Stability of Decentralized Queueing Networks Beyond Complete Bipartite Cases

Hu Fu[1(✉)], Qun Hu[1], and Jia'nan Lin[2]

[1] ITCS, Shanghai University of Finance and Economics, Shanghai, China
fuhu@mail.shufe.edu.cn, 2019212804@sufe.edu.cn
[2] Rensselaer Polytechnic Institute, Troy, NY, USA
linj21@rpi.edu

Abstract. Gaitonde and Tardos [3,4] recently studied a model of queueing networks where queues compete for servers and re-send returned packets in future rounds. They quantify the amount of additional processing power that guarantees a decentralized system's stability, both when the queues adapt their strategies from round to round using no-regret learning algorithms, and when they are patient and evaluate the utility of a strategy over long periods of time.

In this paper, we generalize Gaitonde and Tardos's model and consider scenarios where not all servers can serve all queues (i.e., the underlying graph is an incomplete bipartite graphs) and, further, when packets need to go through more than one server before their completions (i.e., when the underlying graph is a DAG). For the bipartite case, we obtain bounds comparable to those by Gaitonde and Tardos, with the factor slightly worse in the patient queueing model. For the more general multi-layer systems, we show that straightforward generalizations of the queues' utilities and servers' priority rules in [3] may lead to unbounded gaps between centralized and decentralized systems when the queues use no regret strategies. We give new utilities and service priority rules that are aware of the queue lengths, and show that these suffice to restore the bounded gap between centralized and decentralized systems.

Keywords: Queueing networks · Price of anarchy · No-regret learning dynamics

1 Introduction

A recurrent theme in algorithmic game theory is to analyze systems operated by decentralized, strategic agents, in comparison with those run by centralized authorities. Since Koutsoupias and Papadimitriou [6] introduced the concept of *Price of Anarchy*, it has been applied and studied in various games such as routing in congestion games [8], network resource allocation [5], auctions [2],

Supported by the Fundamental Research Funds for the Central Universities of China. Part of the work was done when the third author was visiting Shanghai University of Finance and Economics.

K. A. Hansen et al. (Eds.): WINE 2022, LNCS 13778, pp. 96–114, 2022.
https://doi.org/10.1007/978-3-031-22832-2_6

among many other settings. Recently, Gaitonde and Tardos [3,4] introduced a routing game in queueing systems, where queues compete for servers each round, and packets not processed successfully in one round go back to their queues and have to be re-sent in the future. Unlike most games previously studied, in such systems, the strategies and outcomes of one round have carryover effect in future rounds, introducing intricate dependencies among the rounds. Gaitonde and Tardos developed bicriteria bounds that quantify the loss of efficiency due to decentralized strategic behaviors in such systems in two settings: in [3], the queues evaluate the utility of their strategies from round to round, and adopt no-regret learning algorithms in their routing decisions; in [4], the queues are "patient", and fix their strategies over long periods of time over which they evaluate their performances.

In both [3] and [4], all servers can process requests from all queues, and a packet leaves the system once it is processed by a server. These are simplifying modelling assumptions: in many queueing systems, each queue's packets may only be processed by certain servers, and a packet may need to go through more than one server before leaving the system. In this work, we model such added complexities by seeing the queues and servers as nodes of a directed acyclic graph (DAG). A queue can send requests to a server only if it has an outgoing edge to the server. Packets arrive at given rates to nodes with no incoming edges, and leave the system when they are successfully processed by servers with no outgoing edges; nodes with both incoming and outgoing edges are both servers and queues—after it successfully processes a packet, the packet joins its queue and waits to be sent to the next server. The case considered by Gaitonde and Tardos [3] corresponds to complete bipartite graphs. We examine whether and how their results generalize to more general settings.

Our Results. We first characterize networks that can be stable under a centralized policy, where stability roughly means that the number of packets accumulated in the system is bounded. As in [3], the main lesson of the characterization is that it is without loss of generality for a centralized policy to fix for each queue a distribution and sample a server from this distribution at each time step, independently of the history and all other happenings in the system. For bipartite graphs (Theorem 2) our proof takes a perspective arguably simpler than that in [3], and this perspective is instrumental in showing the conditions for general DAGs, which are considerably more involved.

We then consider decentralized systems where queues use no-regret learning strategies. For general bipartite graphs, we show that the bound in [3] generalizes with minor modification. We inherit much of the proof framework of [3], including a potential function argument and various apparatus for analyzing the random processes, although in the key step of the argument where one uses the no regret property to bound the number of "old" packets processed over a time window, our proof has to take into account the underlying graph structure, and makes a connection with the *dual* form of the conditions for centralized stability. The eventual stability conditions we give (Theorem 3) when queues use no-regret learning strategies is also expressed as a scaled dual form of the centralized stability conditions. As a consequence, the main bicriteria comparison result in [3]

extends to general bipartite graphs: a decentralized system is stable if it can be made stable under a centralized policy with the arrival rates doubled. Interestingly, the dual variables in our decentralized stability condition take values from a smaller range ($\{0, 1\}$) than in the centralized stability condition (where they may be any nonnegative numbers). For complete bipartite graphs, it can be shown that even for the centralized stability condition, the dual variables need only take $0, 1$ values. In this sense, our results suggest that the gap between the two conditions tends to be smaller for incomplete bipartite graphs.

