction of queues is long, i.e. once such a state is reached (which happens infinitely often, almost surely, since the system is positive recurrent under the MaxWeight rule), then it is difficult to leave this state "quickly." Thus, we need to develop a scheduling algorithm that results in small per-user queues at the base station. We propose (in this paper) a class of algorithms called iLQF that serves this purpose.

IV. System Model

We consider a multi-channel wireless network as shown in Figure 1. The systems are indexed by the number of servers (and queues), n, are are denoted by Υ_n. (Note that for our proof techniques to work and results to hold, it is not necessary that the number of queues and servers be the same, and a constant factor relationship between the two works just as fine.) For concreteness, we assume that in a given timeslot, there are first arrivals to the queues (if any), then possible service, and the queue-lengths are measured at the end of the timeslot. The arrivals to the queues, and the channels connecting the queues to the servers are i.i.d. Bernoulli, independent across queues and time.[1] In particular,

$$A_i(t) = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1-p, \end{cases} \tag{1}$$

and

$$X_{ij}(t) = \begin{cases} 1 & \text{with probability } q, \\ 0 & \text{with probability } 1-q, \end{cases} \tag{2}$$

for some $p, q \in (0,1)$. All the random variables $A_i(t)$ and $X_{jk}(s)$ are mutually independent for all possible values of the involved parameters. Each queue maintains a buffer of infinite size, so that no packets are ever dropped. If $X_{ij}(t) = 1$, then the server S_j can potentially serve queue Q_i in timeslot t, reducing the length of Q_i by 1 (unless it is empty). Our aim is to define a service rule for allocating the servers to the queues this system, that meets certain performance metrics to be defined. The service rule is allowed to use the entire history of queue-lengths, arrivals, channel realizations, and server allocation decisions, as well as the queue-lengths, channel realizations and arrivals in the current timeslots, and any amount of external randomness (if necessary), and is required to define the following random variables for each timeslot t :

$$Y_{ij}(t) = \begin{cases} 1 & \text{if } S_j \text{ is allocated to serve } Q_i \text{ in timeslot } t, \\ 0 & \text{otherwise.} \end{cases}$$

As in an OFDM system, a server can serve at most one queue, but a queue may be served by multiple servers. That is, for all t and all $j \in \{1, 2, \ldots, n\}$, we require $\sum_{i=1}^n Y_{ij}(t) \leq 1$. The queue-lengths at the end of a timeslot are defined by the following equation:

$$Q_i(t) = \left(Q_i(t-1) + A_i(t) - \textstyle\sum_{j=1}^n X_{ij}(t) Y_{ij}(t) \right)^+ .$$

The service model is as shown in Figure 2.

A finite integer $b \geq 0$ is fixed. The queuing system is started at time $-\infty$. Our objective is to design a service rule that maximizes

$$\liminf_{n \to \infty} \frac{-1}{n} \log \mathbb{P} \left(\max_{1 \leq i \leq n} Q_i(0) > b \right). \tag{3}$$

The above expression is called a rate-function in the large deviations theory, and thus our goal is to design a service

[1]We adopt such a system model for ease of exposition. Significant generalization of this model is possible, as noted in Section IX.

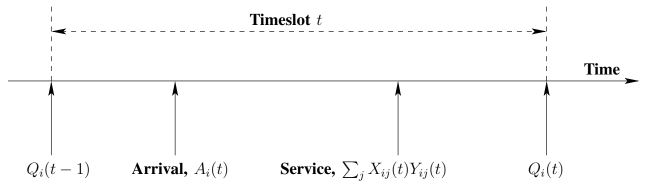

Fig. 2. Service Model

rule that is rate-function optimal. We refer to the event $\{\max_i Q_i(t) > b\}$ as the small buffer overflow event, or simply the *overflow event*. We consider only ergodic service policies that make all the queues in the system positive recurrent, so that the probability in (3) is well defined, and equals the fraction of timeslots for which $\{\max_i Q_i(t) > b\}$. Roughly, for large values of n and any fixed b, (3) is equivalent to designing a scheduling policy that results in the *largest* value of $\alpha(b)$, where

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(0) > b\right) \approx e^{-n\alpha(b)}$$

This means that (for large systems) the algorithm with such a property will result in the smallest buffer overflow probability, for any buffer size b. It is therefore desirable to have as large a value of the rate function as possible.

V. Algorithm-Independent Lower Bound on Overflow Probability

In this section, we present a lower bound on the overflow probability (3). This is an algorithm-independent lower bound, so it holds for any scheduling algorithm. In Section VII, we develop a class of iterative algorithms (iLQF) that achieve this bound.

Theorem 1. *For the system Υ_n, under any rule for allocating servers to queues, and for all possible values of the parameters $n > 0, 0 < p, q < 1, b \ge 0$,*

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(0) > b\right) \ge p^{b+1}(1-q)^{n(b+1)}.$$

Consequently, for any $p > 0$,

$$\limsup_{n\to\infty} \frac{-1}{n}\log\mathbb{P}\left(\max_{1\le i\le n} Q_i(0) > b\right) \le (b+1)\log\frac{1}{1-q}. \tag{4}$$

Proof: Consider the following event which implies that $\{Q_1(0) > b\}$: for $b+1$ consecutive timeslots before (and including) timeslot 0, there are arrivals to Q_1, and all the channels connecting Q_1 to the servers are OFF in each of the $b+1$ timeslots. The probability of this event is equal to $p^{b+1}((1-q)^n)^{b+1}$, and the result follows. ■

VI. Stability and Perfect Matchings

In this section, we first show that the system under consideration is stabilizable under some scheduling rule, for n large. That is, there exists a scheduling algorithm that makes the queue-length Markov chain positive recurrent.

Theorem 2. *For given values of $p, q \in (0,1)$, there exists $n_0 = n_0(p,q)$ such that for all $n \ge n_0$, the queuing system Υ_n can be stabilized by some service rule.*

Proof: Please see Appendix A. ■

Let $\lambda_i^{(n)}$ denote the arrival rate (expected number of arrivals) to queue Q_i, in the system Υ_n. Assuming that

$$\limsup_{n\to\infty}\max_{1\le i\le n}\lambda_i^{(n)} \in (0,1),$$

the above stability result can be generalized to the following cases:

1) Bernoulli arrivals to the queues, with arrival rates to the different queues being different.
2) The number of arrivals to a queue in a given timeslot takes on values in a finite, non-negative integer set.

We analyze the two cases mentioned above in Section IX. Next we prove a result regarding perfect matchings in bipartite graphs that is useful for the analysis of the proposed algorithm later in the paper.

Lemma 1. *Consider an undirected bipartite graph $G(\mathcal{U}\cup\mathcal{V},\mathcal{E})$, where $\mathcal{U}\cup\mathcal{V}$ is the set of vertices with $|\mathcal{U}| = |\mathcal{V}| = n$, and $\mathcal{E}$ is the set of edges. Every edge $e\in\mathcal{E}$ has one of its endpoints in $\mathcal{U}$ and the other in $\mathcal{V}$. For every node $u\in\mathcal{U}$ and $v\in\mathcal{V}$, the edge (u,v) is present in $\mathcal{E}$ with probability q, independently of all other edges. Then, for large n,*

$$(1-q)^n \le \mathbb{P}(G \text{ has no perfect matching}) \le 3n(1-q)^n,$$

where a perfect matching is defined as a matching of cardinality n.

Proof: Please see Appendix B. ■

Qualitatively, this result shows that the large bipartite graphs as described here have perfect matchings with very high probability.

VII. Characteristics of Optimal Service Rules

In this section, we consider a special class of service rules - iLQF (iterated Longest Queues First), and present sufficient conditions for an iLQF scheduling policy to be rate-function optimal. In the next section, we present an algorithm in this class that maximizes (3).

Definition 2 (iterated Longest Queues First (iLQF))**.** *A scheduling rule Λ is said to belong to the iLQF class if, in every timeslot t, it allocates servers to queues in multiple rounds of allocations as follows:*

1) *In a given round, the scheduling rule Λ finds a largest cardinality matching in the bipartite graph whose node-sets are the set of longest queues and set of available servers, and the edges are defined by the channel realizations (an edge from Q_i to S_j is present if $X_{ij}(t) = 1$). If the cardinality of the matching thus found equals the cardinality of the set of the (current) longest queues, then the rule is required assign at least one server to each of the (current) longest queues (for example, by assigning servers to the queues as dictated by the matching, or by any other means).*
2) *The service rule Λ updates the lengths of all the queues (to take into account the service received by a subset of*

the longest queues in the particular round) and the set of available servers (to remove from further consideration the servers allocated to some of the queues) and proceeds to the next round. ◇

Note that the class iLQF contains more than one scheduling algorithm, since the following parameters are unspecified:

1) The tie-breaking rule if in a round, there exist multiple largest cardinality matchings among the (current) longest queues and available servers.
2) The number of rounds to be performed, i.e. the termination condition. In particular, if in a given round not all the (current) longest queues can be allocated a server, then in that round, the rule Λ can arbitrarily allocate the servers to the queues, or terminate and proceed to the next timeslot.

An iLQF-class algorithm can terminate (for the given timeslot) as soon as there happens to be a round where at least one of the (current) longest queues cannot be allocated at least one server. This is potentially wasteful of resources, and other considerations (such as throughput-optimality under more general arrival and channel processes) would demand a more judicious utilization of these "remaining" resources. However, we allow such "wasteful" allocation schemes because: (a) they are easier to analyze, (b) they provide lower bounds on the performance of the non-wasteful algorithms (this can be formally demonstrated in the case of the proposed iLQF with PullUp algorithm (Definition 6), and is a consequence of its dominance property (Lemma 6)), (c) they can still be rate-function optimal (as demonstrated later for the case of iLQF with PullUp), which is the objective of this paper, and (d) it helps limit the complexity of implementation.

This class of algorithms is interesting because it gives priority to the longer queues, thereby trying to minimize the probability of the overflow event.

Lemma 2 (Forward jump bound)**.** *For any algorithm in the iLQF class, and for n large enough,*

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) > \max_{1\le i\le n} Q_i(t)\right) \le 3n(1-q)^n.$$

Proof: Consider the bipartite graph $G(\mathcal{Q}\cup\mathcal{S},\mathcal{E})$, where $\mathcal{E} := \{X_{ij}(t) : X_{ij}(t) = 1\}$. If G has a perfect matching (i.e., a matching of cardinality n), then for an algorithm in the iLQF class, $\max_{1\le i\le n} Q_i(t+1) \le \max_{1\le i\le n} Q_i(t)$. Further, by Theorem 1, the graph G has a perfect matching with probability at least $1-3n(1-q)^n$ for large n. ■

Definition 3 (Dominance property of an iLQF rule Λ)**.** *Consider the queuing system with $\mathcal{Q} = \{Q_i\}_{i=1}^n$ as the queues, and $\mathcal{S} = \{S_i\}_{i=1}^n$ as the servers. Let $A_i(t)$ and $X_{ij}(t)$ be the arrival process and channel processes respectively (see Equations* (1), (2)*). Now, a new queuing system with queues $\mathcal{R} = \{R_i\}_{i=1}^n$ and servers $\mathcal{S} = \{S_i\}_{i=1}^n$ is obtained as follows: at each time t, the queues $R_i(t), i = 1,2,\ldots,n$ see the same arrivals as those incoming to $Q_i(t), i = 1,2,\ldots,n$ and the channel states of the servers are identical to those of system $\mathcal{Q}$ (i.e., the arrival processes and channel states in the system $\mathcal{R}$ are sample-path coupled with the system $\mathcal{Q}$).* In addition, *there are extra packet arrivals (an arbitrary, finite number) that occur to an arbitrary subset of queues in the system $\mathcal{R}$ immediately after service, and at arbitrary timeslots $T_1, T_2, \ldots$ (see Figure 3). The service policy used in the queuing system $\mathcal{R}$ is the same iLQF policy (Λ) that is used in the system $\mathcal{Q}$ (also the process $\mathcal{R}$ is defined over the same probability space as $\mathcal{Q}$).*

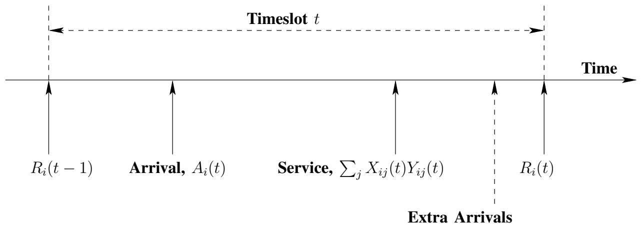

Fig. 3. Service model for the queuing system $\mathcal{R}$

A rule Λ in the iLQF class is said to have the dominance property if the following holds: for all timeslots t, all $b \ge 0$, and over all possible choices for extra arrivals, we have that

$$\mathbb{P}\left(\max_{1\le i\le n} R_i(t) > b\right) \ge \mathbb{P}\left(\max_{1\le i\le n} Q_i(t) > b\right). \qquad \diamond$$

Intuitively, the dominance property requires that *adding extra packets* to the queuing system driven by the iLQF policy Λ does *not decrease* the maximum queue length. In other words, the system under the rule Λ does not "benefit" from an addition of packets. This property is extremely useful because it allows us to "carefully" add packets so that the resulting queuing system can be explicitly analyzed and whose rate function can be computed in closed-form. More precisely, we can derive bounds on the transition probabilities between different states of the queue-length Markov chain, and construct a dummy Markov chain (by adding extra packets to the original Markov chain) for which the transition probability bounds hold with equality, and the dummy Markov chain's rate-function is analyzable in closed-form.

Definition 4 (Drain property of a scheduling rule Λ)**.** *A scheduling rule Λ (not necessarily from the iLQF-class) is said to have the drain property if there exists a constant k_0 independent of n such that*

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+k_0) < \max_{1\le i\le n} Q_i(t) \,\middle|\, \max_{1\le i\le n} Q_i(t) > 0\right) \ge \frac{1}{2},$$

for all n large enough and all integers (timeslots) t. ◇

Theorem 3. *Suppose that a service rule in the iLQF-class has the drain and dominance properties. Then, this iLQF-class service rule results in*

$$\liminf_{n\to\infty} \frac{-1}{n}\log\mathbb{P}\left(\max_{1\le i\le n} Q_i(0) > b\right) = (b+1)\log\frac{1}{1-q}.$$

Further, by Theorem 1, no other service rule can give a larger value for the left hand side of the above expression.

Proof: Please see Appendix C. ■

Thus, the *drain* and *dominance* properties are two of the desired properties of an iLQF-class rule. It turns out that the drain property is typically easier to guarantee than the dominance property, and a significant part of the next Section is devoted to analyzing a particular algorithm in the iLQF-class and proving that it has the dominance property.

VIII. A Specific Algorithm

We now focus our attention on constructing an algorithm in the iLQF class that satisfies the requirements in the statement of Theorem 3. The algorithm employs a particular tie-breaking rule (PullUp) when there exist multiple largest-cardinality matchings in the bipartite graph defined by the set of longest queues and unallocated servers, where the edges are defined by the ON links. This tie-breaking rule ensures the following sample-path property: if a queuing system $\underline{R}$ is obtained by adding an arbitrary number of extra (dummy) packets to some or all the queues in a queuing system $\underline{Q}$ at the end of some timeslot t, and the two queuing systems $\underline{Q}$ and $\underline{R}$ see the same arrivals and channel realizations for all future timeslots, then $Q_i(s) \leq R_i(s)$ for all $1 \leq i \leq n$, and all $s \geq t$.

Definition 5 (PullUp)**.** *Consider a bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$, where the sets of nodes, not necessarily of the same cardinality, are $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ and $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$. Given a matching $\mathcal{M}$ in G, $\mathcal{M}' :=$ PullUp$(G, \mathcal{M}, \mathcal{V})$ is a new matching obtained by the following steps, which we call PullUp:*

1) *Mark all the edges in $\mathcal{M}$ as forward edges (i.e. from $\mathcal{U}$ to $\mathcal{V}$), and all the other edges in $\mathcal{E}$ as backward edges, to get a directed graph G_1. Define $\mathcal{M}_1 := \mathcal{M}$. Initialize $k = 1$.*
2) *Obtain $\mathcal{M}_{k+1}$ from $\mathcal{M}_k$ as follows: If the node v_k has an incoming edge, then define $\mathcal{M}_{k+1} := \mathcal{M}_k$, $G_{k+1} := G_k$. Otherwise, in the directed graph G_k, find the set $\mathcal{N}_k$ of all nodes reachable from v_k. Let $\Gamma(G_k, v_k) := \mathcal{N}_k \cap \mathcal{U}$ and $\Delta(G_k, v_k) := \mathcal{N}_k \cap \mathcal{V}$. Find the smallest index $l > k$ such that $v_l \in \Delta(G_k, v_k)$. If no such l exists, then define $\mathcal{M}_{k+1} := \mathcal{M}_k$ and $G_{k+1} := G_k$. If such an l exists, then reverse the directions of all the edges on a path from v_k to v_l, to obtain a graph G_{k+1}. Define $\mathcal{M}_{k+1}$ to be the set of all forward edges in G_{k+1}.*
3) *Increment k by 1. If $k = n+1$, then return the matching $\mathcal{M}' := \mathcal{M}_{n+1}$, else go to step 2.* ◇

An example of the PullUp operation is shown in Figure 4. In the next lemma, we prove that the output of the PullUp is also a matching.

Lemma 3. *The output $\mathcal{M}'$ of PullUp$(G, \mathcal{M}, \mathcal{V})$ is a matching, and $|\mathcal{M}| = |\mathcal{M}'|$.*

Proof: Please see Appendix D. ■

The objective of the PullUp operation is to efficiently find a "good" matching. Based on the PullUp technique, we construct an iLQF-class algorithm that is rate-function optimal for the small buffer overflow event under consideration. To avoid trivialities, we define the algorithm for the case when at least one of the queues is nonempty.