Networks with more than one layer of servers are even more interesting. A major conclusion reached in [3] is that a server's rule of priority for packets simultaneously sent to it is crucial for the system's stability. In the complete bipartite graphs, it was shown that, if the servers pick a packet uniformly at random, then no said bicriteria bound can be given; in contrast, the bicriteria result was obtained when servers are assumed to prioritize older packets. Another important factor in the model is the queues' utilities: it was assumed in [3] that a queue collects utility of 1 if its packet is successfully cleared by a server, and 0 otherwise. Our results for general bipartite graphs inherit both these modelling assumptions. However, for graphs of even three layers, we give an example showing that no finite bound of the bicriteria form can be obtained if one directly extends the utility and the priority rule from [3]. Intuitively, in order for the system not to lose too much efficiency, information on the underlying graph is important when there are multiple layers: a server with strong processing capacity may be poorly connected in the next layer, and myopic strategies easily send too many packets to such a server. Therefore, the queues' utilities need to incorporate more information for their strategies to better align with the system's stability; on the other hand, if they are fed with too much global information, the difference could blur between centrally controlled systems and decentralized ones. A natural question to raise is whether it is possible to incentivize the queues using only local information so that their selfish behaviors do not hurt the system efficiency too much. We answer this question in the affirmative, showing that the *lengths* of queues in the neighboring nodes provide just this information. We propose a new service priority rule, under which the servers prioritize packets from the *longest* queues. We also propose new utility functions for queues, with which a queue of length L_i, when it sends a packet to a server j whose own queue is of length L_j, obtains utility $L_i - L_j$ if the packet is successfully processed by j. In particular, with this new utility function, a queue never sends its packets to a server whose current queue is longer than itself. We show that when the new service priority rule and utilities are adopted, the bicriteria result is restored: a queueing system is stable with queues that use no-regret strategies as long as it is stable under a centralized policy even when the packet arrival rates are doubled.

Lastly, we extend the model with patient queues to bipartite graphs. Gaitonde and Tardos [4] showed for complete bipartite graphs that, when queues are patient, with appropriately defined long-term utilities, a Nash equilibrium always exists, and a system is stable under any Nash equilibrium as long as it is stable under a centralized policy even with $\frac{e}{e-1}$ times the original arrival rates. To this end, they developed elaborate tools for computing the long-term utilities given the queues' strategies. These tools generalize straightforwardly in general bipartite graphs, but the delicate deformation argument in the proof of their

bicriteria result does not easily generalize. Our proof again makes use of the dual form of the condition for centralized stability, which provides a matching between the fastest growing queues in an equilibrium and servers.

In the full version of this paper, we also consider two other variants of the problem: in one model, whether a server can process a packet is not determined by which queue the packet is from, but is an intrinsic property of the packet; in the other one, the arrival of packets at each queue is not from a Bernoulli distribution, but is controlled by an adversarial, as in the model of Borodin et al. [1]. In both variants, we show that the bicriteria results persist when queues use no-regret strategies. Lastly, we give a tighter bicriteria result for the model in [3], where the underlying graph is a complete bipartite graph. We show that a queueing system is stable with queues that play no-regret strategies as long as it is stable under a centralized policy even when the k-th largest packet arrival rate is increased by a factor $\frac{2k-1}{k}$ for each k.

Further Related Works. We refer to Gaitonde and Tardos [3,4] for pointers to related works in algorithmic game theory and no regret learning. Sentenac et al. [9] considered the same model as in [3] but when queues use *cooperative* learning. When incentives are removed from the problem, they show that the queues can essentially learn the necessary system parameters and reach a stable outcome as long as the system is stable under a centralized policy.

2 Preliminaries

2.1 Queue-G Model

A *Queue-G Model* is a $G = (V, E, \boldsymbol{\lambda}, \boldsymbol{\mu})$, where $(V = S_1 \cup S_2 \cup S_3, E)$ constitutes a directed acyclic graph, and $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the arrival and processing rates on the nodes. A node i with no incoming edge is a *source*, and has an *arrival rate* λ_i. For each i, $\lambda_i \in (0, 1)$. S_1 denotes the set of sources. All the other nodes are *servers*, and each server j has a *processing rate* μ_j. A server with no outgoing edge is a *terminal*. S_3 denotes the set of terminals. The set of non-terminal server nodes is $S_2 \coloneqq V - S_1 - S_3$. An edge $(i, j) \in E$ means that node i can send packets to node j. For $i \in S_1 \cup S_2$, we denote by $N^{\text{out}}(i) \coloneqq \{j \in V : (i, j) \in E\}$ the set of out-neighbors of i, and for a server i, we denote by $N^{\text{in}}(i) \coloneqq \{j \in V : (j, i) \in E\}$ the set of in-neighbors of i.

Let Q_t^i denote the number of packets at node i at the beginning of time step t. For all $i \in V$, $Q_0^i = 0$. At each time step t, the following events happen, in two phases:

(I) Packet sending: each node i with $Q_t^i > 0$ chooses a server j from $N^{\text{out}}(i)$ and sends to j the oldest packet (with the earliest timestamp) in i's queue. In a centralized system, a central authority dictates for each node if and where to send its packet at each time step; in a decentralized system, each node strategizes over this decision.

(II) Packet arrival and processing: at each source $i \in S_1$, a packet with timestamp t arrives with probability λ_i; each server $i \in S_2 \cup S_3$, if it receives any packet in phase (I), chooses one such packet according to some *service priority rule* to process, and succeeds with probability μ_i. The arrivals of packets at each source and the successes of their processing at each server are all mutually independent events. A packet cleared by server $j \in S_2$ joins the queue of server j; a packet cleared by a server in S_3 leaves the system. A packet not chosen by or not successfully processed by a server goes back to the node that sends it. It follows that any $i \in S_3$ has $Q_t^i = 0$ at any time step t.

Gaitonde and Tardos [3] considered a special case of the Queue-G Model, where there are no non-terminal servers and every source can send packets to every server, i.e., $S_2 = \emptyset$ and $E = S_1 \times S_3$, and the service priority rule at each server is to choose the oldest packet (breaking ties arbitrarily).[1]

We refer to this special case as the *Queue-CB Model* ("CB" for complete bipartite).

If we only have $S_2 = \emptyset$ (and allow any $E \subseteq S_1 \times S_3$), we have the *Queue-B Model*.

2.2 Stability and No Regret Learning

Let $Q_t \coloneqq \sum_{i \in V} Q_t^i$ be the total number of packets in the queueing system at the start of time step t. We inherit from [3] the notion of stability:

Definition 1. *Under some scheduling policy (either with a central authority or with queues strategizing), a queueing system is* strongly stable *if for any $a > 0$, there is a constant C_a only related to a, such that $\mathbb{E}[(Q_t)^a] \leq C_a$ for all t. A queueing system is* ***almost surely strongly stable*** *if with probability 1, the following event happens: for any $a > 0$, $Q_t = o(t^a)$.*

Gaitonde and Tardos [3] showed that if a queueing system is strongly stable, then it is almost surely strongly stable. We therefore focus on showing strong stability, and often refer to a strongly stable system simply as stable.

The following theorem by Pemantle and Rosenthal [7], also used in [3], is the workhorse for showing stability.

Theorem 1 ([7]). *Let $X_0, \cdots, X_n$ be nonnegative random variables. If there are constants $b, c, d > 0$ and $p > 2$ such that $X_0 \leq b$ and, for all n,*

$$\mathbb{E}(|X_{n+1} - X_n|^p \mid X_0, \cdots, X_n) \leq d; \tag{1}$$

$$X_n > b \quad \Rightarrow \quad \mathbb{E}(X_{n+1} - X_n \mid X_0, \cdots, X_n) \leq -c, \tag{2}$$

then for any $a \in (0, p-1)$, there is $C = C(p, a, b, c, d)$ such that $\mathbb{E}(X_n)^a < C$ for all n.

[1] For ease of presentation, we made minor changes from Gaitonde and Tardos [3]'s model, in the order of packet sending and packet arrival. It is easy to see that the difference is negligible for the analysis of the system's stability, which is an asymptotic quality, to be defined below.

We refer to (2) as the *negative drift condition*, and (1) as the *bounded jump condition*.

We now introduce utilities of queues, as defined in [3]. The utility of a queue at time step t is the number of packets cleared from this queue at time step t. Let $a_i(t)$ denote the server that node i chooses at time step t. (A node i may not choose any server, in which case we let $a_i(t) = 0$ and we set $\mu_0 = 0$.) Let $\mathcal{F}_t$ denote the history of the system up to the beginning of time step t. We use $u_t^i(a_i(t), a_{-i}(t))|\mathcal{F}_t)$ to denote the utility of node i when node i chooses server $a_i(t)$ and the other nodes choose $a_{-i}(t)$, given history $\mathcal{F}_t$. We should specify the content of a history: $\mathcal{F}_t$ only includes information on which packets, up to time step t, were cleared and the age of the currently oldest packet in each node, but does not include the queue size Q_t^i. This makes sure that, for the k-th packet in node i that is cleared at time step t, the time difference between its arrival and that of the $(k+1)$-st packet is independent of the history $\mathcal{F}_{t'}$ for all $t' < t$, and obeys the geometric distribution with parameter λ_i.

Lastly, we define the *regret* of a node i up to time w as the difference between its utility in a real sample path and what it could have achieved by always playing a best fixed action in hindsight.

Definition 2. *For a time window from time step $t_0 - w$ to $t_0 - 1$, the* regret *of queue i for actions $a_i(t_0 - w), \ldots, a_i(t_0 - 1)$ is*

$$\mathrm{Reg}_i(w, t_0) \coloneqq \max_{j : j \in N^{\mathrm{out}}(i)} \sum_{t=t_0-w}^{t_0-1} u_t^i(j, a_{-i}(t)|\mathcal{F}_t) - \sum_{t=t_0-w}^{t_0-1} u_t^i(a_i(t), a_{-i}(t)|\mathcal{F}_t).$$

Note that, in this definition, the utility obtained by playing the best fixed strategy is evaluated using "real" histories ($\mathcal{F}_t$'s) observed under the actual actions taken by the node. It does not use counterfactual histories generated by playing the fixed strategy. We often drop the parameter t_0 when it is clear from the context.

Definition 3. *Given fixed $\delta \in (0, 1)$, queue i's scheduling policy is* no regret *if, for any time window from time step $t_0 - w$ to $t_0 - 1$, with probability at least $1 - \delta$, $\mathrm{Reg}_i(w, t_0) \le \varphi_\delta(w)$, where $\varphi_\delta(w) = o(w)$ may depend only on δ and the number of nodes in the queueing system.*

3 Bipartite Queueing Systems

In this section we derive necessary and sufficient conditions for the existence of a centralized policy that stabilizes a queueing system in a bipartite graph (a Queue-B model). We then give a sufficient condition that guarantees the stability of such systems when all queues adopt no-regret strategies. For the special case when the underlying graph is complete bipartite, our conditions degenerate to the ones given by Gaitonde and Tardos [3].

3.1 Stability Conditions Under Centralized Policies

A Queue-B model as defined in Sect. 2 simply consists of n queues on one side and m servers on the other. Server j is able to clear a packet from queue i if and only if there is an edge between the two. It is easy to see that a centralized policy never benefits from sending packets from two queues to a same server in a single time step, as the server picks up only one of them. Therefore, with loss of generality, the routing dictated by a centralized policy at any step gives a matching of the queues to the servers. (Some queues may be asked not to send their packets, and some servers may be allowed to be idle for that round.) It is less clear whether a centralized policy benefits from making intricate use of the history when it decides on the matching at each step. It turns out, for the system to be stable (Definition 1), it is without loss of generality to consider history oblivious centralized policy, which samples a matching from a fixed distribution over matchings from step to step. A Queue-B model can be stable under any centralized policy if and only if it can be stable under such a policy. This is the essence of the following theorem.

Recall that a fractional matching matrix $P \in [0,1]^{n\times m}$ is such that $\sum_j P_{ij} \leq 1$ for all $i \in [n]$ and $\sum_i P_{ij} \leq 1$ for all $j \in [m]$.