Definition 6 (iLQF with PullUp)**.**
Input:

1) *The queue lengths, $Q_1(t-1), Q_2(t-1), \ldots, Q_n(t-1)$.*
2) *The channel realizations, $X_{ij}(t)$ for $1 \leq i, j \leq n$.*
3) *The arrivals to the queues, $A_i(t)$ for $1 \leq i \leq n$.*

Steps:

1) *Update the queue-lengths to account for arrivals, that is, compute the new length of queue Q_i after arrivals, $Q_i^{(0)}(t) := Q_i(t-1) + A_i(t)$. Hereafter, the length of a queue always refers to its most current updated length, accounting for arrivals and service. Find the length of the longest queue, $\hat{Q}$. Define $L = \hat{Q}$. Initialize $r = 0$. Let $\mathcal{S}^\star$ denote the set of unallocated servers. To begin with, we have $\mathcal{S}^\star = \mathcal{S}$.*
2) *Let $\mathcal{Q}_L$ denote the set of queues whose length (i.e., $Q_i^{(r)}(t)$) is exactly L. Let G_L denote the (undirected) bipartite graph with nodes $\mathcal{Q}_L \cup \mathcal{S}^\star$, and the edges as defined by the channel realizations. More specifically, an edge (Q_i, S_j) is present in G_L if $Q_i \in \mathcal{Q}_L, S_j \in \mathcal{S}^\star$ and $X_{ij}(t) = 1$. Find a largest cardinality matching $\mathcal{M}_L$ in the graph G_L.*
   a) *If $|\mathcal{M}_L| = |\mathcal{Q}_L|$, then define*
   $\mathcal{M}' :=$ *PullUp*$(G_L, \mathcal{M}_L, \mathcal{S}^\star)$.
   b) *If $|\mathcal{M}_L| < |\mathcal{Q}_L|$, then define*
   $\mathcal{M}_1 :=$ *PullUp*$(G_L, \mathcal{M}_L, \mathcal{S}^\star)$.
   Obtain $\mathcal{M}_{k+1}$ from $\mathcal{M}_k$ as follows: if k is odd, then define $\mathcal{T} := \mathcal{Q}_L$; otherwise $\mathcal{T} := \mathcal{S}^\star$, and $\mathcal{M}_{k+1} :=$ PullUp$(G_L, \mathcal{M}_k, \mathcal{T})$. Continue obtaining $\mathcal{M}_{k+1}$ from $\mathcal{M}_k$ until $\mathcal{M}_{i+1} = \mathcal{M}_i$ for some i. Define $\mathcal{M}' := \mathcal{M}_i$.

   Finally, as defined by the matching $\mathcal{M}'$, allocate the servers to queues. For example, if $(Q_x, S_y) \in \mathcal{M}'$, then define $Y_{xy}(t) = 1$, allocate S_y to serve Q_x, remove S_y from $\mathcal{S}^\star$, decrease the length Q_x by 1, i.e., define $Q_x^{(r+1)}(t) := Q_x^{(r)}(t) - 1$. For a node (queue) Q_z that is not an endpoint of any edge in $\mathcal{M}'$, define $Q_z^{(r+1)}(t) := Q_z^{(r)}(t)$.
3) *If at the end of step 2, we have $|\mathcal{M}_L| < |\mathcal{Q}_L|$, then stop. If $|\mathcal{S}^\star| = 0$ or $L = 1$, then stop. Else, decrease the value of L by 1, increment r by 1, go to step 2.*

Output:

1) *The allocation decisions, $Y_{ij}(t)$ for $1 \leq i, j \leq n$.*
2) *The final queue-lengths, $Q_i(t) := Q_i^{(r+1)}(t)$. (Here, the value of r refers to its value at the end of step 3.)* ◇

Here is a description of the algorithm in words: in every timeslot, the algorithm proceeds in multiple rounds of service. In every round, the algorithm finds a largest-cardinality matching $\mathcal{M}$ in the (bipartite) graph defined by the longest queues and the unallocated servers, where the edges in the graph are defined by the channel realizations. The algorithm then applies the PullUp operation (once or multiple times) to the matching $\mathcal{M}$ and obtains a matching $\mathcal{M}'$. It allocates servers to the queues as defined by the edges in the matching $\mathcal{M}'$, updates (decreases by one) the lengths of the served queues, removes the allocated servers from the set of available servers, and proceeds to the next round.

Let every execution of step 2 be called a round. If in the step 2 we have $|\mathcal{M}_L| = |\mathcal{Q}_L|$, then that round is called a perfect queue matching round, else a maximal matching round.

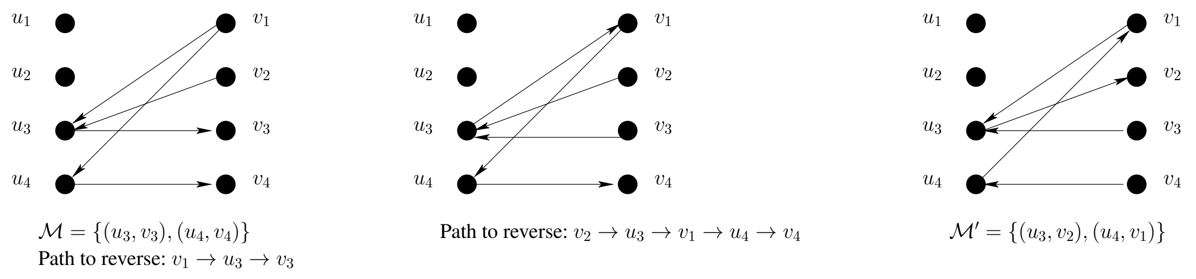
u_1
u_2
u_3
u_4
v_1
v_2
v_3
v_4
$\mathcal{M} = \{(u_3, v_3), (u_4, v_4)\}$
Path to reverse: $v_1 \to u_3 \to v_3$
Path to reverse: $v_2 \to u_3 \to v_1 \to u_4 \to v_4$
$\mathcal{M}' = \{(u_3, v_2), (u_4, v_1)\}$

Fig. 4. An example of the PullUp operation

Theorem 4. *The iLQF with PullUp is rate-function optimal, i.e., it results in*

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i(0) > b\right) = (b+1)\log\frac{1}{1-q}.$$

Further, the algorithm can be implemented in $O(n^4)$ computations per timeslot.

Proof: We prove that the iLQF with PullUp satisfies the drain property (Lemma 8) and the dominance property (Lemma 6). Thus, the first part of the claim holds according to Theorem 3. The second part of the claim (the computational complexity result) follows from Lemma 4. ■

A. Computational Complexity

We first analyze the computational complexity of the iLQF with PullUp.

Lemma 4. *The proposed algorithm (iLQF with PullUp) can be implemented in $O(n^4)$ computations per timeslot.*

Proof: Please see Appendix E. ■

B. Rate-function Optimality

We establish the rate-function optimality of the iLQF with PullUp by proving that the algorithm has the drain property and the dominance property as required by Theorem 3. The following is a technical lemma that is useful in the proof of Lemma 6.

Lemma 5 (No edge, no path)**.** *Under the notation of Definition 5, in the graph G_{n+1}, if a node v_a has no incoming edge, then there does not exist a (directed) path from v_a to any node v_b with $b > a$. Consequently, if PullUp$(G, \mathcal{M}, \mathcal{V}) = \mathcal{M}'$, then PullUp$(G, \mathcal{M}', \mathcal{V}) = \mathcal{M}'$.*

Proof: Please see Appendix F. ■

Lemma 6. *[Sample-path-wise Dominance] Consider two queuing systems $\underline{Q}$ and $\underline{R}$ with queues $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_n\}$ and $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ respectively, with the property that $Q_i(t-1) \le R_i(t-1)$ for all i. Let the two systems have identical channel realizations, $X_{ij}(t)$ and identical arrivals, $A_i(t)$ for $1 \le i, j \le n$. Both the queuing systems implement the algorithm described in Section VIII, i.e. iLQF with PullUp. Then, $Q_i(t) \le R_i(t)$ for all i.*

Proof: Please see Appendix G. ■

Note that this theorem immediately implies that the iLQF with PullUp algorithm has the dominance property as required by Theorem 3.

Corollary 1. *The iLQF with PullUp algorithm has the dominance property defined in Section VII.*

The corollary follows by repeated applications of Lemma 6. The queuing system is started at time $-\infty$, and we are interested in the probability that the length of the longest queue exceeds a constant b at a finite time t. By applying the result of Lemma 6 to timeslots $T_1, T_2, \ldots$ (in the definition of the Dominance property), it follows that the packet-added system has sample-path wise longer queues than the original system. The probabilistic dominance is an immediate consequence of this sample-path dominance.

We now demonstrate a property of the PullUp operation which is useful in proving that the proposed algorithm has the Drain property as required by Theorem 3.

Lemma 7 (Use smaller indexed nodes)**.** *Let a bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$ and a matching $\mathcal{M}$ be given, with*

$$\mathcal{U} = \{u_1, u_2, \ldots, u_n\}, \mathcal{V} = \{v_1, v_2, \ldots, v_n\}.$$

Suppose there exists a matching $\mathcal{M}_\star$ in G with the following properties:

1) $|\mathcal{M}| = |\mathcal{M}_\star|$.
2) *If $u \in \mathcal{U}$ is an endpoint of some edge $e \in \mathcal{M}$, then u is an endpoint of some edge $e' \in \mathcal{M}_\star$.*
3) *Mark all the edges in $\mathcal{M}_\star$ as forward edges (i.e., from $\mathcal{U}$ to $\mathcal{V}$), and all the edges in $\mathcal{E} \backslash \mathcal{M}_\star$ as backward edges, to get a directed graph $G^\ddagger(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$. Then, in the graph $G^\ddagger$, if a node $v_i \in \mathcal{V}$ has no incoming edge, then there does not exist a directed path from v_i to any $v_j,\ j > i$.*
4) *There exists an index $a \le n$ such that for every $b > a$, no node $v_b \in \mathcal{V}$ is an endpoint of any edge in $\mathcal{M}_\star$.*

Let $\mathcal{M}' = PullUp(G, \mathcal{M}, \mathcal{V})$. Then, any edge in $\mathcal{M}'$ does not have, as an endpoint, any node in $\mathcal{V}$ with index larger than a.

Proof: Please refer to Appendix H. ■

Let $\hat{Q}(T)$ denote the length of the longest queue at the end of the timeslot T. We next prove that the iLQF with PullUp satisfies the drain property.

Lemma 8. *(The Drain property) For the proposed algorithm, there exists a constant $k = k(p) = \left\lceil \frac{3}{1-p} \right\rceil$ such that, for all n large enough, all $m > 0$ and all T,*

$$\mathbb{P}(\hat{Q}(T+k) < m | \hat{Q}(T) = m) \ge \frac{1}{2}.$$

Proof: Please refer to Lemma 10, which proves a more general claim. Substituting $L = 1$ in that lemma gives the desired result. ■

This completes our analysis of the iLQF with PullUp algorithm for the basic system model, Υ_n.

IX. Generalizing the System Model

In this section, we consider a number of natural extensions of the system model Υ_n defined in Section IV, and analyze the performance of the proposed iLQF with PullUp algorithm for them.

A. The Asymmetric Arrivals Case

Consider a queuing system Υ'_n that is a modification of the system Υ_n. Let the queues, servers, arrivals and channels for the system Υ'_n be indexed by n and denoted by $Q_i^{[n]}, S_j^{[n]}, A_i^{[n]}(t)$ and $X_{ij}^{[n]}(t)$ respectively. Let

$$A_i^{[n]}(t) = \begin{cases} 1 & \text{with probability } p_i^{(n)}, \\ 0 & \text{with probability } 1 - p_i^{(n)}, \end{cases}$$

and

$$X_{ij}^{[n]}(t) = \begin{cases} 1 & \text{with probability } q, \\ 0 & \text{with probability } 1 - q. \end{cases}$$

In particular, the number of packets arriving to the i^{th} queue in timeslot t in the system Υ'_n is a Bernoulli random variable whose parameter is arbitrarily fixed in $(0,1)$. For stability, we impose the condition

$$\limsup_{n\to\infty} \max_{1\le i\le n} p_i^{(n)} = \alpha \in (0,1). \tag{5}$$

Under this condition, following an argument similar to that in the proof of Theorem 2, the system Υ'_n is stable for all n large enough. We refer to this system as a system with asymmetric arrivals.

Theorem 5. *For any given $\epsilon \in (0,\alpha)$, there exists a constant $n_0 = n_0(\epsilon)$ such that under any rule for allocating servers to queues, and for all possible values of the parameters $0 < \alpha, q < 1, b \ge 0$, and for infinitely many $n \ge n_0$,*

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge (\alpha - \epsilon)^{b+1}(1-q)^{n(b+1)}.$$

Consequently,

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \le (b+1)\log\frac{1}{1-q}.$$

Proof: Please see Appendix I. ■

The next claim establishes that the proposed iLQF with PullUp algorithm results in a matching lower bound on the rate function for the system with asymmetric arrivals, and is therefore rate function optimal for the small buffer overflow event for this system.

Theorem 6. *For the system with asymmetric arrivals, the iLQF with PullUp algorithm has the property*

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge (b+1)\log\frac{1}{1-q}.$$

Proof: Please see Appendix J. ■

As a result of Theorems 5 and 6, the proposed algorithm (iLQF with PullUp) is rate-function optimal for the system with asymmetric arrivals.

B. Symmetric, Bursty, ON-OFF Arrivals

Let the arrival process for the system Υ'_n be given by

$$A_i^{[n]}(t) = \begin{cases} L & \text{with probability } p, \\ 0 & \text{with probability } 1 - p, \end{cases}$$

for some fixed constants p and L, with $pL \in (0,1)$. We refer to this system as a system with symmetric, bursty, ON-OFF arrivals. Note that the channel process of this system is exactly as that of the system Υ_n defined in Section IV.

Hereafter in this section, for ease of notation, we drop the explicit dependence of the variables on n. As before, we let $\hat{Q}(t) := \max_{1\le i\le n} Q_i(t)$.

Theorem 7. *For a system with symmetric, bursty, ON-OFF arrivals implementing any algorithm for assigning servers to queues, and for all $b \ge 0$,*

$$\limsup_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \le \left\lceil \frac{b+1}{L} \right\rceil \log\frac{1}{1-q}.$$

Proof: The proof is similar to that of Theorem 1, and presented in Appendix K. ■

Lemma 9. *Fix any $\tilde{p} \in (p, 1/L)$. Define $\Theta_1 := nH(\tilde{p}|p)$ and $\Theta_2 := 3n\tilde{p}(1-q)^{n\tilde{p}}$. Then for the symmetric, bursty, ON-OFF arrivals system, for n large enough, and for all t,*

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) > \max_{1\le i\le n} Q_i(t)\right) \le \exp(-\Theta_1) + L\Theta_2.$$

Proof: Please see Appendix L. ■

Lemma 10. *For the symmetric, bursty, ON-OFF arrivals system implementing the iLQF with PullUp algorithm, there exists a constant $k = k(L,p) = \left\lceil \frac{3}{1-pL} \right\rceil$ such that for all n large enough, for all $m > 0$ and all T,*

$$\mathbb{P}(\hat{Q}(T+k) < m | \hat{Q}(T) = m) \ge \tfrac{1}{2}.$$

Proof: Please see Appendix M. ■

Theorem 8. *For a system with symmetric, bursty, ON-OFF arrivals implementing the iLQF with PullUp algorithm, for all $b \ge 0$, and for $\tilde{p} = (1/L + p)/2$,*

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge \left\lceil \frac{b+1}{L} \right\rceil \min\left(\tilde{p}\log\frac{1}{1-q}, H(\tilde{p}|p)\right) > 0.$$

Proof: Please see Appendix N. ■

C. Symmetric Arrivals with Bounded Support

Let the arrival process for the system Υ'_n be given by

$$A_i^{[n]}(t) = \begin{cases} 0 & \text{with probability } p_0, \\ 1 & \text{with probability } p_1, \\ \vdots & \vdots \\ L & \text{with probability } p_L, \end{cases} \tag{6}$$

and $\mathbb{P}(A_i^{[n]}(t) > L) = 0$. We require that $p_i \ge 0$ for all i, $p_L > 0$, $\sum_{i=0}^{L} p_i = 1$ and $\sum_{i=0}^{L} ip_i \in (0,1)$ for stability for n large. We refer to this system as a system with symmetric arrivals with bounded support.

Notation:
Let $r = [r_0, r_1, \dots, r_L]$ be a probability vector, that is, $r_i \geq 0$ for all i and $\sum_i r_i = 1$. Let $M_1(\Sigma)$ denote the probability simplex in $\Re^{L+1}$, that is, the set of all probability vectors in $\Re^{L+1}$.

Let $\lambda_p := \sum_i i p_i < 1$ denote the expected number of arrivals to a queue in the system. Define

$$\begin{aligned} F_\epsilon \quad := \quad & \Big\{ r \in M_1(\Sigma), p_k = 0 \Rightarrow r_k = 0, \\ & \sum_{k=0}^{L} k r_k \leq \frac{1+\lambda_p}{2}, \max_{0 \leq k \leq L} |r_k - p_k| \leq \epsilon \Big\}. \end{aligned}$$

For all $\epsilon > 0$, the set F_ϵ is nonempty ($\because p \in F_\epsilon$) and compact. There exists $\epsilon_0 \in (0, p_L)$ such that for all $\epsilon \leq \epsilon_0$, the complement of the set F_ϵ with respect to $M_1(\Sigma)$ is contained in the closure of its interior w.r.t. $M_1(\Sigma)$. Since $H(r|p) = 0$ if and only if $r = p$, and using results from [21], Section 2.1.1, it follows that

$$\zeta := \inf_{s \in M_1(\Sigma) \setminus F_{\epsilon_0}} H(s|p) > 0.$$

Define $J := \min_{d \in F_{\epsilon_0}} d_L$, where d_L denotes the L^{th} co-ordinate of $d = [d_0, d_1, \dots, d_L]$.

Theorem 9. *For the system with symmetric arrivals with bounded support, under the iLQF with PullUp algorithm, for all $b \geq 0$,*

$$\begin{aligned} \liminf_{n \to \infty} \frac{-1}{n} \log \mathbb{P} \left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b \right) \\ \geq \left\lceil \frac{b+1}{L} \right\rceil \min \left(J \log \frac{1}{1-q}, \zeta \right) > 0. \end{aligned}$$

Proof: Please see Appendix O. ■

Theorem 10. *For a system with symmetric arrivals with bounded support, implementing any algorithm for assigning servers to queues, and for all $b \geq 0$,*

$$\limsup_{n \to \infty} \frac{-1}{n} \log \mathbb{P} \left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b \right) \leq \left\lceil \frac{b+1}{L} \right\rceil \log \frac{1}{1-q}.$$

Proof: Similar to that of Theorem 7. ■

D. Asymmetric Arrivals with Bounded Support

Let the arrival process for the system Υ'_n be given by

$$A_i^{[n]}(t) = \begin{cases} 0 & \text{with probability } p_i^{(n)}(0), \\ 1 & \text{with probability } p_i^{(n)}(1), \\ \vdots & \vdots \\ L & \text{with probability } p_i^{(n)}(L), \end{cases} \tag{7}$$

and $\mathbb{P}(A_i^{[n]}(t) > L) = 0$. We require $p_i^{(n)}(j) \geq 0$ for all i, j, $\sum_{j=0}^{L} p_i^{(n)}(j) = 1$ for all i and (for stability for n large)

$$\limsup_{n \to \infty} \max_{1 \leq i \leq n} \sum_{j=0}^{L} j p_i^{(n)}(j) \in (0, 1).$$

Theorem 11. *For the system under consideration, under the iLQF with pullup algorithm, for all $b \geq 0$,*

$$\liminf_{n \to \infty} \frac{-1}{n} \log \mathbb{P} \left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b \right) > 0.$$

Proof: Similar to that of Theorems 6 and 9. ■

Thus, we see that the proposed iLQF with PullUp algorithm is robust to changes in the system model, and results in a strictly positive value of the rate function for the small buffer overflow event under a number of scenarios.

X. Simulation Results

In this section, we compare the performance of the proposed iLQF-class algorithms with the standard MaxWeight (MW) algorithm [4] under a number of conditions. We consider a system with $n = 20$ queues and 20 servers, with the channel between a queue and a server being ON with probability $q = 0.5$. We run the simulations for 5×10^5 timeslots, based on which the empirical probabilities that the maximum queue-length exceeds a constant b are computed.

In the first set of simulations (Figure 5), we run the two algorithms for the system Υ_n described in Section IV, with $\{0, 1\}$ i.i.d. arrivals, with different values of the probability of arrival (p).

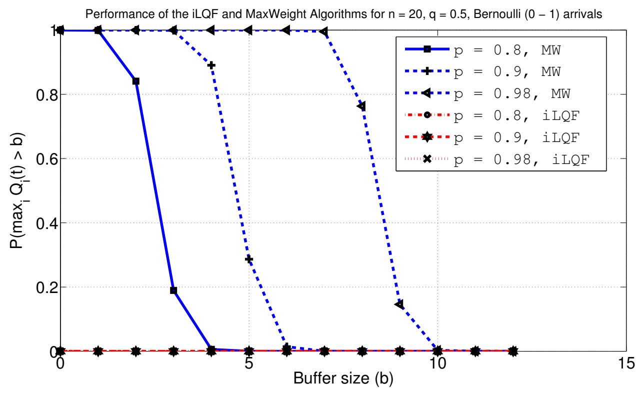

Fig. 5. Arrivals as per the system model, Υ_n

In the second set of simulations (Figure 6), we study the performance of the algorithms under bursty arrivals – in a given timeslot, every queue sees either 0 or 4 arrivals. The probability of arrival is adjusted so that the system is stable but heavily loaded (96% for $p = 0.24$). The results are summarized in Figure 6.

In the third set of simulations (Figure 7), we consider a system with asymmetric arrival rates to the queues. The arrivals to queues Q_{11}, Q_{15}, Q_{19} are modeled by a uniform random variable in $[0, 2L]$, while the rest of the queues see Bernoulli packet arrivals with $p = 0.12$. For $L = 5$, the system is at about 85.7% of the maximum stable load.

The results are summarized in the accompanying plots. As can be seen, the proposed iLQF with PullUp algorithm performs consistently better than the MaxWeight algorithm as far as the finite buffer overflow probabilities are concerned. The intuition for this is due to the reasoning that iLQF

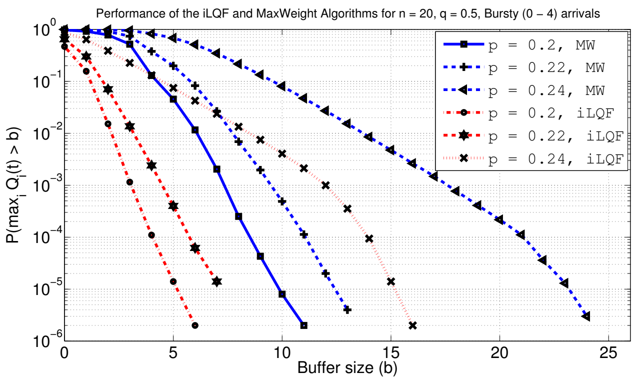

Fig. 6. Bursty arrivals

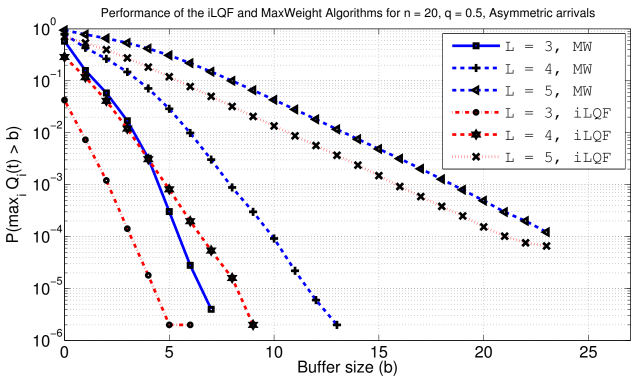

Fig. 7. Asymmetric arrivals

balances the servers among the long-queues, whereas the traditional MaxWeight algorithm focuses on a single longest-queue. When the buffers are large, this does not affect stability. However, for small buffer performance, there is a definite improvement as seen from the plots.

XI. Conclusions and Future Work

We considered the problem of designing scheduling algorithms for multi-user, multi-channel (e.g., OFDM-based) wireless downlink networks. Our aim was to design an algorithm that guarantees small per-user delay, which in turn is closely related to the per-user queue-lengths at the base-station. We formulated this problem as a rate-function maximization problem in a large deviations setting. We proposed a class of algorithms called iLQF (iterated Longest Queues First) as a solution to this problem. An iLQF-class scheduling rule proceeds by repeatedly finding a maximum cardinality allocation of servers to serve the longest queues, updating the queue-lengths to account of service, and proceeding to the next round *without back-tracking*. We showed that the iLQF-class algorithms, under certain mild technical conditions (namely, the *drain* and *dominance* properties) are rate-function optimal for the problem. We further showed that an algorithm in this class, namely iLQF with PullUp, satisfies the two properties and is therefore rate-function optimal for the problem.

This algorithm does not need to know or learn the arrival or channel process statistics, or the history of past decisions, and can be implemented with the knowledge of the current queue-lengths and channel realizations. Its computational complexity is $O(n^4)$ computations per timeslot. We investigated the performance of the iLQF with PullUp algorithm under a variety of modifications to the basic system model, and showed that it results in a strictly positive rate function under those changes. This implies a good small-queues performance, and a high quality of service for all the users. Further, through simulations, we compared the performance of the iLQF with PullUp algorithm with the classic MaxWeight algorithm, and showed that the iLQF with PullUp algorithm consistently performs better that the MaxWeight algorithm.

While this paper has dealt with rate-function optimality, one could potentially consider giving up rate-function optimality for lower computational complexity. A working draft that includes such a complexity vs. rate-function trade-off, as well as a proof of throughput-optimality of iLQF is available in [13].

In closing, the new intuition that emerges from this work is that, for guaranteeing good (low) per-user delay or small per-user queue-lengths, the scheduling algorithm must proceed in an iterative fashion in every timeslot, taking into account the effect of prior resource allocations while making allocation decisions for the remaining resources (channels).

References

[1] S. Bodas, S. Shakkottai, L. Ying, and R. Srikant, "Scheduling in Multi-Channel Wireless Networks: Rate Function Optimality in the Small-Buffer Regime," in *Proc. SIGMETRICS/Performance Conf.*, Jun. 2009.

[2] W. Forum, "Mobile WiMAX Part I: A technical overview and performance evaluation," March 2006, white Paper.

[3] G. T. 25.913, "Requirements for Evolved UTRA (E-UTRA) and Evolved UTRAN (E-UTRAN)," March 2006.

[4] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Trans. Inform. Theory*, vol. 39, pp. 466–478, March 1993.

[5] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting, "CDMA data QoS scheduling on the forward link with variable channel conditions," *Bell Labs Tech. Memo*, April 2000.

[6] S. Shakkottai, "Effective capacity and QoS for wireless scheduling," *IEEE Trans. Automat. Contr.*, vol. 53, no. 3, pp. 749–761, February 2008.

[7] L. Ying, R. Srikant, A. Eryilmaz, and G. Dullerud, "A large deviations analysis of scheduling in wireless networks," *IEEE Trans. Inform. Theory*, vol. 52, no. 11, pp. 5088–5098, November 2006.

[8] A. Stolyar, "Large deviations of queues sharing a randomly time-varying server," *Queueing Systems*, vol. 59, pp. 1–35, 2008.

[9] V. J. Venkataramanan and X. Lin, "Structural properties of LDP for queue-length based wireless scheduling algorithms," in *Proc. Ann. Allerton Conf. Communication, Control and Computing*, Monticello, Illinois, September 2007.

[10] A. Stolyar, "MaxWeight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic," *Ann. Appl. Prob.*, vol. 14, no. 1, 2004.

[11] S. Shakkottai, R. Srikant, and A. Stolyar, "Pathwise optimality of the exponential scheduling rule for wireless channels," *Ann. Appl. Prob.*, vol. 36, no. 4, pp. 1021–1045, December 2004.

[12] S. Meyn, "Stability and asymptotic optimality of generalized MaxWeight policies," *SIAM J. Control and Optimization*, vol. 47, no. 6, pp. 3259–3294, 2009.

[13] S. Bodas, S. Shakkottai, L. Ying, and R. Srikant, "SSG Tech Report: Working Draft," http://users.ece.utexas.edu/~bodas/ssg-tech-report-wip.pdf, 2009.

[14] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Automat. Contr.*, vol. 4, pp. 1936–1948, December 1992.

[15] S. Shakkottai and A. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," *Ann. Math. Statist.*, vol. 207, pp. 185–202, 2002.

[16] M. J. Neely, E. Modiano, and C. E. Rohrs, "Power and server allocation in a multi-beam satellite with time varying channels," in *Proc. IEEE Infocom*, vol. 3, New York, NY, June 2002, pp. 1451–1460.

[17] A. Eryilmaz, R. Srikant, and J. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Trans. Network.*, vol. 13, pp. 411–424, April 2005.

[18] A. Ganti, E. Modiano, and J. Tsitsiklis, "Optimal transmission scheduling in symmetric communication models with intermittent connectivity," *IEEE Trans. Inform. Theory*, vol. 53, pp. 998–1008, March 2007.

[19] M. J. Neely, "Delay Analysis for Max Weight Opportunistic Scheduling in Wireless Systems," in *Forty-Sixth Annual Allerton Conference On Communication, Control, and Computing*, Sep. 2008.

[20] S. Kittipiyakul and T. Javidi, "Delay-Optimal Server Allocation in Multiqueue Multiserver Systems with Time-Varying Connectivities," *IEEE Trans. Inform. Theory*, vol. 55, no. 5, pp. 2319 – 2333, May 2009.

[21] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*, 2nd ed. Springer-Verlag New York, Inc., 1998.

[22] J. Kleinberg and E. Tardos, *Algorithm Design*. Pearson Education, 2006.

[23] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs," *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, Dec. 1973.

[24] R. Durrett, *Probability: Theory and Examples*, 3rd ed. Brooks-Cole, Thomson Learning, 2005.

Appendix A
Proof of Theorem 2

Claim: For given values of $p, q \in (0,1)$, there exists $n_0 = n_0(p,q)$ such that for all $n \geq n_0$, the queuing system Υ_n can be stabilized by some service rule.

Proof: Consider a service rule where in each timeslot, each server uniformly and randomly picks a queue to which it has an ON channel, and serves it. If that particular chosen queue is empty, then that server does not serve any queue in that timeslot. (Multiple servers can serve the same queue, but there is no co-ordination between the servers.)

Then, the probability that the first server offers its service to the first queue in a particular timeslot is

$$\mathbb{P}(Y_{11}(t) = 1) = \mathbb{P}(Y_{11}(t) = 1 | X_{11}(t) = 1) \cdot \mathbb{P}(X_{11}(t) = 1).$$

Now, for the service rule under consideration,

$$\begin{aligned}
&\mathbb{P}(Y_{11}(t) = 1 | X_{11}(t) = 1) \\
&= \sum_{j=0}^{n-1} \mathbb{P}(S_1 \text{ offers service to } Q_1 \text{ in timeslot t} | X_{11}(t) = 1, \\
&\quad \text{Exactly } j \text{ of the rest } n-1 \text{ channels from } S_1 \text{ are ON}) \\
&\quad \cdot \mathbb{P}(\text{Exactly } j \text{ of the rest } n-1 \text{ channels from } S_1 \text{ are ON}) \\
&= \sum_{j=0}^{n-1} \frac{1}{j+1} \binom{n-1}{j} q^j (1-q)^{n-1-j} \\
&= \frac{1-(1-q)^n}{qn}, \text{ after some calculations.}
\end{aligned}$$

Thus, $\mathbb{P}(Y_{11}(t) = 1) = \dfrac{1-(1-q)^n}{n}$, implying that the total amount of service offered to the first queue (or to any other queue, by symmetry) in timeslot t is $1-(1-q)^n$. If $p < 1$ and $q > 0$ are fixed, then $1-(1-q)^n > p$ for large enough $n \geq n_0(p,q)$, where $n_0(p,q) := \left\lceil \dfrac{\log(1-p)}{\log(1-q)} \right\rceil$, implying that all the queues are stable (positive recurrent) under the specified policy, for n large enough. ■

Appendix B
Proof of Lemma 1

Claim: Consider an undirected bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$, where $\mathcal{U} \cup \mathcal{V}$ is the set of vertices with $|\mathcal{U}| = |\mathcal{V}| = n$, and $\mathcal{E}$ is the set of edges. Every edge $e \in \mathcal{E}$ has one of its endpoints in $\mathcal{U}$ and the other in $\mathcal{V}$. For every node $u \in \mathcal{U}$ and $v \in \mathcal{V}$, the edge (u,v) is present in $\mathcal{E}$ with probability q, independently of all other edges. Then, for large n,

$$(1-q)^n \leq \mathbb{P}(G \text{ has no perfect matching}) \leq 3n(1-q)^n,$$

where a perfect matching is defined as a matching of cardinality n.

Proof: For $\mathcal{A} \subseteq \mathcal{U}$, let $\Gamma(\mathcal{A})$ denote the neighborhood $\mathcal{A}$, i.e.,

$$\Gamma(\mathcal{A}) := \{b \in \mathcal{V} : (a,b) \in \mathcal{E} \text{ for some } a \in \mathcal{A}\}.$$

We know from Hall's theorem ([22], Thm. 7.40) that if a bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$ does not have a perfect matching, then there exists a subset $\mathcal{A} \subseteq \mathcal{U}$ such that $|\Gamma(\mathcal{A})| < |\mathcal{A}|$. Fix a nonempty subset $\mathcal{A} \subseteq \mathcal{U}$ and a subset $\mathcal{B} \subseteq \mathcal{V}$. Let $|\mathcal{A}| = a$. Then, we have

$$\begin{aligned}
&\mathbb{P}(\Gamma(\mathcal{A}) \subseteq \mathcal{B}) \\
&= \mathbb{P}(\text{No node in } \mathcal{A} \text{ connects to any node in } \mathcal{V} \backslash \mathcal{B}) \\
&= (1-q)^{(n-|\mathcal{B}|)a}.
\end{aligned}$$

If the graph has no perfect matching, then by Hall's theorem, there must exist sets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{U}, \mathcal{B} \subseteq \mathcal{V}, |\mathcal{B}| = |\mathcal{A}| - 1$, and $\Gamma(\mathcal{A}) \subseteq \mathcal{B}$. Hence, by union bound over all possible subsets $\mathcal{A} \subseteq \mathcal{U}$ and all possible corresponding subsets $\mathcal{B} \subseteq \mathcal{V}$, we have

$$\begin{aligned}
&\mathbb{P}(G \text{ has no perfect matching}) \\
&\leq \sum_{a=1}^{n} \binom{n}{a} \cdot \binom{n}{a-1} \cdot (1-q)^{a(n-a+1)} \\
&\leq 2 \sum_{a=1}^{\lceil n/2 \rceil} \binom{n}{a} \cdot \binom{n}{a-1} \cdot (1-q)^{a(n-a+1)}, \qquad (8)
\end{aligned}$$

where the last inequality holds with equality if n is even.