Theorem 2. *Given a Queue-B model with n queues and m servers, with arrival rates $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_n)$ and processing rates $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$, there is a centralized policy under which the system is stable if and only if there exists a fractional matching matrix $P \in [0,1]^{n\times m}$, such that $P\boldsymbol{\mu} \succ \boldsymbol{\lambda}$, where $\succ$ denotes element-wise greater than.*

Sufficiency of the condition is a consequence of Birkhoff-von Neumann theorem. The argument of necessity makes use of the observation that, conditioning on any event in the system, the expected routing decision made by a centralized policy is expressible as a fractional matching matrix. One may as well condition on the event that all queues have arrivals considerably larger than the expectations, which occurs with constant probability. This part of the argument is arguably simpler than the proof in [3], and makes possible the more involved proof for more general graphs (Theorem 4). The proofs missing due to lack of space are deferred to the full version of this paper.

Before moving on to decentralized Queue-B models, we derive a dual form of the conditions in Theorem 2. The dual form plays a crucial role in our analysis of the systems' stability under no-regret policies.

Lemma 1. *Given a Queue-B model with arrival rates $\boldsymbol{\lambda}$ and processing rates $\boldsymbol{\mu}$, the following two conditions are equivalent:*

(1) There is a fractional matching matrix P such that $P\boldsymbol{\mu} \succ \boldsymbol{\lambda}$.
(2) For any $\boldsymbol{\alpha} \in \mathbb{R}^n_+$, there is a matching matrix $M \in \{0,1\}^{n\times m}$, such that $\boldsymbol{\alpha}^\top M\boldsymbol{\mu} > \boldsymbol{\alpha}^\top\boldsymbol{\lambda}$.

The lemma is an application of Farkas' lemma. The proof of this lemma is deferred to the full version. It is worth pointing out that, when the underlying

graph is a complete bipartite graph, it suffices to have the condition (2) satisfied for only $\boldsymbol{\alpha} \in \{0,1\}^n$. This difference plays a role in the contrast between complete and incomplete bipartite graphs when the system is decentralized, as we explain in the next section.

3.2 Stability Conditions Under Decentralized, No-Regret Policies

In this section we give conditions under which, in a queueing system on an incomplete bipartite graph (the Queue-B model), if all queues use no-regret strategies, the system is stable. Our conditions are most easily comparable with the dual form of centralized stability conditions stated in Lemma 1. When the underlying graph is a complete bipartite graph, the conditions are identical to those by Gaitonde and Tardos [3], as we discuss below. The technique in this part is largely inherited from [3], although our proof reveals an interesting connection between the dual form of stability conditions and key steps in the proof. The sufficient condition is the following:

Assumption 1. *There is a constant $\beta > 0$, such that for any $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_n) \in \{0,1\}^n$, there is a matching matrix M, such that $\frac{1}{2}(1-\beta)\boldsymbol{\alpha}^\top M\boldsymbol{\mu} > \boldsymbol{\alpha}^\top \boldsymbol{\lambda}$.*

A quick comparison between this and dual condition in Lemma 1 suggests that, if one has a Queue-B model which can be made stable by a centralized policy, then, doubling its processing capabilities guarantees its stability when the queues use no-regret strategies. Note though that the range of $\boldsymbol{\alpha}$ is much smaller in Assumption 1 ($\{0,1\}^n$) than in Lemma 1 ($\mathbb{R}_+^n$). For complete bipartite graphs, this difference vanishes (see remark following Lemma 1), but in general bipartite graphs, this difference is real. This suggests that in incomplete bipartite graphs, the gap between centralized and decentralized systems tends to be smaller than in complete bipartite graphs.

Theorem 3. *If a Queue-B model queueing system satisfies Assumption 1, and queues use no-regret learning strategies with $\delta = \frac{\beta}{128n}$, then the system is strongly stable.*

Following Gaitonde and Tardos [3], we introduce a potential function with the intention to apply Theorem 1 to its square root. The *age* of a packet that arrives in the system at time t_1 is defined to be $t_2 - t_1$ at time t_2. Let T_t^i be the age of the oldest packet in queue i at time step t, and let $\boldsymbol{T}$ be the vector $(T_t^1, \cdots, T_t^n)$. Note that Q_t^i, the length of the queue, is at most T_t^i. For a positive integer $\tau > 0$, define

$$\Phi_\tau(\mathbf{T}_t) := \sum_{i:T_t^i \geq \tau} \lambda_i (T_t^i - \tau).$$

The potential function Φ is defined as

$$\Phi(\mathbf{T}_t) := \sum_{\tau=1}^{\infty} \Phi_\tau(\mathbf{T}_t) = \sum_{\tau=1}^{\infty} \sum_{i:T_t^i \geq \tau} \lambda_i (T_t^i - \tau) = \frac{1}{2} \sum_{i=1}^{n} \lambda_i T_t^i (T_t^i - 1).$$

We analyze the system by dividing the time steps into windows of length w each, for some large enough w. Let $Z_\ell := \sqrt{\Phi\left(\mathbf{T}_{\ell \cdot w}\right)}$ be the square root of the potential function at the beginning of the ℓ-th window. The main work lies in showing that $(Z_\ell)_\ell$ satisfies the conditions of Theorem 1, which implies $\mathbf{E}[Z_\ell^a]$ is bounded for any $a > 0$. This in turn implies that $\mathbf{E}[(\sum_i T_t^i)^a]$ is bounded, and so is $\mathbf{E}[(Q_t)^a]$.