We consider the case when n is large, in particular $n > 2$. Now, for $n > 2$ and $1 < a \leq \lceil n/2 \rceil$, $a - 1 \geq a/2$, $n - a \geq n/3$, and we have

$$\begin{aligned}
&\frac{\binom{n}{a}\binom{n}{a-1}(1-q)^{a(n-a+1)}}{n(1-q)^n} \\
&\leq \frac{n^a \cdot n^{a-1} \cdot (1-q)^{a(n-a+1)}}{n(1-q)^n} \\
&\leq n^{2a}(1-q)^{(n-a)(a-1)} \\
&\leq n^{2a}(1-q)^{na/6} \\
&= \exp\left(2a \log n - \frac{na}{6} \log \frac{1}{1-q}\right) \\
&= \exp\left[-\frac{a}{6}\left\{n \log \frac{1}{1-q} - 12 \log n\right\}\right] \\
&\leq \exp\left\{-\frac{a}{6} \cdot \frac{n}{2} \cdot \log \frac{1}{1-q}\right\}, \quad \text{for } n \text{ large enough} \\
&\leq \exp\left\{\frac{-n}{12} \log \frac{1}{1-q}\right\}, \quad \text{since } a > 1.
\end{aligned}$$

Hence, from Equation (8), we have for any fixed $\epsilon > 0$,

$$\begin{aligned}
&\mathbb{P}(G \text{ has no perfect matching}) \\
&\leq \quad 2n(1-q)^n \cdot \left(1 + (\left\lceil \frac{n}{2} \right\rceil - 1)\exp\left\{\frac{-n}{12}\log\frac{1}{1-q}\right\}\right) \\
&\leq \quad 2n(1-q)^n \cdot (1+\epsilon), \quad \text{for } n \text{ large enough.} \qquad (9)
\end{aligned}$$

Now, fix a node $u_i \in \mathcal{U}$. Let E_i denote the event that u_i is an isolated node. Then, $\mathbb{P}(E_i) = (1-q)^n$. It follows that

$$\mathbb{P}(G \text{ has no perfect matching}) \geq (1-q)^n.$$

Hence, putting $\epsilon = 0.5$ in Equation (9), we have (for large enough n)

$$(1-q)^n \leq \mathbb{P}(G \text{ has no perfect matching}) \leq 3n(1-q)^n.$$

This completes the proof. ■

Appendix C
Proof of Theorem 3

Claim: Suppose that a service rule in the iLQF-class has the drain and dominance properties. Then, this iLQF-class service rule results in

$$\liminf_{n\to\infty} \frac{-1}{n}\log\mathbb{P}\left(\max_{1\leq i\leq n} Q_i(0) > b\right) = (b+1)\log\frac{1}{1-q}.$$

Further, by Theorem 1, no other service rule can give a larger value for the left hand side of the above expression.

Proof: The proof proceeds according to the following steps:

1) We know that under any iLQF class algorithm with the drain property, in any given timeslot, the maximum queue-length in the system increases by 1 with a very small probability (at most $\alpha_n = 3n(1-q)^n$). Further, over k_0 timeslots, the maximum queue-length decreases by at least 1 (provided it is nonzero to begin with) with probability at least $1/2$.
2) If we sample the original queue-length process every k_0^{th} timeslot, then the maximum queue-length increases by $1, 2, \ldots, k_0$ with probabilities at most $k_0\alpha_n, \binom{k_0}{2}\alpha_n^2, \ldots, \alpha_n^{k_0}$, and decreases by 1 with probability at least $1/2$. Note that the maximum queue-length in the system (on its own) does not have the Markovian property.
3) We construct a dummy Markov chain by "carefully" adding extra packets to the original Markov chain's queues, where the aforementioned bounds on transition probability are met with equality. By the dominance property, the stationary distribution of the dummy Markov chain stochastically dominates that of the original Markov chain.
4) We derive bounds on the stationary distribution of the dummy Markov chain.

The proof in detail: Under any algorithm in the iLQF class, the queue-length vector

$$\underline{Q}(t) := [Q_1(t), Q_2(t), \ldots, Q_n(t)]^T$$

forms a Markov chain on a countable state space, $\mathbb{W}^n$, where $\mathbb{W} = \{0, 1, 2, \ldots\}$. This is because the algorithm takes decisions based only upon the current queue-lengths and arrivals and channel states. We consider a new Markov chain

$$\underline{Z}(t) := [\underline{Q}(t), \underline{Q}(t-1), \ldots, \underline{Q}(t-k_0+1)],$$

where we recall that k_0 is the parameter in the *Drain* property.

$\underline{Z}(t)$ is a Markov chain under the given algorithm. We sample $\underline{Z}(t)$ every k_0^{th} time-slot to get a Markov chain

$$\underline{B}(t) := \underline{Z}(k_0 t).$$

The Markov chains $\underline{B}(t)$ and $\underline{Z}(t)$ have the same stationary distribution. Let $\underline{B}(t) = [B_{ij}(t)]$ with $1 \leq i \leq n$ and $1 \leq j \leq k_0$. Specifically, $B_{ij}(t) = Q_i(k_0 t - j + 1)$. Define $B^\star(t) := \max_{1\leq i\leq n} B_{i1}(t) = \max_{1\leq i\leq n} Q_i(k_0 t)$. Let $\alpha_n = 3n(1-q)^n$. Then, for all m and sufficiently large n, we have:

$$\begin{aligned}
\mathbb{P}(B^\star(t+1) < m | B^\star(t) = m > 0) &\geq \frac{1}{2} \qquad (10)\\
\mathbb{P}(B^\star(t+1) = m+1 | B^\star(t) = m) &\leq k_0\alpha_n \\
\mathbb{P}(B^\star(t+1) = m+2 | B^\star(t) = m) &\leq \binom{k_0}{2}\alpha_n^2 \\
&\vdots \quad \vdots \\
\mathbb{P}(B^\star(t+1) = m+r | B^\star(t) = m) &\leq \binom{k_0}{r}\alpha_n^r \\
&\vdots \quad \vdots \\
\mathbb{P}(B^\star(t+1) = m+k_0 | B^\star(t) = m) &\leq \alpha_n^{k_0} \\
\mathbb{P}(B^\star(t+1) > m+k_0 | B^\star(t) = m) &= 0. \qquad (11)
\end{aligned}$$

The inequality (10) follows from the Drain property in the statement of the theorem. For the transition probability bounds in (11), we use Lemma 2 (Forward jump bound) which shows that in a given timeslot, the probability that the maximum queue-length increases with a probability at most $\alpha_n = 3n(1-q)^n$. Over k_0 timeslots, for the maximum queue-length to increase by r, there must exist r timeslots in which the queue-length increases by 1, and the union bound over the $\binom{k_0}{r}$ choices yields the desired bounds.

Recall that the dominance property allows us to add packets to a queuing system and the resulting tail probabilities are only larger than without the packet additions. This motivates us to construct a queuing system $\underline{R}$ where the packets are "carefully" added to ensure that the bounds in (10) and (11) are met with equality, as explained in the following.

We consider n large enough such that $(1+\alpha_n)^{k_0} - 1 \leq 0.1$. We consider a queuing system $\underline{R}$ that has the same sample-path wise external arrivals and channel realizations as the system $\underline{Q}$, and we add extra packets to one of the longest queues in $\underline{R}$ at timeslots that are integer multiples of k_0. Define

$$\tilde{\underline{Z}}(t) := [\underline{R}(t), \underline{R}(t-1), \ldots, \underline{R}(t-k_0+1)],$$

$$\tilde{\underline{B}}(t) := \tilde{\underline{Z}}(k_0 t),$$

and

$$\tilde{B}^\star(t) = \max_{1\leq i\leq n} \tilde{\underline{B}}_{i1}(t).$$

We want to ensure that the inequalities (10) and (11) are met with equality if we replace $B^\star(t)$ with $\tilde{B}^\star(t)$.

Matching the transition probabilities: In Appendix P, we define the procedure for adding extra packets to the queuing system. Because the argument is heavy in notation, we present it in a separate Appendix. The conclusion is that under the specified packet addition method, the transition probabilities of $\tilde{B}^\star(t')$ obey (10) and (11) with equality. With the specified packet-additions, the process $\underline{\tilde{B}}(t)$ is Markovian, because given $\underline{\tilde{B}}(t)$, $\{\underline{\tilde{B}}(u)\}_{u>t}$ is independent of $\{\underline{\tilde{B}}(s)\}_{s<t}$.

Thus, for the Markov chain $\underline{\tilde{B}}$, $\tilde{B}^\star(t)$ obeys the inequalities in (10) and (11) with equality. The Markov chain $\underline{\tilde{B}}(t)$ is positive recurrent for n large enough, because it is aperiodic, irreducible and the Lyapunov function for any valid state $\underline{\tilde{B}}$ of the Markov chain is $Lyap(\underline{\tilde{B}}) = \tilde{B}^\star$ (the reason this works is that the maximum queue-length decreases by one with probability half, and increases by a finite amount with an arbitrarily small probability when n is large – thus, there is a negative drift whenever the value of the Lyapunov function is strictly positive, and Foster's theorem completes the proof).

Computing $\mathbb{P}(\tilde{B}^\star(0) > b)$ in the Steady State:
Partition $\mathbb{W}^{k_0 n}$ into the following disjoint sets, for $j \geq 0$:

$$V_j := \left\{ \begin{bmatrix} x_{11} & x_{21} & \ldots & x_{k_0 1} \\ x_{12} & x_{22} & \ldots & x_{k_0 2} \\ \vdots & \vdots & \vdots & \vdots \\ x_{1n} & x_{2n} & \ldots & x_{k_0 n} \end{bmatrix} \in \mathbb{W}^{k_0 n} : \max_{1 \leq i \leq n} x_{1i} = j \right\}.$$

Let $\pi_j := \mathbb{P}(\underline{\tilde{B}}(0) \in V_j)$, i.e., $\pi_j = \mathbb{P}(\tilde{B}^\star(0) = j)$.

Let $V_j = \{v_{j_1}, v_{j_2}, \ldots\}$, $\sigma(j_i) = \mathbb{P}(\underline{\tilde{B}}(0) = v_{j_i})$ in the steady state, and let $p(i_1, j_2)$ denote the probability of transition from state v_{i_1} to state v_{j_2} for the Markov chain $\underline{\tilde{B}}(t)$.

Now consider a "cut" in the state-space of the Markov chain $\underline{\tilde{B}}(t)$ that separates the states $\{x \in V_j : j < m\}$ from the states $\{x \in V_j : j \geq m\}$. In the steady-state, the total probability mass crossing this cut in either direction must be equal (by flow-balance of the (positive recurrent) Markov chain $\underline{\tilde{B}}(\cdot)$). Thus, for every $m \geq 1$,

$$\sum_{j=(m-k_0)^+}^{m-1} \sum_{\ell=0}^{k_0+j-m} \mathbb{P}(\underline{\tilde{B}}(t) \in V_j, \underline{\tilde{B}}(t+1) \in V_{m+\ell}) = \mathbb{P}(\underline{\tilde{B}}(t) \in V_m, \underline{\tilde{B}}(t+1) \in V_{m-1}).$$

The above equation holds because from any given state in V_j, the Markov chain $\underline{\tilde{B}}(t)$ can transition to some state in V_ℓ for $(j-1)^+ \leq \ell \leq j + k_0$. Rewriting, for all $m \geq 1$,

$$\sum_{j=(m-k_0)^+}^{m-1} \pi_j \sum_{\ell=0}^{k_0+j-m} \mathbb{P}(\underline{\tilde{B}}(t+1) \in V_{m+\ell} \mid \underline{\tilde{B}}(t) \in V_j) = \pi_m \mathbb{P}(\underline{\tilde{B}}(t+1) \in V_{m-1} \mid \underline{\tilde{B}}(t) \in V_m),$$

or

$$\sum_{j=(m-k_0)^+}^{m-1} \pi_j \sum_{\ell=0}^{k_0+j-m} \overbrace{\mathbb{P}(\tilde{B}^\star(t+1) = m+\ell \mid \tilde{B}^\star(t) = j)}^{=\binom{k_0}{m+\ell-j}\alpha_n^{m+\ell-j}} = \pi_m \underbrace{\mathbb{P}(\tilde{B}^\star(t+1) = m-1 \mid \tilde{B}^\star(t) = m)}_{=1/2}. \quad (12)$$

We consider n large enough, so that $\alpha_n = 3n(1-q)^n \leq 1$. For $m \geq 0$, we prove the following statement about π_m by induction:

$$g(m) : \pi_m \leq \pi_0 \cdot 5^{k_0 m} \alpha_n^m.$$

Base Case:
Clearly, $g(0)$ is true, since $\pi_0 = \pi_0$.

Induction Step:
Let $g(0), g(1), \ldots, g(m-1)$ be true, and we need to prove $g(m)$. From (10), (11) and (12), and noting that $\pi_j = 0$ for $j < 0$, we have:

$$\begin{aligned} \frac{\pi_m}{2} &= \pi_{m-1} \sum_{j=1}^{k_0} \binom{k_0}{j} \alpha_n^j + \pi_{m-2} \sum_{j=2}^{k_0} \binom{k_0}{j} \alpha_n^j \\ &\quad + \cdots + \pi_{m-k_0} \alpha_n^{k_0}. \end{aligned}$$

Thus,

$$\begin{aligned} \pi_m &= 2 \sum_{r=1}^{k_0} \left(\pi_{m-r} \sum_{j=r}^{k_0} \binom{k_0}{j} \alpha_n^j \right) \\ &\leq 2 \sum_{r=1}^{k_0} \left(\pi_{m-r} \alpha_n^r \sum_{j=r}^{k_0} \binom{k_0}{j} \right) \\ &\leq 2 \sum_{r=1}^{k_0} \left(\pi_{m-r} \alpha_n^r 2^{k_0} \right) \\ &\overset{(a)}{\leq} 2 \sum_{r=1}^{k_0} \left(\pi_0 \cdot 5^{k_0(m-r)} \alpha_n^{m-r} \alpha_n^r 2^{k_0} \right) \\ &= 2^{k_0+1} \alpha_n^m \cdot \pi_0 \sum_{r=1}^{k_0} 5^{k_0(m-r)} \\ &= \pi_0 \alpha_n^m 2^{k_0+1} 5^{k_0(m-k_0)} \frac{5^{k_0^2} - 1}{5^{k_0} - 1} \\ &\leq \pi_0 \alpha_n^m 5^{k_0 m} \frac{2^{k_0+1}}{5^{k_0} - 1} \\ &\leq \pi_0 \alpha_n^m 5^{k_0 m}, \end{aligned}$$

since $2^{k+1} \leq 5^k - 1$ for all $k \geq 1$. Here, the inequality (a) follows from induction hypothesis. Hence, by principle of mathematical induction, $g(m)$ is true for all integers $m \geq 0$.

Now,

$$\sum_{m=1}^{\infty} 5^{k_0 m} [3n(1-q)^n]^m = \frac{5^{k_0}[3n(1-q)^n]}{1 - 5^{k_0}[3n(1-q)^n]} \overset{n \to \infty}{\to} 0.$$

Since $\pi_0 + \pi_1 + \cdots = 1$, we have

$$\pi_0 = \frac{1}{1 + \sum_{m=1}^{\infty} \frac{\pi_m}{\pi_0}} \geq \frac{1}{1 + \sum_{m=1}^{\infty} 5^{k_0 m}[3n(1-q)^n]^m} > \frac{1}{3},$$

for n large enough. Further,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(0) > b) \\
&= \sum_{m=b+1}^{\infty} \pi_m = \pi_0 \sum_{m=b+1}^{\infty} \frac{\pi_m}{\pi_0} \\
&\leq \pi_0 5^{k_0(b+1)}[3n(1-q)^n]^{b+1} \sum_{m=0}^{\infty} 5^{k_0 m}[3n(1-q)^n]^m \\
&= \pi_0 5^{k_0(b+1)}[3n(1-q)^n]^{b+1} \frac{1}{1-5^{k_0}[3n(1-q)^n]} \\
&\leq 2\pi_0 5^{k_0(b+1)}[3n(1-q)^n]^{b+1},
\end{aligned}$$

for n large enough. Hence,

$$\begin{aligned}
&\limsup_{n\to\infty} \tfrac{1}{n} \log \mathbb{P}(\tilde{B}^\star(0) > b) \\
&\leq \limsup_{n\to\infty} \frac{1}{n} [\log 2 + \log \pi_0 + k_0(b+1)\log 5 \\
&\quad + (b+1)\log(3n) + n(b+1)\log(1-q)]
\end{aligned}$$

Noting that $\limsup_n (a_n + b_n) \leq \limsup_n a_n + \limsup_n b_n$ and that $\limsup_{n\to\infty} \frac{1}{n} \log \pi_0 = 0$ since $\pi_0 > \frac{1}{3}$ for n large enough, we get

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}(\tilde{B}^\star(0) > b) \geq (b+1) \log \frac{1}{1-q},$$

and, by (4), the proof of Theorem 3 is complete. ■

Appendix D
Proof of Lemma 3

Claim: The output $\mathcal{M}'$ of PullUp$(G, \mathcal{M}, \mathcal{V})$ is a matching, and $|\mathcal{M}| = |\mathcal{M}'|$.

Proof: We prove that if $\mathcal{M}_k$ is a matching, then so is $\mathcal{M}_{k+1}$, and $|\mathcal{M}_k| = |\mathcal{M}_{k+1}|$. We need to focus only on the case where in step 2 of PullUp, the node v_k has no incoming edge and there exists $v_l \in \Delta(G_k, v_k)$ with $l > k$. Let the path from v_k to v_l be $v_k \to u_{i_1} \to v_{j_1} \to u_{i_2} \to v_{j_2} \cdots \to u_{i_c} \to v_l$. Let

$$\mathcal{M}_k = \{(u_{i_1}, v_{j_1}), (u_{i_2}, v_{j_2}), \ldots, (u_{i_x}, v_{j_x})\},$$

with $v_{j_c} = v_l$, and $v_{j_y} \neq v_k$ for any y. Then, by definition,

$$\begin{aligned}
\mathcal{M}_{k+1} = \; & \{(u_{i_1}, v_k), (u_{i_2}, v_{j_1}), \ldots, (u_{i_c}, v_{i_{c-1}}), \\
& (u_{i_{c+1}}, v_{j_{c+1}}), \ldots, (u_{i_x}, v_{j_x})\}.
\end{aligned}$$

Hence, $|\mathcal{M}_{k+1}| = |\mathcal{M}_k|$. Further, the edges in $\mathcal{M}_{k+1}$ are node-disjoint because all the nodes $v_k, v_{i_1}, v_{i_2}, \ldots, v_{i_x}$ are different. Hence, $\mathcal{M}_{k+1}$ is a matching. Therefore, $\mathcal{M}' = PullUp(G, \mathcal{M}, \mathcal{V})$ is a matching and $|\mathcal{M}'| = |\mathcal{M}|$. ■

Appendix E
Proof of Lemma 4

Claim: The proposed algorithm (iLQF with PullUp) can be implemented in $O(n^4)$ computations per timeslot.

Proof: Consider a bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$ with a given matching $\mathcal{M}$, $|\mathcal{U}| = |\mathcal{V}| = n$ and $|\mathcal{E}| = m$.

1) For each node $v_k \in \mathcal{V}$, the set of all nodes reachable from v_k can be found in $O(m+n)$ computations via Depth First Search (DFS) (Theorem 3.13 in [22]). Since $m = O(n^2)$ and there are at most n nodes from which the set of reachable nodes needs to be found out, the operation $PullUp(G, \mathcal{M}, \mathcal{V})$ can be completed in $O(n^3)$ operations.
2) For a matching

$$\mathcal{M} = \{(u_{a_1}, v_{b_1}), (u_{a_2}, v_{b_2}), \ldots, (u_{a_k}, v_{b_k})\},$$

define

$$SUM(\mathcal{M}) := \textstyle\sum_{i=1}^{k} (a_i + b_i).$$

Let $PullUp(G, \mathcal{M}, \mathcal{V}) = \mathcal{M}'$. If $\mathcal{M} \neq \mathcal{M}'$, then

$$SUM(\mathcal{M}') \leq SUM(\mathcal{M}) - 1.$$

Since $SUM(\mathcal{M}_1)$ is $O(n^2)$, the number of times the PullUp operation is performed in the step 2b of the algorithm is $O(n^2)$.

The step 1 of the algorithm can be implemented in $O(n)$ computations. For step 2, the set $\mathcal{Q}_L$ can be found in $O(n)$ computations. From [23], we know that in a bipartite graph $G(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$ with $|\mathcal{U}| = |\mathcal{V}| = n$ and $|\mathcal{E}| = m$, it is possible to find a largest cardinality matching in $O(m\sqrt{n})$ computations. Adding isolated dummy nodes if necessary, the graph G_L can be made to have n nodes on either side. Further, $m = O(n^2)$. A largest matching can therefore be found with $O(n^{2.5})$ computations. For any fixed server node, the set of all nodes to which this server node has a path can be found by depth-first search (DFS) in $O(m+n)$ computations ([22], Thm. 3.13) and since $m = O(n^2)$, in $O(n^2)$ computations. The step 2a can thus be performed in $O(n^3)$ computations, while the step 2b can be performed in $O(n^4)$ computations. Noting that the step 2b needs to be executed at most once, every round (except possibly the last round) can be implemented in $O(n^3)$ computations, and the last round can be performed in $O(n^4)$ computations.

By step 3, the number of rounds to be completed is at most L. However, if $\mathcal{Q}_L = \emptyset$ for a round, then no computations need to be performed for that round at all, since $\mathcal{M}_L$ is the vacuous matching of cardinality 0. Hence, instead of redefining L to $L-1$ in step 3, we can define L to be the length of the longest queue at the end of that round, and the algorithm will result in the same set of allocations of queues to servers as before. The maximum of the queue-lengths can be obtained in $O(n)$ computations. With this modification, the number of rounds is at most n, since every round allocates at least one server, or else it is the last round.

Thus, all the perfect matching rounds (i.e., except possibly the last round of maximal matching) can be implemented in $O(n^4)$ computations, and the last round of largest matching (step 2b) can be implemented in $O(n^4)$ computations. Finally, the output (updated queue-lengths and server allocation decisions) can be reported in $O(n)$ computations (memory reads), since at most n of the $Y_{ij}(t)$ are nonzero.

Therefore, the proposed algorithm can be implemented in $O(n^4)$ computations per timeslot. ■

Appendix F
Proof of Lemma 5

Claim: Under the notation of Definition 5, in the graph G_{n+1}, if a node v_a has no incoming edge, then there does not exist a (directed) path from v_a to any node v_b with $b > a$. Consequently, if $\text{PullUp}(G, \mathcal{M}, \mathcal{V}) = \mathcal{M}'$, then $\text{PullUp}(G, \mathcal{M}', \mathcal{V}) = \mathcal{M}'$.

Proof: Consider $b > a$. If the node v_a has an incoming edge in the graph G_b, then it has an incoming edge in G_{b+1}, even if $G_b \neq G_{b+1}$. This is because if $G_b \neq G_{b+1}$, then the node v_b does not have an incoming edge in G_b and there exists a path from v_b to v_c, $c > b$ in G_b. Even if this path contains v_a, the incoming edge to v_a (in G_b) becomes an outgoing edge, while another one of v_a's outgoing edges, on reversal, becomes an incoming edge, since the path cannot terminate on v_a. Hence, if v_a has an incoming edge in G_{a+1}, then it has an incoming edge in G_{n+1}.

Now, let $v_a \in \mathcal{V}$ and assume that v_a has no incoming edge in G_{n+1}, therefore in G_{a+1}, implying $G_a = G_{a+1}$. The following notation is used throughout this proof:

$\mathcal{N}_a(b)$	=	The set of all nodes reachable from v_a in the graph G_b
$\Gamma(G_b, v_a)$	=	$\mathcal{N}_a(b) \cap \mathcal{U}$
$\Delta(G_b, v_a)$	=	$\mathcal{N}_a(b) \cap \mathcal{V}$
$\mathcal{E}_a$	=	The set of edges in the graph G_a

(According to this notation, $\mathcal{N}_a = \mathcal{N}_a(a)$.) Thus, we have $\Delta(G_a, v_a) \subseteq \{v_1, v_2, \ldots, v_{a-1}\}$, and $\Gamma(G_a, v_a) = \mathcal{S}_1 \cup \mathcal{S}_2$, where

$$\begin{aligned}\mathcal{S}_1 &= \{u_i : (u_i, v_j) \in \mathcal{M}_a \text{ for some } v_j \in \Delta(G_a, v_a)\},\\ \mathcal{S}_2 &= \{u_i : (v_j, u_i) \in \mathcal{E}_a \text{ for some } v_j \in \Delta(G_a, v_a),\\ &\quad \text{and } u_i \text{ has no outgoing edge}\}.\end{aligned}$$

To see this, note that any node $v_j \in \Delta(G_b, v_a)$ has exactly one incoming (forward) edge, since the forward edges belong to a matching. Therefore, $\mathcal{S}_1 \cup \mathcal{S}_2 \subseteq \Gamma(G_a, v_a)$. Further, every node $u_i \in \Gamma(G_a, v_a)$ is reachable from v_a, so if u_i has an outgoing (forward) edge, it must, by definition, end on some $v_j \in \Delta(G_a, v_a)$, or else it must not have an outgoing edge. Hence, $\Gamma(G_a, v_a) = \mathcal{S}_1 \cup \mathcal{S}_2$.

We now prove the following statement, which immediately implies the claim: Fix any $b > a$. In the graph G_b, if v_b has no incoming edge, and there exists a path from v_b to v_c (with $c > b$) in G_b, then that path does not contain any node from $\mathcal{N}_a$. $(\star)$

Before proving the statement $(\star)$, let us see how it implies the claim that in the graph G_{n+1}, there exists no directed path from v_a to any node v_b for $b > a$. Fix any $b > a$. If v_b has an incoming edge, then we have $G_b = G_{b+1}$. If v_b has no incoming edge, and in the graph G_b there exists no path from v_b to v_c for any $c > b$, then again $G_b = G_{b+1}$. If v_b has no incoming edge in G_b and there exists a path from v_b to v_c with $c > b$, then (by $(\star)$) this path contains no node from $\mathcal{N}_a$. Thus, in every possible scenario, the edge-configuration (i.e., the (directed and labeled) pattern of incoming and outgoing edges) for the nodes in $\mathcal{N}_a$ is the same in both the graphs G_b and G_{b+1}. Thus, if the node v_a has no incoming edge in the graph G_{a+1}, then it does not have an incoming edge in the graph G_{n+1}, proving the first part of the claim.

Now, if $\text{PullUp}(G, \mathcal{M}', \mathcal{V}) \neq \mathcal{M}'$, then the following is true: if all the edges in G that belong to $\mathcal{M}'$ are forward edges and all the other edges are backward, then there exists a node v_a with no incoming edges, and has a directed path to a node v_b, with $b > a$. No such path exists by the previous argument, completing the proof if $(\star)$ is true.

Proof of $(\star)$:

If the statement $(\star)$ is true, then the set of nodes in $\mathcal{V}$ reachable from v_a under G_a is the same as those under G_b for any $b > a$, in particular for $b = n+1$. Suppose, for obtaining a contradiction, that the statement $(\star)$ is false, and let b be the smallest index greater than a for which the statement is false. Therefore, $\mathcal{N}_a(a) = \mathcal{N}_a(b)$, since the edge-configurations for all the nodes in $\mathcal{N}_a(a)$ are unchanged under the transformations that convert the graph G_a to G_b, by definition of b. Let the path from v_b to a node v_c, $c > b$ contain one or more nodes from $\mathcal{N}_a(a)$. Let this path be $v_b \to u_{i_1} \to v_{j_1} \to u_{i_2} \to v_{j_2} \to \cdots \to v_{j_k} = v_c$. This path does not contain any node from $\mathcal{S}_2$, since the nodes in $\mathcal{S}_2$ have no outgoing edges. If the path contains a node $u_i \in \mathcal{S}_1$, then it contains the node v_j for which $(u_i, v_j) \in \mathcal{M}_b$, the *only* forward edge associated with u_i. Since the edge configurations of the nodes in $\mathcal{N}_a(a) = \mathcal{N}_a(b)$ are the same under the graphs G_a and G_b, we have $(u_i, v_j) \in \mathcal{M}_a$. Hence, the path from v_b to v_c contains at least one node from $\Delta(G_a, v_a) = \Delta(G_b, v_a)$.

Define

$$j_0 := \min\{j : v_\ell \notin \mathcal{N}_a(b) \ \forall \ \ell \geq j\}.$$

Since $v_c = v_{j_k} \notin \mathcal{N}_a(b)$, we have $0 < j_0 \leq k$. Then, $v_{j_0 - 1} \in \mathcal{N}_a(a)$, and the edge $v_{j_0-1} \to u_{j_0}$ is present in G_a, since the edge configuration of v_{j_0-1} is the same under G_a and G_b. Hence, $u_{j_0} \in \mathcal{N}_a(a)$, and the edge configuration of u_{j_0} is the same under G_a and G_b, implying $v_{j_0} \in \mathcal{N}_a(a)$, a contradiction to the definition of j_0 since $\mathcal{N}_a(a) = \mathcal{N}_a(b)$. Therefore, the statement $(\star)$ is true. ∎

Appendix G
Proof of Lemma 6

Claim: Consider two queuing systems $\underline{Q}$ and $\underline{R}$ with queues $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_n\}$ and $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ respectively, with the property that $Q_i(t-1) \leq R_i(t-1)$ for all i. Let the two systems have identical channel realizations, $X_{ij}(t)$ and identical arrivals, $A_i(t)$ for $1 \leq i, j \leq n$. Both the queuing systems implement the algorithm described in Section VIII, i.e. iLQF with PullUp. Then, $Q_i(t) \leq R_i(t)$ for all i.

Proof: The proof proceeds according to the following steps: in timeslot t, if the number of perfect queue-matching rounds in the system $\underline{R}$, n_R, is less than that in the system $\underline{Q}$, n_Q, then it is straightforward to show that the claimed queue-length inequalities (i.e., $Q_i(t) \leq R_i(t)$ for all i) holds. If $n_R = n_Q + w > n_Q$, then we inductively show that

1) The set of servers available for allocation in round r in the system $\underline{Q}$ is a superset of those available in the round $r + w$ in the system $\underline{R}$, and
2) The set of longest queues in the system $\underline{Q}$ in the round r is a subset of those in the system $\underline{R}$ in the round $r + w$.

Note that the *length* of the queue(s) served in round r in the system $\underline{Q}$ is the same as those served in th system $\underline{R}$ in the round $r + w$. Hence, when the algorithm reaches a given queue-length level in the two systems, it has "more" resources (servers) and "fewer" queues to work with in the system $\underline{Q}$ than in the system $\underline{R}$, where more, fewer, etc. are used in the set inclusion sense. Thus the claimed queue-length inequality holds in this case as well.

The proof in detail: The following notation is used throughout this proof.

$\mathcal{M}_r$	=	The set of queues served in the r^{th} round, in the system $\underline{R}$
$\mathcal{Y}_r$	=	The set of servers allocated in the r^{th} round, in the system $\underline{R}$
$\mathcal{N}_r$	=	The set of queues served in the r^{th} round, in the system $\underline{Q}$
$\mathcal{Z}_r$	=	The set of servers allocated in the r^{th} round, in the system $\underline{Q}$

For simplicity of notation, throughout this proof, we use $Q_i^{(r)}$ and $R_i^{(r)}$ to denote $Q_i^{(r)}(t)$ and $R_i^{(r)}(t)$ respectively. By definition, $R_i^{(0)} := R_i(t-1)+A_i(t)$ and $Q_i^{(0)} := Q_i(t-1)+A_i(t)$. Let $\hat{R} := \max_i R_i^{(0)}$, $\hat{Q} := \max_i Q_i^{(0)}$ and $w := \hat{R}-\hat{Q}$. Let there exist n_R and n_Q rounds of perfect queue matchings in the system $\underline{R}$ and $\underline{Q}$ respectively.

Case 1: $n_R < w$.
If a queue R_i was served even once in the n_R rounds, then at the end of n_R rounds, $R_i^{(n_R)} = \hat{R} - n_R > \hat{R} - w = \hat{Q}$. Since there are exactly n_R rounds of perfect queue matching in the system $\underline{R}$,

$$R_i(t) \geq R_i^{(n_R)} - 1 \geq \hat{Q} \geq Q_i(t).$$

If R_i was not served even once in the first n_R rounds of perfect queue matching, but was served in the last round of maximal matching, then

$$R_i(t) = \hat{R} - (n_R+1) \geq \hat{R} - w = \hat{Q} \geq Q_i(t).$$

Finally, if the queue R_i was not served at all, then

$$R_i(t) = R_i(t-1)+A_i(t) \geq Q_i(t-1)+A_i(t) \geq Q_i(t),$$

and the claim is true in this case.

Case 2: $n_R = w$.
We have $R_i^{(n_R)} \geq \hat{R} - n_R = \hat{Q}$, with equality holding if and only if $R_i^{(0)} \geq \hat{Q}$. Let $\mathcal{R}_{last} = \{R_{i_1}, R_{i_2}, \ldots, R_{i_a}\}$ denote the set of longest (i.e. of length $\hat{Q}$) queues at the beginning of the maximal matching round for the system $\underline{R}$, with $i_1 < i_2 < \cdots < i_a$. Let $\mathcal{Q}_{first} = \{Q_{j_1}, Q_{j_2}, \ldots, Q_{j_b}\}$ denote the set of longest queues in the system $\underline{Q}$, at the beginning of the first round, with $j_1 < j_2 < \cdots < j_b$. Then, $\{j_1, j_2, \ldots, j_b\} \subseteq \{i_1, i_2, \ldots, i_a\}$. If the first round in the system $\underline{Q}$ is a perfect queue matching round (i.e. $n_Q > 0$), then all of the queues in $\mathcal{Q}_{first}$ are served, and only some of $\mathcal{R}_{last}$, and the claim is true because the queues in the system $\underline{R}$ are not served for more than n_R+1 rounds.

Now, let $n_Q = 0$. Let a queue R_{i_c} be served by a server S_a in the $(n_R+1)^{th}$ round, but Q_{i_c} is not served in the 1^{st} (maximal matching) round. Then, S_a must serve a queue Q_{i_d} with $d < c$, otherwise the size of the largest matching can be strictly increased ($\because X_{i_c a} = 1$), or there exists a directed path $Q_{i_c} \rightarrow S_a \rightarrow Q_{i_d}$, contradicting Lemma 5 (No edge, no path). Specifically, the queue Q_{i_c} has no incoming edge, because it is not assigned a server, and has a directed path to a higher-indexed queue Q_{i_d}, contradicting Lemma 5 (No edge, no path).

The queue R_{i_d} must be served by a server S_e, otherwise there exists a directed path $R_{i_d} \rightarrow S_d \rightarrow R_{i_c}$, again contradicting Lemma 5 (No edge, no path). The server S_e must serve a queue Q_{i_f} with $f < c$, otherwise the size of the largest matching in $\underline{Q}$ can be strictly increased (by allocating S_e to Q_{i_d}, S_a to Q_{i_c}), or there exists a directed path $Q_{i_c} \rightarrow S_a \rightarrow Q_{i_d} \rightarrow S_e \rightarrow Q_{i_f}$ and $f > c$, contradicting the specifications of the algorithm and in particular, Lemma 5 (No edge, no path). This process of finding newer servers and queues in the two systems can be continued indefinitely, contradicting the finiteness of the number of queues and servers in the system. Therefore, if a queue R_{i_c} is served in the largest matching round of the system $\underline{R}$, then so is Q_{i_c} in the system $\underline{Q}$, and the claim holds in this case.

Case 3: $n_R > w$.

We prove the following statement $f(r)$, for $0 \leq r \leq n_R - w$, by induction: let $f(r) = f_1(r) \wedge f_2(r) \wedge f_3(r)$, where

$$\begin{aligned} f_1(r) &: \quad \mathcal{N}_r \subseteq \bigcup_{i=1}^{r+w} \mathcal{M}_i, \\ f_2(r) &: \quad \mathcal{Z}_r \subseteq \bigcup_{i=1}^{r+w} \mathcal{Y}_i, \quad \text{and} \\ f_3(r) &: \quad Q_i^{(r)} \leq R_i^{(r+w)} \text{ for all } 1 \leq i \leq n. \end{aligned}$$

Base case:

We need to prove that $f(0)$ is true. Since $\mathcal{N}_0 = \emptyset$ and $\mathcal{Z}_0 = \emptyset$, we only need to prove that $Q_i^{(0)} \leq R_i^{(w)}$. If R_i is not served during the first w rounds, then $R_i^{(w)} = R_i^{(0)} \geq Q_i^{(0)}$. If R_i was served in at least one of the first w rounds of service, then $R_i^{(w)} \geq \hat{R} - w = \hat{Q} \geq Q_i^{(0)}$. Hence, $f(0)$ is true.

Induction step:

Suppose $f(0), \ldots, f(r-1)$ are true for some $r \geq 1$. We need to prove $f(r)$.