Lemma 2. *[Negative drift condition.] Denote by $\lambda_{(n)}$ the minimum element of $\boldsymbol{\lambda}$.*
Let $b = \frac{w}{\sqrt{2\lambda_{(n)}}} \max\left(\frac{8}{\beta}\left(\sum_{i=1}^n \lambda_i\right), 16n^2\right)$, $c = -\frac{\sqrt{2\lambda_{(n)}}\beta w}{64}$. Then $Z_0 = 0 \leq b$ and, for all ℓ,

$$Z_\ell > b \quad \Rightarrow \mathbf{E}\left[Z_{\ell+1} - Z_\ell \mid Z_0, \cdots, Z_\ell\right] \leq -c.$$

Lemma 3. *[Bounded jump condition.] For each even integer $p \geq 2$, there is a constant d_p, such that for all ℓ*

$$\mathbf{E}\left[|Z_{\ell+1} - Z_\ell|^p \mid Z_0, \cdots, Z_\ell\right] \leq d_p.$$

Lemma 3 is identical to the corresponding part in Gaitonde and Tardos [3], and we omit its proof. The main difference between our proof and [3] is in the proof of the negative drift condition (Lemma 2). We present here the key steps of our proof, and the rest is deferred to full version.

Following Gaitonde and Tardos [3], for a given $\tau > 0$, we say a packet is *τ-old* if its age is at least τ at time step $\ell \cdot w$, i.e., if its arrival time is no later than $\ell w - \tau$. Let J_τ be the set of queues which have τ-old packets at time step $(\ell+1)\cdot w$. For a queue i, if by time step $(\ell+1)\cdot w$, it still has packets that arrived before time step $\ell \cdot w$, let $\tau_i = \max_{\tau > 0 : J_\tau \ni i} \tau$ be the age of the oldest packet in queue i; otherwise, set $\tau_i = 0$. Let N_τ^i be the number of τ-old packets cleared from queue i during the time window from time step $\ell \cdot w$ to $(\ell+1)\cdot w$. Similarly, for a server j, let L_τ^j be the number of τ-old packets cleared by server j during this time window. Next, define $N_\tau = \sum_{i \in [n]} N_\tau^i = \sum_{j \in [m]} L_\tau^j$ as the number of τ-old packets cleared during this time window. Lastly, let C_t^j be the indicator variable for server j succeeding in processing a packet if it picks one up.

Lemma 4. *For any $\tau > 0$ and $\epsilon > 0$, if $\sum_{t=\ell \cdot w}^{(\ell+1)\cdot w - 1} C_t^j \geq (1-\epsilon)\mu_j w$ for each j, then $N_\tau \geq \frac{1-\epsilon}{1-\beta} \sum_{i \in J_\tau} \lambda_i w - \sum_{i=1}^n \mathrm{Reg}_i(w, (\ell+1)\cdot w)$.*

Proof. Any queue $i \in J_\tau$ has a τ-old packet throughout the time window. For a server j which can serve queue i, consider the counterfactual utility i may gain during this time window by sending a request to j at each step. Let X_{jt}^τ be the indicator variable for the event that *some queue* (which may not be i) sends a τ-old packet to server j at time step t. Then at any time step t when $X_{jt}^\tau = 0$, server i's packet would have been picked up by server j had i sent a request, because no other packet sent to j is τ-old, so the packet from i has priority. Recall that C_t^j is the indicator variable for server j succeeding in processing a packet if it picks one up. So queue i would have gained utility 1 at time t by

sending a request to j if $C_t^j = 1$ and $X_{jt}^\tau = 0$. Over the time window, queue i's counterfactual utility could have been $\sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1}(1 - X_{jt}^\tau)$. Note that queue i's actual utility is N_τ^i, so by definition of regret, we have

$$N_\tau^i \geq \sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} C_t^j(1 - X_{jt}^\tau) - \mathrm{Reg}_i(w, (\ell+1)\cdot w).$$

On the other hand, whenever $X_{jt}^\tau S_t^j = 1$, server j successfully clears a τ-old packet. Therefore, $L_\tau^j = \sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} C_t^j X_{jt}^\tau$. Then, for a pair of queue $i \in J_\tau$ and server j that can serve i, we have

$$N_\tau^i + L_\tau^j \geq \sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} C_t^j - \mathrm{Reg}_i(w, (\ell+1)\cdot w). \tag{3}$$

Now we are ready to apply Assumption 1. Let $\boldsymbol{\alpha}$ be the indicator vector for the set $J_\tau \subseteq [n]$, i.e., $\alpha_i = 1$ if $i \in J_\tau$, and $\alpha_i = 0$ otherwise. By Assumption 1, we can find a matching matrix M_τ such that $\frac{1}{2}(1-\beta)\boldsymbol{\alpha}^\top M_\tau \boldsymbol{\mu} > \boldsymbol{\alpha}^\top \boldsymbol{\lambda}$. Let U_τ be the edge set such that $(i,j) \in U_\tau \Leftrightarrow M_\tau(i,j) = 1$. Then,

$$\frac{1}{2}(1-\beta) \sum_{(i,j)\in U_\tau} \mu_j > \sum_{i\in J_\tau} \lambda_i. \tag{4}$$

Now, we are ready to give a lower bound for N_τ:

$$\begin{aligned}
2N_\tau &= \sum_{i=1}^n N_\tau^i + \sum_{j=1}^m L_\tau^j \geq \sum_{(i,j)\in U_\tau} (N_\tau^i + L_\tau^j) \\
&\geq \sum_{(i,j)\in U_\tau} \left(\sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} C_t^j - \mathrm{Reg}_i(w, (\ell+1)\cdot w) \right) && (5) \\
&\geq \sum_{(i,j)\in U_\tau} (1-\epsilon)\mu_j w - \sum_{i=1}^n \mathrm{Reg}_i(w, (\ell+1)\cdot w) && (6) \\
&\geq \frac{2(1-\epsilon)}{1-\beta} \sum_{i\in J_\tau} \lambda_i w - \sum_{i=1}^n \mathrm{Reg}_i(w, (\ell+1)\cdot w), && (7)
\end{aligned}$$

where the second inequality uses (3), and the last inequality uses (4).

We sketch the rest of the proof, and all details are deferred to the full version. When the servers' realized processing capacities are close to their expectations (as in the condition of Lemma 4) and when the queues' regret are small (which should happen with high probability by assumption), the lower bound given by Lemma 4 on N_τ implies a lower bound on the decrease in the potential function due to packet clearing (Lemma 5). We can further bound the increase in the

potential function due to aging over the time interval. (Lemma 6). We define an event A, specified in the full version of this paper, which happens with high probability, and under which all of these events (of concentration and no regret) happen.

Recall that τ_i is the age of the oldest packet in queue i at time step $(\ell+1)\cdot w$, where the age is measured by time step $\ell \cdot w$. Let $\boldsymbol{\tau} = \{\tau_1, \cdots, \tau_n\}$.

Lemma 5. *Under* event A, $\Phi(\mathbf{T}_{\ell\cdot\mathbf{w}}) - \Phi(\boldsymbol{\tau}) \geq \frac{1-2\epsilon}{1-\beta} \sum_{i=1}^{n} \lambda_i \tau_i w$.

Lemma 6. *Under* event A, $\Phi(\mathbf{T}_{(\ell+\mathbf{1})\cdot\mathbf{w}}) - \Phi(\boldsymbol{\tau}) \leq \sum_{i=1}^{n} \lambda_i \tau_i w + \frac{1}{2} \sum_{i=1}^{n} \lambda_i w^2$.

With a small probability, event A does not happen, and it is relatively straightforward to upper bound the increase in the potential in this case. (Most pessimistically, no packets is cleared during the time window and T_t^i in each queue grows by w.)

Lemma 7. *If event A does not happen,* $\Phi(\mathbf{T}_{(\ell+\mathbf{1})\cdot\mathbf{w}}) - \Phi(\mathbf{T}_{\ell\cdot\mathbf{w}}) \leq \sum_{i=1}^{n} \lambda_i T_{\ell\cdot w}^i w + \frac{1}{2} \sum_{i=1}^{n} \lambda_i w^2$.

Lemma 2 follows from combining Lemma 5, 6 and 7.

4 Queueing Systems with Multiple Layers

In this section we study queueing systems where packets or tasks may need to go through more than one servers before their completions. After a packet is successfully processed by an intermediate server, it immediately joins the queue forming at their server, waiting to be sent to the next server. In Sect. 4.1, we give sufficient and necessary conditions for such a queueing system to be stable under a centralized policy. In Sect. 4.2, we show that, when one extends the utility and service priority rules from Gaitonde and Tardos [3]'s model to such networks, it is impossible to obtain a PoA result comparable to Theorem 3. In Sect. 4.2, we introduce new utilities and service priority rules that are aware of local queue lengths, and show that they suffice to restore conditions for stability under decentralized, no-regret strategies.

4.1 Stability Under Centralized Policies

As we reasoned for the bipartite case, it never benefits a central planner to send packets from more than one queues to the same server in a single time step, therefore, it is without loss of generality to consider policies under which, at each time step, the edges along which packets are sent from a set of vertex-disjoint paths. (Note that one such path need not start from a source or end at a terminal.) In general, at each step this set of paths may be sampled from a distribution that depends on the history. As in the bipartite case, the following characterization of stable systems shows it without loss of generality to let this

distribution be the same from step to step, regardless of what happened in the past. The proof though is considerably more involved than in the bipartite case.[2]

Theorem 4. *Given a Queue-G model* $(V, E, \boldsymbol{\lambda}, \boldsymbol{\mu})$*, the following statements are equivalent.*

1. *There exists a centralized policy under which the system is stable.*
2. *The following linear system is feasible:*

$$\lambda_i < \sum_j z_{ij}\mu_j, \qquad \forall i \in S_1; \tag{8}$$

$$\mu_i \sum_j z_{ji} < \sum_j z_{ij}\mu_j, \qquad \forall i \in S_2 \text{ with } \prod_{j \in N^{\text{in}}(i)} z_{ji} > 0; \tag{9}$$

$$\sum_j z_{ij} \leq 1, \qquad \forall i \in S_1 \cup S_2; \tag{10}$$

$$\sum_j z_{ji} \leq 1, \qquad \forall i \in S_2 \cup S_3; \tag{11}$$

$$z_{ij} = 0, \qquad \forall (i,j) \notin E; \tag{12}$$

$$z_{ij} \geq 0, \qquad \forall (i,j) \in E. \tag{13}$$

3. *The following linear system in* $(f_{i\pi})_{i \in S_1, \pi \in \Pi}$ *is feasible, where* Π *is the set of paths from a node in* S_1 *to a node in* S_3*:*

$$\sum_{i \in S_1} \sum_{y \in N^{\text{out}}(x)} \sum_{\pi \in \Pi : \pi \ni (x,y)} \frac{f_{i\pi}}{\mu_y} \leq 1, \forall x \in S_1 \cup S_2; \tag{14}$$

$$\sum_{i \in S_1} \sum_{y \in N^{\text{in}}(x)} \sum_{\pi \in \Pi : \pi \ni (y,x)} f_{i\pi} \leq \mu_x, \forall x \in S_2 \cup S_3; \tag{15}$$

$$\sum_{\pi \in \Pi} f_{i\pi} > \lambda_i, \forall i \in S_1; \tag{16}$$

$$\sum_{i \in S_1} \sum_{\pi \in \Pi : \exists y, (y,x) \in \pi} f_{i\pi} = \sum_{i \in S_1} \sum_{y \in N^{\text{out}}(x)} \sum_{\pi \in \Pi : \pi \ni (x,y)} f_{i\pi}, \forall x \in S_2; \tag{17}$$

$$f_{i\pi} = 0, \forall i \in S_1, \pi \in \Pi \, s.t. \; i \text{ not on } \pi. \tag{18}$$

We relegate the proof of the theorem to the full version. The fact that constraints (8)–(13) being feasible implies the stability of a centralized policy is a relatively straightforward consequence of a generalization of Birkhoff-von Neumann theorem. Proving the other direction is considerably more involved than for the bipartite case, and it is for this purpose that we introduce the third condition in Theorem 4. We show that the feasibility of constraints (14)–(18) implies the feasibility of constraints (8)–(13), then we show that a system for which constraints (14)–(18) are not feasible cannot be stable.