We first show that $f(0), \ldots, f(r-1)$ implies $f_1(r)$. For some $1 \leq i \leq n$, let $Q_i \in \mathcal{N}_r$. We show that $R_i \in \mathcal{M}_{r+w}$.

Since $r \leq n_R - w$, we have $r-1+w \leq n_R - 1$. Hence the $(r-1+w)^{th}$ round (and all of the rounds before that) in the system $\underline{R}$ are perfect queue matching rounds. Hence, $R_i^{(r-1+w)} \leq \hat{R} - (r-1+w) = \hat{Q} - (r-1)$. Since the queue Q_i is served in the r^{th} round, at the end of $r-1$ rounds it is one of the longest queues in the system $\underline{Q}$. Since a queue can lose at most one packet in a given round, we have $\hat{Q} - (r-1) = Q_i^{(r-1)}$. Together, the last two inequalities imply $R_i^{(r-1+w)} \leq Q_i^{(r-1)}$. By induction hypothesis, we have $R_i^{(r-1+w)} \geq Q_i^{(r-1)}$. Thus, all of the previous inequalities must hold with equality, and

$$\hat{R} - (r-1+w) = R_i^{(r-1+w)} = Q_i^{(r-1)} = \hat{Q} - (r-1).$$

Thus, if $Q_i \in \mathcal{N}_r$, then the queue R_i is a longest queue at the beginning of the $(r+w)^{th}$ round in the system $\underline{R}$. Since $r+w \leq n_R$, the $(r+w)^{th}$ round in the system $\underline{R}$ is a perfect queue-matching round, implying that the queue R_i is allocated a server, i.e., $R_i \in \mathcal{M}_{r+w}$. This proves that $f(0), \ldots, f(r-1)$ implies $f_1(r)$.

Now we show that $f(0), \ldots, f(r-1)$ implies $f_3(r)$. For some $1 \leq i \leq n$, let $R_i \in \mathcal{M}_{r+w}$. By induction hypothesis, we have $R_i^{(r-1+w)} \geq Q_i^{(r-1)}$. If $R_i^{(r-1+w)} > Q_i^{(r-1)}$, then (because R_i can lose at most 1 packet in a given round) we have $R_i^{(r+w)} \geq Q_i^{(r)}$, and the claim holds.

If $R_i^{(r-1+w)} = Q_i^{(r-1)}$, then we show that $Q_i \in \mathcal{N}_r$. Since $R_i \in \mathcal{M}_{r+w}$, it was, at the beginning of that round, a longest queue. Let $R_i \in \mathcal{M}_{r+w}$ be allocated a server S_a in the $(r+w)^{th}$ round. Therefore, $X_{ia} = 1$. Since (by induction hypothesis)

$$\bigcup_{i=1}^{r-1} \mathcal{Z}_i \subseteq \bigcup_{i=1}^{r-1+w} \mathcal{Y}_i, \text{ and } S_a \notin \bigcup_{i=1}^{r-1+w} \mathcal{Y}_i,$$

we have

$$S_a \notin \bigcup_{i=1}^{r-1} \mathcal{Z}_i,$$

so the server S_a is available to serve Q_i in the r^{th} round. Therefore, if there exists a perfect matching in the system $\underline{R}$ in the $(r+w)^{th}$ round, then there exists a perfect matching in the r^{th} round in the system $\underline{Q}$: a perfect queue matching can be found by mimicking the corresponding allocations in the system $\underline{R}$. Thus, $Q_i \in \mathcal{N}_r$, implying that $Q_i^{(r)} \leq R_i^{(r+w)}$ and proving that $f(0), \ldots, f(r-1)$ implies $f_3(r)$.

Next we show that $f(0), \ldots, f(r-1)$ implies $f_2(r)$. For the purpose of obtaining a contradiction, let $S_c \in \mathcal{Z}_r$, and $S_c \notin \mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_{r+w}$. Let Q_i be served by S_c in the r^{th} round, while R_i was served by S_d in $(r+w)^{th}$ round. Hence, $d < c$. S_d must serve some queue Q_e in the system $\underline{Q}$ in r^{th} round, because otherwise it can replace S_c to serve Q_i and the server S_d was unused (in the system $\underline{Q}$) until the beginning of the r^{th} round by induction hypothesis. R_e, in turn, must be served by a server S_f in the $(r+w)^{th}$ round in the system $\underline{R}$. We must have $f < c$, otherwise there exists a connecting path $S_c \to R_i \to S_d \to R_e \to S_f$ and S_c cannot remain unused in the system $\underline{R}$, according to Lemma 5 (No edge, no path). This process can be continued indefinitely, contradicting the fact that the number of queues and servers is finite. Hence, $\mathcal{Z}_r \subseteq \mathcal{Y}_1 \cup \mathcal{Y}_2 \cup \cdots \cup \mathcal{Y}_{r+w}$. Thus $f(0), \ldots, f(r-1)$ implies $f_2(r)$, and the induction is complete, i.e., $f(0), \ldots, f(r-1)$ implies $f(r)$.

Hence, if we compare the state of the system $\underline{R}$ after n_R rounds of perfect matching (i.e. at the beginning of the maximal matching round) and $\underline{Q}$ at the end of $n_R - w$ rounds of perfect matching, we have the following structure:

1) The set of unallocated servers available in the system $\underline{Q}$ is a superset of the set of unallocated servers available in the system $\underline{R}$.
2) The set of longest queues in the system $\underline{Q}$ is a subset of the set of longest queues in the system $\underline{R}$.

As before, let $\mathcal{R}_{last} = \{R_{i_1}, R_{i_2}, \ldots, R_{i_a}\}$ denote the set of longest queues at the beginning of the maximal matching round for the system $\underline{R}$, with $i_1 < i_2 < \cdots < i_a$. Let $\mathcal{Q}_{first} = \{Q_{j_1}, Q_{j_2}, \ldots, Q_{j_b}\}$ denote the set of longest queues in the system $\underline{Q}$, at the beginning of the $(n_R - w + 1)^{th}$ round, with $j_1 < j_2 < \cdots < j_b$. Then, $\{j_1, j_2, \ldots, j_b\} \subseteq \{i_1, i_2, \ldots, i_a\}$. If the $(n_R - w + 1)^{th}$ round in the system $\underline{Q}$ is a perfect matching round (i.e. $n_Q > n_R - w$), then all of the queues in $\mathcal{Q}_{first}$ are served, and only some of $\mathcal{R}_{last}$, and the claim is true because the queues in the system $\underline{R}$ are not served for more than $n_R + 1$ rounds.

Now, let $n_Q = n_R - w$. We need to prove that if a queue R_i is served in the largest matching round of the system $\underline{R}$, then so is Q_i in the system $\underline{Q}$. The proof is almost identical to that of the case $n_R = w$, and is skipped to avoid repetition. Therefore, the proof of the theorem is complete. ■

Appendix H
Proof of Lemma 7

Let

$$\begin{aligned} \mathcal{M} &= \{(u_{i_1}, v_{j_1}), (u_{i_2}, v_{j_2}), \ldots, (u_{i_x}, v_{j_x})\}, \\ \mathcal{M}_\star &= \{(u_{i_1}, v_{k_1}), (u_{i_2}, v_{k_2}), \ldots, (u_{i_x}, v_{k_x})\}, \end{aligned}$$

with $k_y \leq a$ for all $y \in \{1, 2, \ldots, x\}$. For obtaining a contradiction, if possible, let there exist an edge $(u_{i_c}, v_b) \in \mathcal{M}'$ with $c \leq x$ and $b > a$. Then there exists a node $v_{k_d}, d \leq x$, such that no edge in $\mathcal{M}'$ has v_{k_d} as one of its endpoints. Let $(u_{i_d}, v_{\alpha_1}) \in \mathcal{M}'$ for some $\alpha_1 < k_d$ (by Lemma 5 (No edge, no path)). Therefore, in the matching $\mathcal{M}_\star$, the node v_{α_1} is an endpoint of some edge $(u_{\beta_1}, v_{\alpha_1})$, else there exists a directed path in $G^\ddagger$ from v_{α_1} to v_{k_d}, namely $v_{\alpha_1} \to u_{i_d} \to v_{k_d}$, contradicting property 3 of $\mathcal{M}_\star$ as required by the statement of the Lemma. There must exist an edge $(u_{\beta_1}, v_{\alpha_2}) \in \mathcal{M}'$, with $\alpha_2 < k_d$ by Lemma 5 (No edge, no path). Hence, the node v_{α_2} is an endpoint of an edge in $\mathcal{M}_\star$, since there exists a directed path $v_{\alpha_2} \to u_{\beta_1} \to v_{\alpha_1} \to u_{i_d} \to v_{k_d}$. This process can be continued indefinitely, contradicting the finiteness of the number of nodes in $\mathcal{U} \cup \mathcal{V}$. Hence, the proof is complete.

Appendix I
Proof of Theorem 5

Claim: For any given $\epsilon \in (0, \alpha)$, there exists a constant $n_0 = n_0(\epsilon)$ such that under any rule for allocating servers to queues, and for all possible values of the parameters $0 < \alpha, q < 1, b \geq 0$, and for infinitely many $n \geq n_0$,

$$\mathbb{P}\left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b\right) \geq (\alpha - \epsilon)^{b+1}(1-q)^{n(b+1)}.$$

Consequently,

$$\liminf_{n \to \infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b\right) \leq (b+1) \log \frac{1}{1-q}.$$

Proof: Fix any $\epsilon \in (0, \alpha)$. By (5), there exists a strictly increasing sequence of natural numbers $\{n_1, n_2, \ldots\}$ such that for all $k \geq 1$,

$$\max_{1 \leq i \leq n_k} p_i^{(n_k)} \in (\alpha - \epsilon, \alpha + \epsilon).$$

Hence, in the system Υ'_{n_k}, there exists a queue $Q_{i_k}^{[n_k]}$ that has the packet-arrival probability at least $\alpha - \epsilon$. Consider the following event that, under *any* scheduling algorithm, implies $\{Q_{i_k}^{[n_k]}(0) > b\}$: for $b+1$ consecutive timeslots before (and including) timeslot 0, there are arrivals to $Q_{i_k}^{[n_k]}$, and all the channels connecting $Q_{i_k}^{[n_k]}$ to the servers are OFF in each of the $b+1$ timeslots. The probability of this event is at least $(\alpha - \epsilon)^{b+1}((1-q)^{n_k})^{b+1}$. Hence, for all $k \geq 1$,

$$\mathbb{P}\left(\max_{1 \leq i \leq n_k} Q_i^{[n_k]}(0) > b\right) \geq (\alpha - \epsilon)^{b+1}((1-q)^{n_k})^{b+1},$$

implying

$$\liminf_{n \to \infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1 \leq i \leq n} Q_i^{[n]}(0) > b\right) \leq (b+1) \log \frac{1}{1-q},$$

and the result is proved.[2] ■

APPENDIX J
PROOF OF THEOREM 6

Claim: For the system with asymmetric arrivals, the iLQF with PullUp algorithm has the property

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge (b+1)\log\frac{1}{1-q}.$$

Proof: From (5), it follows that there exists a natural number n_0 such that for all $n \ge n_0$,

$$\max_{1\le i\le n} p_i^{(n)} \le \frac{1+\alpha}{2} < 1.$$

Consider a sequence of systems $\{\Upsilon''_n\}$. The channel process of the system Υ''_n is identical to that of the system Υ'_n (and Υ_n), but the arrival process is given by $A''^{[n]}_i(t) = A'^{[n]}_i(t)$ for $n < n_0$, and by

$$A''^{[n]}_i(t) = \begin{cases} 1 & \text{with probability } \frac{1+\alpha}{2}, \\ 0 & \text{with probability } 1-\frac{1+\alpha}{2}, \end{cases}$$

for $n \ge n_0$. Further, let $\{A'^{[n]}_i(t) = 1\} \Rightarrow \{A''^{[n]}_i(t) = 1\}$. By Kolmogorov's extension theorem ([24], Appendix A), there exists a probability space on which the above construction is valid and well-defined. Let the two systems Υ'_n and Υ''_n be started with the same initial queue-lengths. Then, by the sample-path-wise dominance property of the iLQF with PullUp algorithm (Lemma 6), it follows that for the two systems,

$$\mathbb{P}\left(\max_{1\le i\le n} Q'^{[n]}_i(0) > b\right) \le \mathbb{P}\left(\max_{1\le i\le n} Q''^{[n]}_i(0) > b\right).$$

The sequence of systems Υ''_n, for $n \ge n_0$ is identical to the one considered in Section IV, in particular a symmetric arrival system. Hence, Theorem 4 implies

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q''^{[n]}_i(0) > b\right) \ge (b+1)\log\frac{1}{1-q}.$$

The result follows by combining the last two inequalities. ■

APPENDIX K
PROOF OF THEOREM 7

Consider the following event which implies $\{Q_1(0) > b\}$ under any scheduling rule: for $m := \left\lceil \frac{b+1}{L} \right\rceil$ consecutive timeslots before (and including) timeslot 0, there are L arrivals per timeslots to Q_1, and all the channels connecting Q_1 to the servers are OFF in each of these timeslots. The probability of this event is $p^m(1-q)^{nm}$, and the result follows.

[2]For a sequence of real numbers $\{a_n\}$, if $a_n \le b$ for infinitely many values of n, then $\liminf_{n\to\infty} a_n \le b$.

APPENDIX L
PROOF OF LEMMA 9

Claim: Fix any $\tilde{p} \in (p, 1/L)$. Define $\Theta_1 := nH(\tilde{p}|p)$ and $\Theta_2 := 3n\tilde{p}(1-q)^{n\tilde{p}}$. Then for the symmetric, bursty, ON-OFF arrivals system, for n large enough, and for all t,

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) > \max_{1\le i\le n} Q_i(t)\right) \le \exp(-\Theta_1) + L\Theta_2.$$

Proof: This proof proceeds by showing that the event $\left(\max_{1\le i\le n} Q_i(t+1) > \max_{1\le i\le n} Q_i(t)\right)$ is a subset of the union of the following two (low-probability) events:

1) In timeslot $t+1$, the number of queues with a nonzero number of arrivals is $\ge n\tilde{p}$ with $\tilde{p} > p$.
2) In a bipartite graph with $2n\tilde{p}$ nodes, where each edge is present with probability q, there does not exist a perfect matching (in fact, a union of L such events).

The union bound then gives the desired result.

The proof in detail: Let $\hat{Q}(t) = m$. By adding packets to the queues if necessary, we ensure $Q_i(t) = m$ for all i. By the sample-path-wise dominance property of the iLQF with PullUp algorithm (Lemma 6), we know that the probability of overflow of the packet-added system is at least as large as that of the original system.

By the Chernoff bound, in any given timeslot $t+1$, we have

$$\mathbb{P}\left(\textstyle\sum_{i=1}^n A_i^{[n]}(t+1) \ge n\tilde{p}L\right) \le \exp\left\{-nH(\tilde{p}|p)\right\}.$$

Consider the event when in the $(t+1)^{th}$ timeslot, the number of queues with a nonzero number of arrivals does not exceed $n\tilde{p}$. Adding packets to queues if necessary, let the number of queues with L packet arrivals in the timeslot $(t+1)$ be *exactly* $n\tilde{p}$. By the sample-path-wise dominance property of the iLQF with PullUp algorithm (Lemma 6), by adding packets to the queues we get a system whose overflow probability is larger than the original system. We focus our analysis on this packet-added system.

First round of service:

With probability $\ge 1 - \exp\{-nH(\tilde{p}|p)\}$, the queue-length distribution at the beginning of the first round of service is:

Queue-length	Number of queues
$m+L$	$n\tilde{p}$
m	$n(1-\tilde{p})$

There exists a perfect matching between the set of the longest queues and the first $n\tilde{p}$ servers with probability at least $1 - 3n\tilde{p}(1-q)^{n\tilde{p}}$, by Lemma 1. Hence, using this matching for $\mathcal{M}_\star$ in Lemma 7 (Use smaller indexed nodes), it follows that the first $n\tilde{p}$ servers are allocated to serve the longest queues in the first round, decreasing their queue-lengths by one each. Thus, the queue-length distribution at the end of the first round of service is:

Queue-length	Number of queues
$m+L-1$	$n\tilde{p}$
m	$n(1-\tilde{p})$

The servers with indices greater than $n\tilde{p}$ are available for allocation in the subsequent rounds, if any.

r^{th} round of service, for $1 < r \le L$:
Let the queue-length distribution at the beginning of r^{th} round of service be:

Queue-length	Number of queues
$m + L - (r-1)$	$n\tilde{p}$
m	$n(1-\tilde{p})$

Then, by an argument exactly as in the case $r = 1$, it follows that with probability at least $1 - 3n\tilde{p}(1-q)^{n\tilde{p}}$, the servers with indices in the set $\{(r-1)n\tilde{p}+1, (r-1)n\tilde{p}+2, \ldots, rn\tilde{p}\}$ are allocated to serve the longest queues, so that the queue-length distribution at the end of the r^{th} round of service is:

Queue-length	Number of queues
$m + L - r$	$n\tilde{p}$
m	$n(1-\tilde{p})$

The servers with indices greater than $rn\tilde{p}$ are available for allocation in the subsequent rounds, if any.

Further, by Lemma 7 (Use smaller indexed nodes), the event of finding a bipartite perfect matching between the set of longest queues and the set of servers indexed $(r-1)n\tilde{p}+1$ to $rn\tilde{p}$ (in the r^{th} round) is conditionally independent of all previous rounds, conditioned on the existence of perfect matchings in the earlier rounds, such that for all $s < r$, the round s resulted in allocation of a (perfect) matching between the set of longest queues and the set of servers indexed $(s-1)n\tilde{p}+1$ to $sn\tilde{p}$. Hence, considering the first L rounds of service,

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) \le \max_{1\le i\le n} Q_i(t)\right) \ge (1-e^{-\Theta_1})(1-\Theta_2)^L.$$

Therefore,

$$\begin{aligned}
&\mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) > \max_{1\le i\le n} Q_i(t)\right)\\
&\quad= 1 - \mathbb{P}\left(\max_{1\le i\le n} Q_i(t+1) \le \max_{1\le i\le n} Q_i(t)\right)\\
&\quad\le 1 - (1-\exp(-\Theta_1))(1-\Theta_2)^L\\
&\quad\le 1 - (1-\exp(-\Theta_1))(1-L\Theta_2)\\
&\quad= \exp(-\Theta_1) + L\Theta_2 - L\Theta_2\exp(-\Theta_1)\\
&\quad\le \exp(-\Theta_1) + L\Theta_2\\
&\quad= \exp(-nH(\tilde{p}|p)) + L\Theta_2.
\end{aligned}$$

Thus, the proof is complete. ■

Appendix M
Proof of Lemma 10

Claim: For the symmetric, bursty, ON-OFF arrivals system implementing the iLQF with PullUp algorithm, there exists a constant $k = k(L,p) = \left\lceil \frac{3}{1-pL} \right\rceil$ such that for all n large enough, for all $m > 0$ and all T,

$$\mathbb{P}(\hat{Q}(T+k) < m | \hat{Q}(T) = m) \ge \tfrac{1}{2}.$$

Proof: To simplify notation, let $T = 0$ and $\hat{Q}(0) = m$ in a queuing system $\underline{Q}$. Consider a queuing system $\underline{Q'}$ where $Q'_i(0) = m$ for all $\underline{i}$, implying $Q'_i(0) \ge Q_i(0)$ for $\underline{\text{all}}$ i, and this property continues to hold for all further timeslots if the arrivals and the channel realizations are identical for the two systems (Lemma 6). Hereafter in this proof, we will not make references to the system $\underline{Q}$.

Fix $\tilde{p} \in (p, 1/L)$, say $\tilde{p} = (p+1/L)/2$ for concreteness. The probability that in a given timeslot there are, in all, more than $n\tilde{p}$ queues with a nonzero number of arrivals is upper bounded by $\exp(-nH(\tilde{p}|p))$ (by the Chernoff bound). Hence, by union bound, the probability that there are no more than $n\tilde{p}$ queues with a nonzero number of arrivals in *any* of the k consecutive timeslots from 1 to k is at least $1 - k\exp(-nH(\tilde{p}|p))$. We condition the rest of the proof on this (high probability) event, and further (if necessary), in every timeslot, we artificially add packets to queues that did not receive packets to enforce the condition that the number of queues receiving L packets is *exactly* $n\tilde{p}$.

Timeslot 1:
After packet arrivals, there are $n\tilde{p}$ queues each with a length = $m + L$, and the rest with length = m. Following an analysis similar to that in the proof of Lemma 9, there is service for at least L rounds with probability at least $(1 - 3n(1-\tilde{p})(1-q)^{n(1-\tilde{p})})^L$, so that after L rounds of service, all the queues have a length $= m$ and there exist $n(1-\tilde{p}L)$ unallocated servers, with indices $\{n(1-\tilde{p}L)+1, n(1-\tilde{p}L)+2, \ldots, n\}$. In the $(L+1)^{th}$ round of service, there exists a matching of cardinality $n(1-\tilde{p}L)$ between the set of unallocated servers and the set of longest queues (i.e. the set of all the queues) with probability at least $1 - 3n(1-\tilde{p}L)(1-q)^{n(1-\tilde{p}L)}$, so that (with high probability) the queue-length distribution at the end of the first timeslot is:

Queue-length	Number of queues
m	$n\tilde{p}L$
$m-1$	$n(1-\tilde{p}L)$

Timeslot $r, 1 < r \le k$:
We show that if, at the beginning of timeslot r the queue-length profile is

Queue-length	Number of queues
m	x
$m-1$	$n-x$

then with high probability, at the end of the timeslot r (i.e., at the beginning of the timeslot $r+1$) the queue-length profile is given by (or, can be obtained by adding dummy packets):

Queue-length	Number of queues
m	$(x-n\epsilon)^+$
$m-1$	$n-(x-n\epsilon)^+$

Since $x \le n$, the above claim implies that with high probability, at the end of at most $k = \lceil 1/\epsilon \rceil$ timeslots, the maximum queue-length in the system decreases from m to $m-1$ (or less). Further, by Lemma 2 (Forward jump bound), if the maximum queue-length is less than m at a timeslot before k, then with high probability it remains less than m until the end of timeslot k. Hence, the above claim yields essentially the desired result with a careful choice of ϵ.

To this end, consider the queue-length distribution after arrivals, i.e., at the beginning of the first round of service, given by

Queue-length	Number of queues
$m+L$	y
$m+L-1$	$n\tilde{p}-y$
m	$x-y$
$m-1$	$n-x-n\tilde{p}+y$

Here we have $y \le n\tilde{p}$, because we have conditioned on the (high-probability) event that in any timeslot, the number of queues with a nonzero number of arrivals is no more than $n\tilde{p}$.

Fix any $\delta \in (0, 1-L\tilde{p})$, say $\delta = (1-L\tilde{p})/3$ for concreteness.

CASE 1: $y \ge n\delta$.

By Lemma 1, the set of the first y servers and the y longest queues have a perfect matching with probability at least $3y(1-q)^y \ge 3n\delta(1-q)^{n\delta}$, where the inequality is valid for n large. Thus, by Lemma 7 (Use smaller indexed nodes), this matching (if it exists) is used to serve the y longest queues, regardless of the channel realizations of the servers indexed from $y+1$ to n.

Thus, at the end of the first round of service, the queue-length profile (with high probability) is given by

Queue-length	Number of queues
$m+L-1$	$n\tilde{p}$
m	$x-y$
$m-1$	$n-x-n\tilde{p}+y$

Further, all of the servers with indices from $y+1$ to n are available for allocation, and their channel realizations are conditionally independent of any allocations in the first round of service.

At the end of the next $L-1$ rounds of service, following a similar argument to the one above, with high probability, the queue-length profile is given by

Queue-length	Number of queues
m	$n\tilde{p}+x-y$
$m-1$	$n-x-n\tilde{p}+y$

Further, all of the servers with indices from $y+(L-1)n\tilde{p}+1$ to n are available for allocation, and their channel realizations are conditionally independent of any allocations in the first round of service.

Next, the (last) $n-y-(L-1)n\tilde{p}$ servers are available for service, and the set of longest queues is of cardinality $n\tilde{p}+x-y$. Hence, with high probability, after one more (i.e., $(L+1)^{th}$) round of service, the number of queues of length m is at most $(x+n\tilde{p}-y-n+y+(L-1)n\tilde{p})^+ = (x-n(1-L\tilde{p}))^+$.

CASE 2: $y < n\delta$.

By Lemma 1, the set of the first $n\delta$ servers and the y longest queues have a perfect queue matching with probability at least $3n\delta(1-q)^{n\delta}$, where the inequality is valid for n large. (Lemma 1 holds for a bipartite graph with equal number of nodes on either side of the cut, but we can imagine adding $n\delta-y$ dummy queue-nodes for the purpose of finding a perfect matching, and allocating servers to the (real) queues according to its projection on the set of (real) queues, which results in a perfect queue matching.) Thus, by Lemma 7 (Use smaller indexed nodes), this matching (if it exists) is used to serve the y longest queues, regardless of the channel realizations of the servers indexed from $n\delta+1$ to n.

Thus, at the end of the first round of service, the queue-length profile (with high probability) is given by

Queue-length	Number of queues
$m+L-1$	$n\tilde{p}$
m	$x-y$
$m-1$	$n-x-n\tilde{p}+y$

Further, all of the servers with indices from $n\delta+1$ to n are available for allocation, and their channel realizations are conditionally independent of any allocations in the first round of service.

At the end of the next $L-1$ rounds of service, following an argument similar to CASE 1 above, with high probability, the queue-length profile is given by

Queue-length	Number of queues
m	$n\tilde{p}+x-y$
$m-1$	$n-x-n\tilde{p}+y$

Further, all of the servers with indices from $n\delta+(L-1)n\tilde{p}+1$ to n are available for allocation, and their channel realizations are conditionally independent of any allocations in the first round of service.

Next, the (last) $n-n\delta-(L-1)n\tilde{p}$ servers are available for service, and the set of longest queues is of cardinality $n\tilde{p}+x-y$. Hence, with high probability, after one more (i.e., $(L+1)^{th}$) round of service, the number of queues of length m is at most $(x+n\tilde{p}-y-n+n\delta+(L-1)n\tilde{p})^+ \le (x-n(1-L\tilde{p}-\delta))^+$.

We have $\tilde{p} = (p+1/L)/2$ and $\delta = (1-L\tilde{p})/3$. Hence, with the choice

$$\epsilon = 1 - L\tilde{p} - \delta = \frac{2(1-L\tilde{p})}{3} = \frac{1-Lp}{3},$$

the proof is complete. ∎

APPENDIX N
PROOF OF THEOREM 8

Claim: For a system with symmetric, bursty, ON-OFF arrivals implementing the iLQF with PullUp algorithm, for all $b \ge 0$, and for $\tilde{p} = (1/L+p)/2$,

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge \left\lceil \frac{b+1}{L} \right\rceil \min\left(\tilde{p}\log\frac{1}{1-q}, H(\tilde{p}|p)\right) > 0.$$

Proof: As a result of Lemmas 9 and 10, and adding dummy packets if necessary, we obtain a system whose queue-length process is a stochastic process with the following properties:

1) In a given timeslot, the maximum queue-length either remains unchanged, or increases by an amount L. If the timeslot index is a multiple of k (from Lemma 10), then the maximum queue-length may also decrease by 1, in addition to remaining unchanged or increasing by an amount L.
2) The probability that in a given timeslot, the queue-length increases by an amount L, is at most
$$2\max\left(\exp(-nH(\tilde{p}|p)), 3nL\tilde{p}(1-q)^{n\tilde{p}}\right),$$

for all $\tilde{p} \in (p, 1/L)$. In particular, $\tilde{p} = (1/L + p)/2$ is a valid choice.

3) Let k be the parameter from Lemma 10. For an integer T, if the maximum queue-length in the system at the end of the timeslot T is positive, then over the next k timeslots, it decreases by at least 1 with probability at least $1/2$.

By the sample-path-wise dominance property of the iLQF with PullUp algorithm (Lemma 6), the probability of finite buffer overflow of the modified queue-length process is an upper bound on the probability of finite buffer overflow in the original system. Hence, we analyze the modified process that has the three properties described above.

By an argument similar to the one presented in Appendix C, the queue-length process is stable (the Lyapunov function that equals the maximum queue-length works here too). Arguing along the same lines as the Markov-chain coupling proof in the Appendix C, we get for all $\tilde{p} \in (p, 1/L)$, in particular for $\tilde{p} = (1/L + p)/2$:

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge \left\lceil \frac{b+1}{L} \right\rceil \min\left(\tilde{p}\log\frac{1}{1-q}, H(\tilde{p}|p)\right),$$

completing the proof since the right hand side is strictly positive for all $b \geq 0$. ■

Appendix O
Proof of Theorem 9

Claim: For the system with symmetric arrivals with bounded support, under the iLQF with PullUp algorithm, for all $b \geq 0$,

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge \left\lceil \frac{b+1}{L} \right\rceil \min\left(J\log\frac{1}{1-q}, \zeta\right) > 0.$$

Proof: We give a sketch of the proof here. The main idea, as before, is:
(1) To upper-bound the probability of an "upward" transition, i.e., $\mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(t+1) > \max_{1\le i\le n} Q_i^{[n]}(t)\right)$ by $\exp(-n\alpha)$ for some $\alpha > 0$. In case this upward transition occurs, we bound the magnitude of increment in $\max_{1\le i\le n} Q_i^{[n]}(t)$ by L, the maximum number of arrivals to a queue in the timeslot t.
(2) To lower-bound (by a constant, say $1/2$) the probability of a "downward" transition, i.e.,

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(t+k) < \max_{1\le i\le n} Q_i^{[n]}(t) \mid \max_{1\le i\le n} Q_i^{[n]}(t) > 0\right).$$

Then we use an argument similar to that in the proof of Theorem 4 to conclude the desired rate-function result. Since this argument has been spelled out in detail in the earlier proof, we only sketch a proof of the upper-bound on the upward transition. (The downward transition proof and the calculation of bounds on the stationary distribution, similar to those in Appendix C, are routine calculations and have been omitted.)

Fix any timeslot t and any $\delta > 0$. Consider the event that the empirical probability mass function of the arrivals in the timeslot $t + 1$ lies outside the set F_{ϵ_0}. By Sanov's theorem (Theorem 2.1.10 in [21]), the probability of the aforementioned event is at most $\exp(-n(\zeta - \delta))$ for n large enough. We condition the rest of the proof on the (high-probability) event that the empirical probability mass function of the arrivals in timeslot $t + 1$ is in the set F_{ϵ_0}.

Consider the case when $\max_{1\le i\le n} Q_i^{[n]}(t) = m$, and (adding packets if necessary) $Q_i^{[n]}(t) = m$ for all i. After arrivals in timeslot $t + 1$, let the empirical probability mass function of the number of arrivals be given by $a = [a_0, a_1, \ldots, a_L] \in F_{\epsilon_0}$. Under the iLQF with PullUp algorithm, in the first round of service, the na_L queues at length $m + L$ are served (if possible). For n large enough, the probability of finding a matching that can serve the na_L queues with the first na_L servers is at least $1-3na_L(1-q)^{na_L}$. In the next round of service, the $n(a_L+a_{L-1})$ queues at length $m + L - 1$ are served with the next $n(a_L + a_{L-1})$ servers (i.e., servers indexed from $na_L + 1$ to $n(a_L + a_{L-1})$) with probability at least $1 - 3n(a_L + a_{L-1})(1 - q)^{n(a_L+a_{L-1})}$, and so on, for the L rounds when each of the queues is of length m and $n\sum_i ia_i \le n(1 + \lambda_p)/2$ servers are used (allocated), thus resulting in a feasible allocation. Using the standard arguments about the principle of the smallest exponent, the probability that at the end of timeslot $t+1$, the longest queue is of length at least $m + 1$ is at most $L \times 3na_L(1 - q)^{na_L}$ for n large enough. Since the smallest value of a_L in the set F_{ϵ_0} is J, and since $a_L \le 1$, we have (for n large enough, and by the union bound)

$$\mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(t+1) > \max_{1\le i\le n} Q_i^{[n]}(t)\right) \le e^{-n(\zeta-\delta)} + 3nL(1-q)^{nJ}.$$

Since the above result holds for any prespecified $\delta > 0$ for n large enough, using calculations similar to those in Appendix C, in the limit as $n \to \infty$, we have

$$\liminf_{n\to\infty} \frac{-1}{n} \log \mathbb{P}\left(\max_{1\le i\le n} Q_i^{[n]}(0) > b\right) \ge \left\lceil \frac{b+1}{L} \right\rceil \min\left(J\log\frac{1}{1-q}, \zeta\right) > 0.$$

This completes (an outline of) the proof. ■

Appendix P
Matching the Transition Probabilities

Fix an integer t, define $t' := k_0(t + 1)$, and for all j such that $-\tilde{B}^\star(t) \le j \le k_0$, consider the set of sample-paths

$$\mathcal{C}_j := \left\{\omega \in \Omega : \tilde{B}^\star(t)(\omega) + j = \max_{1\le i\le n}\left(R_i(t'-1) + A_i(t') - \sum_{\ell=1}^{n} X_{i\ell}(t')Y_{i\ell}(t')\right)^+(\omega)\right\},$$

where Ω is the space over which all the random variables are defined. Each $\omega \in \Omega$ belongs to precisely one of the sets $\mathcal{C}_j$. The sets $\mathcal{C}_j$ are functions of the time-index, which we suppress for ease of notation.

The set $\mathcal{C}_j$ is the set of all sample paths where in the system $\underline{R}$, the maximum queue length changes (increases) by j over the k_0 timeslots after timeslot t, up to and including the timeslot t'. (Note that j can be negative.) This follows because in k_0 timeslots, the maximum queue-length can increase by an amount at most k_0, and can possibly decrease to 0, therefore $\mathbb{P}(\mathcal{C}_j)$ can possibly be nonzero only for $-\tilde{B}^\star(t) \le j \le k_0$.

In the course of the proof, we need to make references to the set

$$\Xi := \arg\max_{1\le i\le n}\left(R_i(t'-1)+A_i(t')-\sum_{\ell=1}^{n}X_{i\ell}(t')Y_{i\ell}(t')\right)^+$$

which contains the indices of the longest queues after service-completion in the timeslot t', before any extra packets are added. Note that Ξ is a function of ω, which we suppress for ease of notation.

The drain property implies that for $\tilde{B}^\star(t) > 0$,

$$\Theta_t := \mathbb{P}(\mathcal{C}_{-1})+\mathbb{P}(\mathcal{C}_{-2})+\cdots+\mathbb{P}(\mathcal{C}_{-\tilde{B}^\star(t)}) \ge \frac{1}{2}.$$

We now define the procedure for adding extra packets to the queuing system. There are two cases: $\tilde{B}^\star(t) = 0$ and $\tilde{B}^\star(t) > 0$. For $1 \le r \le k_0$, define $\beta_r := \binom{k_0}{r}\alpha_n^r - \mathbb{P}(\mathcal{C}_r)$. Note that $\beta_r \ge 0$ for n large enough. We only consider the case $\sum_{i=1}^{k_0}\beta_i > 0$, which corresponds to at least one of the inequalities (11) being strict. The other case can be handled similarly.

Case 1**:** $\tilde{B}^\star(t) > 0$.

There are four sub-cases:

Sub-case C1. $\mathbb{P}(\mathcal{C}_{-1}) \ge 1/2, \Theta_t - 1/2 \ge \sum_{r=1}^{k_0}\beta_r$.

Define $\beta_0 := \Theta_t - 1/2 - \sum_{r=1}^{k_0}\beta_r \ge 0$. For each $2 \le j \le \tilde{B}^\star(t)$, define a random variable $D_j(t')$, independent of all other random variables, as follows:

$$\mathbb{P}(D_j(t') = j+r) = \frac{\beta_r}{\sum_{i=0}^{k_0}\beta_i}, \quad 0 \le r \le k_0.$$

Define the random variable $D_1(t')$, independent of all other random variables, as

$$\mathbb{P}(D_1(t') = r) = \begin{cases} (1/2)/\mathbb{P}(\mathcal{C}_{-1}), & r = 0, \\ \frac{\beta_{r-1}}{\sum_{i=0}^{k_0}\beta_i}\cdot\frac{\mathbb{P}(C_{-1})-1/2}{\mathbb{P}(C_{-1})}, & 1 \le r \le k_0+1. \end{cases}$$

If $\omega \in \mathcal{C}_{-j}$ for some $j \ge 1$, then add $D_j(t')$ packets to the smallest-indexed element of the set Ξ.