[2] It is relatively easy to extend the argument in Theorem 2 to show the necessity of the conditions in Theorem 4, except for the strictness of the signs in (8) and (9).

Again we give a dual form of conditions for centralized stability, which are central to our analysis of the systems' stability under no-regret policies. Its proof can be found in the full version of this paper.

Definition 4. *A vertex-disjoint path (one such path need not start from a source or end at a terminal) is a collection of edges. Any two edges in the path can't have the same head or the same tail.*

Lemma 8. *Given a Queue-G model* $(V, E, \boldsymbol{\lambda}, \boldsymbol{\mu})$ *with* n *sources and* m *servers, the following two conditions are equivalent:*

(1) The linear system of the second statement in Theorem 4 is feasible.
(2) For any $\boldsymbol{\alpha} \in \{\mathbb{R}_+^{n+m} \mid \alpha_i = 0 \text{ if } i \in S_3\}$*, there is a vertex-disjoint path set* U*, such that* $\sum_{(i,j)\in U}(\alpha_i - \alpha_j)\mu_j > \sum_{i\in S_1} \alpha_i \lambda_i$.

4.2 Decentralized Multi-Layer Networks

System Failure with Myopic Queues. In a queueing system with multiple layers, (i.e., when $S_2 \neq \emptyset$), a natural extension of the utility in bipartite systems as defined in Sect. 2 is to let a queue earn utility 1 at a time step if one of its packet is successfully processed by the server it is sent to. The hope is that when all queues focus on getting their packets processed by the *next* server, the system runs relatively efficiently. Unfortunately, as the following example shows, when the queues run no-regret strategies on such utilities, they may be too short-sighted for the decentralized system to have performance comparable to a centralized one, even if one increases the processing capacities by any constant factor.

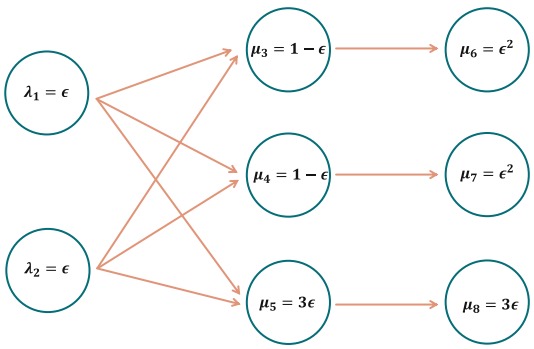

Fig. 1. A queueing system with two layers of servers. A centralized policy sending packets from both sources to server 5 makes the system stable, but the two sources may find it a no-regret strategy to send requests to servers 3 and 4, respectively.

Example 1. The system shown in Fig. 1 is stable under a centralized policy. One feasible solution to the linear system given in Theorem 4 is $z_{15} = z_{25} = 0.4$, $z_{58} = 1$, with all other coordinates of $\mathbf{z}$ set to 0. It is not difficult to see that, if both queue 1 and queue 2 send their requests to server 3 and server 4, respectively, they play no-regret strategies, but the system is unstable because packets accumulate at servers 3 and 4. The phenomenon persists even when the processing capacities are increased by a factor of $\frac{1}{\epsilon}$.

Stability with Queue Length Aware Utilities. Example 1 suggests that the instantaneous, local feedback is not enough to align the queues' interests with the system's efficiency. In this section we show that, when we incorporate one other piece of local information, the *queue lengths*, into the queues' utilities and the service priority rule, we can recover the bicriteria results we showed for single-layer systems in Sect. 3.

Recall that Q_t^i denotes the length of queue i at time t. Our new utility for queue $i \in S_1 \cup S_2$ for sending a request to server j at time t is

$$u_t^i(j, a_{-i}(t) \mid \mathcal{F}_t) = \begin{cases} Q_t^i - Q_t^j, & \text{if the packet sent to } j \text{ is successfully processed;} \\ 0, & \text{otherwise.} \end{cases}$$

Note that this utility function immediately implies that it is never in a queue's interest to send a request to a server with a queue longer than itself. Also recall that $Q_t^j = 0$ for any $j \in S_3$ at any time t. The history $\mathcal{F}_t$ now includes information on which packets have been cleared and the queue size Q_t^i.[3]

We also change the servers' priority rules to preferring requests from longer queues. With the new utilities and service priority rules, the sufficient condition we obtain for decentralized stability is:

Assumption 2. *There is a $\beta > 0$ such that for any $\boldsymbol{\alpha} \in \{\mathbb{R}_+^{n+m} \mid \alpha_i = 0$ if $i \in S_3\}$, there is a vertex-disjoint path set U such that*

$$\frac{1}{2}(1-\beta) \sum_{(i,j)\in U} (\alpha_i - \alpha_j)\mu_j > \sum_{i \in S_1} \alpha_i \lambda_i$$

Theorem 5. *If a Queue-G model queueing system satisfies Assumption 2 holds, and nodes use no-regret learning strategies with $\delta = \frac{\beta \mu_{(m)}}{96(n+m)^2}$, then the queueing system is strongly stable.*

A quick comparison between Assumption 2 and the dual form of condition for centralized stability in Lemma 8 shows that, a queueing system is guaranteed

[3] This is different from the setup in Sect. 2. We now no longer have the independence between the time interval between packet arrivals and histories prior to the their clearing. As will be clear in the proof, this independence is no longer needed in the proof. The introduction of queue lengths makes the change in the potential more directly connected with the queues' utilities.

to be stable with queues using no-regret learning strategies as long as the system can be stable under a centralized policy with twice as many packet arrivals.