Sub-case C2. $\mathbb{P}(\mathcal{C}_{-1}) \ge 1/2, \Theta_t - 1/2 < \sum_{r=1}^{k_0}\beta_r$.

For each $2 \le j \le \tilde{B}^\star(t)$, define a random variable $E_j(t')$, independent of all other random variables, as follows:

$$\mathbb{P}(E_j(t') = j+r) = \frac{\beta_r}{\sum_{i=1}^{k_0}\beta_i}, \quad 1 \le r \le k_0.$$

Define the random variable $E_1(t')$, independent of all other random variables, as

$$\mathbb{P}(E_1(t') = r) = \begin{cases} (1/2)/\mathbb{P}(\mathcal{C}_{-1}), & r = 0, \\ \frac{\beta_{r-1}}{\sum_{i=1}^{k_0}\beta_i}\cdot\frac{\mathbb{P}(C_{-1})-1/2}{\mathbb{P}(C_{-1})}, & 2 \le r \le k_0+1. \end{cases}$$

Define the random variable $E_0(t')$, independent of all other random variables, as

$$\mathbb{P}(E_0(t') = r) = \begin{cases} \frac{\mathbb{P}(C_0)+\Theta_t-1/2-\sum_{i=1}^{k_0}\beta_i}{\mathbb{P}(C_0)}, & r = 0, \\ \frac{\sum_{i=1}^{k_0}\beta_i-(\Theta_t-1/2)}{\mathbb{P}(C_0)}\cdot\frac{\beta_r}{\sum_{i=1}^{k_0}\beta_i}, & 1 \le r \le k_0. \end{cases}$$

If $\omega \in \mathcal{C}_{-j}$ for some $j \ge 0$, then add $E_j(t')$ packets to the smallest-indexed element of the set Ξ.

Sub-case C3. $\mathbb{P}(\mathcal{C}_{-1}) < 1/2, \Theta_t - 1/2 \ge \sum_{r=1}^{k_0}\beta_r$.

Define $\beta_{-1} := 1/2 - \mathbb{P}(\mathcal{C}_{-1})$, and $\beta_0 := \Theta_t - 1/2 - \sum_{r=1}^{k_0}\beta_r \ge 0$.

For each $2 \le j \le \tilde{B}^\star(t)$, define a random variable $F_j(t')$, independent of all other random variables, as follows:

$$\mathbb{P}(F_j(t') = j+r) = \frac{\beta_r}{\sum_{i=-1}^{k_0}\beta_i}, \quad -1 \le r \le k_0.$$

If $\omega \in \mathcal{C}_{-j}$ for some $j \ge 2$, then add $F_j(t')$ packets to the smallest-indexed element of the set Ξ.

Sub-case C4. $\mathbb{P}(\mathcal{C}_{-1}) < 1/2, \Theta_t - 1/2 < \sum_{r=1}^{k_0}\beta_r$.

Define $\beta_{-1} := 1/2 - \mathbb{P}(\mathcal{C}_{-1})$.

For each $2 \le j \le \tilde{B}^\star(t)$, define a random variable $G_j(t')$, independent of all other random variables, as follows:

$$\mathbb{P}(G_j(t') = j+r) = \begin{cases} \frac{\beta_{-1}}{\Theta_t-\mathbb{P}(\mathcal{C}_{-1})}, & r = -1, \\ \frac{\beta_r}{\sum_{i=1}^{k_0}\beta_i}\cdot\frac{\Theta_t-1/2}{\Theta_t-\mathbb{P}(\mathcal{C}_1)}, & 1 \le r \le k_0. \end{cases}$$

Define the random variable $G_0(t')$, independent of all other random variables, as

$$\mathbb{P}(G_0(t') = r) = \begin{cases} \frac{\beta_r}{\sum_{i=1}^{k_0}\beta_i}\cdot\frac{\sum_{i=1}^{k_0}\beta_i-(\Theta_t-1/2)}{\mathbb{P}(\mathcal{C}_0)}, & 1 \le r \le k_0, \\ \frac{\mathbb{P}(\mathcal{C}_0)+\Theta_t-1/2-\sum_{i=1}^{k_0}\beta_i}{\mathbb{P}(\mathcal{C}_0)}, & r = 0. \end{cases}$$

If $\omega \in \mathcal{C}_{-j}$ for some $j \ge 0, j \ne 1$ then add $G_j(t')$ packets to the smallest-indexed element of the set Ξ.

Case 2**:** $\tilde{B}^\star(t) = 0$.

As before, define a random variable $H_0(t')$ with

$$\mathbb{P}(H_0(t') = r) = \begin{cases} \frac{\beta_r}{\mathbb{P}(\mathcal{C}_0)}, & 1 \le r \le k_0, \\ \frac{\mathbb{P}(\mathcal{C}_0)-\sum_{i=1}^{k_0}\beta_i}{\mathbb{P}(\mathcal{C}_0)}, & r = 0. \end{cases}$$

If $\omega \in \mathcal{C}_0$, then add $H_0(t')$ packets to the smallest-indexed element of the set Ξ.

We now analyze the four sub-cases of Case 1, and the Case 2. The analysis of the different cases is similar, but we present it for the sake of completeness.

Case 1, **Sub-case C1:** For each $1 \le j \le \tilde{B}^\star(t)$, the random variables $D_j(t')$ are valid and well-defined, and non-negative. Thus, we add only a non-negative number of packets to one of the queues in the system. For $1 \le r \le k_0$,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m+r \mid \tilde{B}^\star(t) = m > 0) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \mathbb{P}(C_{-1}) \cdot \mathbb{P}(D_1(t') = r+1) \\
&\qquad + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \mathbb{P}(D_j(t') = j+r) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \mathbb{P}(\mathcal{C}_{-1}) \cdot \frac{\mathbb{P}(\mathcal{C}_{-1}) - 1/2}{\mathbb{P}(\mathcal{C}_{-1})} \cdot \frac{\beta_r}{\sum_{i=0}^{k_0} \beta_i} \\
&\qquad + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \frac{\beta_r}{\sum_{i=0}^{k_0} \beta_i} \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \frac{\beta_r}{\sum_{i=0}^{k_0} \beta_i} \cdot \underbrace{\left(\sum_{j=1}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) - \frac{1}{2} \right)}_{=\Theta_t - 1/2 = \sum_{i=0}^{k_0} \beta_i} \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \beta_r \\
&\quad= \binom{k_0}{r} \alpha_n^r.
\end{aligned}$$

Further,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m-1 \mid \tilde{B}^\star(t) = m > 0) \\
&\qquad\qquad = \mathbb{P}(\mathcal{C}_{-1}) \cdot \frac{1/2}{\mathbb{P}(\mathcal{C}_{-1})} = 1/2,
\end{aligned}$$

and for any $r > 1$,

$$\mathbb{P}(\tilde{B}^\star(t+1) = m-r \mid \tilde{B}^\star(t) = m > 0) = 0.$$

Case 1, **Sub-case C2:** For each $1 \le j \le \tilde{B}^\star(t)$, the random variables $E_j(t')$ are valid and well-defined, and non-negative. Since $\Theta_t - 1/2 < \sum_{r=1}^{k_0} \beta_r$, in order to prove that $E_0(t')$ is a valid random variable, we need to show that $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r \ge 0$ and $\mathbb{P}(\mathcal{C}_0) > 0$. We show that $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r \ge 0.3$, which implies $\mathbb{P}(\mathcal{C}_0) > 0$.

We have

$$\sum_{i=1}^{k_0} \beta_i \le \sum_{i=1}^{k_0} \binom{k_0}{i} \alpha_n^i = (1+\alpha_n)^{k_0} - 1 \le 0.1.$$

Further, $\mathbb{P}(\mathcal{C}_0) + \Theta_t + \sum_{i=1}^{k_0} \mathbb{P}(\mathcal{C}_i) = 1$, and

$$\sum_{i=1}^{k_0} \mathbb{P}(\mathcal{C}_i) \le \sum_{i=1}^{k_0} \binom{k_0}{i} \alpha_n^i = (1+\alpha_n)^{k_0} - 1 \le 0.1.$$

Thus, $\mathbb{P}(\mathcal{C}_0) + \Theta_t \ge 0.9$, or $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 \ge 0.4$, implying $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r \ge 0.3$. Since $\Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r < 0$, we have $\mathbb{P}(\mathcal{C}_0) \ge 0.3 > 0$.

Next, we have for $1 \le r \le k_0$,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m+r \mid \tilde{B}^\star(t) = m > 0) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \sum_{j=0}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \mathbb{P}(E_j(t') = j+r) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \mathbb{P}(\mathcal{C}_{-1}) \cdot \frac{\mathbb{P}(\mathcal{C}_{-1}) - 1/2}{\mathbb{P}(\mathcal{C}_{-1})} \cdot \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \\
&\qquad + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \\
&\qquad + \mathbb{P}(\mathcal{C}_0) \cdot \frac{\sum_{i=1}^{k_0} \beta_i - (\Theta_t - 1/2)}{\mathbb{P}(\mathcal{C}_0)} \cdot \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \\
&\quad= \mathbb{P}(C_r) + \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \left(\sum_{j=1}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) - \frac{1}{2} \right. \\
&\qquad \left. + \sum_{i=1}^{k_0} \beta_i - \Theta_t + 1/2 \right) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \beta_r, \quad \text{since } \Theta_t = \sum_{j=1}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \\
&\quad= \binom{k_0}{r} \alpha_n^r.
\end{aligned}$$

Further,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m-1 \mid \tilde{B}^\star(t) = m > 0) \\
&\qquad\qquad = \mathbb{P}(\mathcal{C}_{-1}) \cdot \frac{1/2}{\mathbb{P}(\mathcal{C}_{-1})} = 1/2,
\end{aligned}$$

and for any $r > 1$,

$$\mathbb{P}(\tilde{B}^\star(t+1) = m-r \mid \tilde{B}^\star(t) = m > 0) = 0.$$

Case 1, **Sub-case C3:** For each $2 \le j \le \tilde{B}^\star(t)$, the random variables $F_j(t')$ are valid and well-defined, and non-negative. For $1 \le r \le k_0$,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m+r \mid \tilde{B}^\star(t) = m > 0) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \mathbb{P}(F_j(t') = j+r) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \frac{\beta_r}{\sum_{i=-1}^{k_0} \beta_i} \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \frac{\beta_r}{\sum_{i=-1}^{k_0} \beta_i} \cdot \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \\
&\quad\overset{(a)}{=} \mathbb{P}(\mathcal{C}_r) + \beta_r \\
&\quad= \binom{k_0}{r} \alpha_n^r.
\end{aligned}$$

Here the step (a) holds because $\sum_{i=-1}^{k_0} \beta_i = \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j})$. Further, applying the above argument verbatim (until step (a)) for the case $r = -1$,

$$\mathbb{P}(\tilde{B}^\star(t+1) = m-1 \mid \tilde{B}^\star(t) = m > 0) = \mathbb{P}(\mathcal{C}_{-1}) + \beta_{-1} = \frac{1}{2},$$

and for any $r > 1$,

$$\mathbb{P}(\tilde{B}^\star(t+1) = m - r \mid \tilde{B}^\star(t) = m > 0) = 0.$$

Case 1, **Sub-case C4:** For each $2 \le j \le \tilde{B}^\star(t)$, the random variables $G_j(t')$ are valid and well-defined, and non-negative. (To see that they are valid random variables, note that $\beta_{-1} = 1/2 - \mathbb{P}(\mathcal{C}_{-1}) \le \Theta_t - \mathbb{P}(\mathcal{C}_{-1})$, and $\Theta_t - \mathbb{P}(\mathcal{C}_{-1}) > \Theta_t - 1/2 \ge 0$.)

In order to prove that $G_0(t')$ is a valid random variable, we need to show that $\mathbb{P}(C_0) \ge \sum_{i=1}^{k_0} \beta_i - (\Theta_t - 1/2)$ and $\mathbb{P}(\mathcal{C}_0) > 0$. We show that $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r \ge 0.3$, which implies $\mathbb{P}(\mathcal{C}_0) > 0$ (since $\Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r < 0$).

We have

$$\sum_{i=1}^{k_0} \beta_i \le \sum_{i=1}^{k_0} \binom{k_0}{i} \alpha_n^i = (1+\alpha_n)^{k_0} - 1 \le 0.1.$$

Further, $\mathbb{P}(\mathcal{C}_0) + \Theta_t + \sum_{i=1}^{k_0} \mathbb{P}(\mathcal{C}_i) = 1$, and

$$\sum_{i=1}^{k_0} \mathbb{P}(\mathcal{C}_i) \le \sum_{i=1}^{k_0} \binom{k_0}{i} \alpha_n^i = (1+\alpha_n)^{k_0} - 1 \le 0.1.$$

Thus, $\mathbb{P}(\mathcal{C}_0) + \Theta_t \ge 0.9$, or $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 \ge 0.4$, implying $\mathbb{P}(\mathcal{C}_0) + \Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r \ge 0.3$. Since $\Theta_t - 1/2 - \sum_{r=1}^{k_0} \beta_r < 0$, we have $\mathbb{P}(\mathcal{C}_0) \ge 0.3 > 0$.

Next, for $1 \le r \le k_0$,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m + r \mid \tilde{B}^\star(t) = m > 0) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \mathbb{P}(G_j(t') = j + r) \\
&\qquad + \mathbb{P}(\mathcal{C}_0) \cdot \mathbb{P}(G_0(t') = j + r) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \cdot \frac{\Theta_t - 1/2}{\Theta_t - \mathbb{P}(\mathcal{C}_{-1})} \\
&\qquad + \mathbb{P}(\mathcal{C}_0) \cdot \frac{\sum_{i=1}^{k_0} \beta_i - (\Theta_t - 1/2)}{\mathbb{P}(\mathcal{C}_0)} \cdot \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \\
&\quad\overset{(a)}{=} \mathbb{P}(\mathcal{C}_r) + \frac{\beta_r}{\sum_{i=1}^{k_0} \beta_i} \left(\sum_{i=1}^{k_0} \beta_i \right) \\
&\quad= \binom{k_0}{r} \alpha_n^r,
\end{aligned}$$

where the step (a) holds because $\Theta_t - \mathbb{P}(\mathcal{C}_{-1}) = \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j})$, and the final step holds because $\beta_r = \binom{k_0}{r} \alpha_n^r - \mathbb{P}(\mathcal{C}_r)$.

Further,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = m - 1 \mid \tilde{B}^\star(t) = m > 0) \\
&\quad= \mathbb{P}(\mathcal{C}_{-1}) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \mathbb{P}(G_j(t') = j - 1) \\
&\quad= \mathbb{P}(\mathcal{C}_{-1}) + \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j}) \cdot \frac{\beta_{-1}}{\Theta_t - \mathbb{P}(\mathcal{C}_{-1})} \\
&\quad\overset{(a)}{=} \mathbb{P}(\mathcal{C}_{-1}) + \beta_{-1} \\
&\quad\overset{(b)}{=} 1/2,
\end{aligned}$$

where the step (a) holds because $\Theta_t - \mathbb{P}(\mathcal{C}_{-1}) = \sum_{j=2}^{\tilde{B}^\star(t)} \mathbb{P}(\mathcal{C}_{-j})$, and the step (b) holds because $\beta_{-1} = 1/2 - \mathbb{P}(\mathcal{C}_{-1})$. Finally, for any $r > 1$,

$$\mathbb{P}(\tilde{B}^\star(t+1) = m - r \mid \tilde{B}^\star(t) = m > 0) = 0.$$

Case 2**:** To show that $H_0(t')$ is a valid and well-defined random variable, we need to show that $\mathbb{P}(\mathcal{C}_0) > 0$ and $\mathbb{P}(\mathcal{C}_0) - \sum_{i=1}^{r} \beta_i \ge 0$. We have

$$\sum_{i=1}^{k_0} \beta_i \le \sum_{i=1}^{k_0} \binom{k_0}{i} \alpha_n^i = (1+\alpha_n)^{k_0} - 1 \le 0.1.$$

Further, since $\mathbb{P}(\mathcal{C}_i) \le \binom{k_0}{i} \alpha_n^i$ for $1 \le i \le k_0$ and $\mathbb{P}(\mathcal{C}_0) + \sum_{i=1}^{k_0} \mathbb{P}(\mathcal{C}_i) = 1$, we have $\mathbb{P}(\mathcal{C}_0) \ge 0.9 > 0$, and $\mathbb{P}(\mathcal{C}_0) - \sum_{i=1}^{r} \beta_i \ge 0.8 > 0$. Also, the random variable $H_0(t')$ is nonnegative.

Next, for $1 \le r \le k_0$,

$$\begin{aligned}
&\mathbb{P}(\tilde{B}^\star(t+1) = r \mid \tilde{B}^\star(t) = 0) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \mathbb{P}(\mathcal{C}_0) \cdot \mathbb{P}(H_0(t') = r) \\
&\quad= \mathbb{P}(\mathcal{C}_r) + \mathbb{P}(\mathcal{C}_0) \cdot \frac{\beta_r}{\mathbb{P}(\mathcal{C}_0)} \\
&\quad= \binom{k_0}{r} \alpha_n^r,
\end{aligned}$$

since $\beta_r = \binom{k_0}{r} \alpha_n^r - \mathbb{P}(\mathcal{C}_r)$.

Thus, under the specified packet addition method, the transition probabilities of $\tilde{B}^\star(t')$ obey (10) and (11) with equality. Clearly with the specified packet-additions, the process $\underline{\tilde{B}}(t)$ is Markovian, because given $\underline{\tilde{B}}(t)$, $\{\underline{\tilde{B}}(u)\}_{u>t}$ is independent of $\{\underline{\tilde{B}}(s)\}_{s<t}$.