As in the proof for Theorem 3, we introduce a similar potential function

$$\Phi(\mathbf{Q}_t) \coloneqq \frac{1}{2}\sum_{i\in S_1\cup S_2} Q_t^i(Q_t^i-1), \tag{19}$$

and define Z_ℓ as its square root at the beginning of the ℓ-th window of length w. There is change in the proofs for both the negative drift condition and the bounded jump condition. We detail here the main different steps in Lemma 9 for the negative drift condition, and relegate the rest to the full version.

Lemma 9. *[Negative drift condition.] Let* $b = \frac{8\sqrt{2(n+m)}}{\beta\mu_{(m)}}(\sum_{i\in S_1}\lambda_i^2 w^2 + \sum_{i\in S_2\cup S_3}\mu_j w^2 + w)$, $c = -\frac{\sqrt{2}\mu_{(m)}\beta w}{128(n+m)}$. *Then* $Z_0 = 0 \le b$ *and, for all* ℓ,

$$Z_\ell > b \quad \Rightarrow \mathbf{E}\left[Z_{\ell+1} - Z_\ell \mid Z_0, \cdots, Z_\ell\right] \le -c.$$

Lemma 10. *[Bounded jump condition.] For each even integer* $p \ge 2$, *there is a constant* d_p, *such that for all* ℓ,

$$\mathbf{E}\left[|Z_{\ell+1} - Z_\ell|^p \mid Z_0, \cdots, Z_\ell\right] \le d_p.$$

At time step $(\ell+1)\cdot w$, let τ_i be the number of unprocessed packets at node i which arrived before time step $\ell \cdot w$. Note the difference from the definition of τ_i in the proof of Theorem 3—there, τ_i is the *age* of the oldest packet in queue i at time ℓw. For any node i and $t \in [\ell\cdot w, (\ell+1)\cdot w]$, $\tau_i \le Q_t^i \le \tau_i + w$. Recall that C_t^j is the indicator variable for server j succeeding in processing a packet if it picks one up, and u_t^i is the utility of queue i at time step t. For a server j, define its *contribution* v_t^j to be u_t^i if j successfully clears a packet from queue i at time step t, and is 0 if it fails to do so. Then, at any time step t, $\sum_{i\in S_1\cup S_2} u_t^i = \sum_{j\in S_2\cup S_3} v_t^j$. A key observation is that, with the new utility functions, when a packet is cleared, the decrease in potential is exactly equal to the increase in the corresponding queue's utility. We can therefore calculate the decrease in potential function by tracking the sum of utilities of all nodes. The following key lemma lower bounds the total utility over a time window as a function of the number of old packets (τ_i's), assuming the realized processing capacities of all servers are around their expectations. Its use of vertex-disjoint paths paves the way for applying Assumption 2. The following lemma gives the lower bound of the utility of queues since queues use no regret learning strategies: for a pair of a long queue and a server, either the server clears many packets from long queues, generating a certain amount of utility or queues will generate a certain amount of utility since its utility will no less than the utility if it always sends packets to this server and its packets have priority since its queue length is large.

Lemma 11. *For any* $\epsilon > 0$, *if* $\sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} C_t^j \ge (1-\epsilon)\mu_j w$ *for each* $j \in S_2 \cup S_3$, *then for any set* U *of vertex-disjoint paths,* $\sum_{i\in S_1\cup S_2}\sum_{t=\ell\cdot w}^{(\ell+1)\cdot w-1} u_t^i \ge \frac{1}{2}\sum_{(i,j)\in U}(\tau_i - \tau_j - w)(1-\epsilon)\mu_j w - \sum_{i\in S_1\cup S_2}\mathrm{Reg}_i(w, (\ell+1)\cdot w)$.

We sketch the rest of the proof, and relegate details to the full version.

When the servers' realized processing capacities are close to their expectations (as in the condition of Lemma 11) and when the queues' regrets are small (which should happen with high probability by assumption), the lower bound given by Lemma 11 on the utility of queues implies a lower bound on the decrease in the potential function due to packet processing (Lemma 12). When the packet arrivals in the sources are close to their expectations, we can further bound the increase in the potential function due to packet arrival (Lemma 13). We define an event B, specified in the full version of this paper, which happens with high probability, and under which all of these events (of concentration and no regret) happen. Roughly speaking, when B happens, the increase in the potential due to packet arrivals is at least offset by the decrease due to packet clearing when we use Assumption 2 to generate an appropriate U in Lemma 11 and relate the term $\frac{1}{2}\sum_{(i,j)\in U}(\tau_i - \tau_j)\mu_j w$ to $\sum_{i\in S_1}\tau_i\lambda_i$. With a small probability, event B does not happen, and it is relatively straightforward to upper bound the increase in the potential in this case.

To put things formally, recall that τ_i is the length of packets in queue i at time step $(\ell{+}1)\cdot w$, which arrived in queue i before time step $\ell\cdot w$; let $\boldsymbol{\tau} = \{\tau_1, \cdots, \tau_n\}$. Given τ_i for each node i, by Assumption 2, let U^* be the vertex-disjoint path such that

$$\frac{1}{2}(1-\beta)\sum_{(i,j)\in U^*}(\tau_i - \tau_j)\mu_j > \sum_{i\in S_1}\tau_i\lambda_i.$$

Lemma 12. *Under* event B, $\Phi(\mathbf{Q}_{\ell\cdot\mathbf{w}}) - \Phi(\boldsymbol{\tau}) \geq \frac{\beta}{4}\sum_{(i,j)\in U^*}(\tau_i - \tau_j)\mu_j w + (1 + \frac{\beta}{8})\sum_{i\in S_1}\lambda_i\tau_i w - \sum_{i\in S_2\cup S_3}\mu_i w^2 - w$.

Lemma 13. *Under* event B, $\Phi(\mathbf{Q}_{(\ell+\mathbf{1})\cdot\mathbf{w}}) - \Phi(\boldsymbol{\tau}) \leq (1 + \frac{\beta}{8})\sum_{i\in S_1}\lambda_i\tau_i w + \sum_{i\in S_1}\lambda_i^2 w^2$.

With a small probability, event B does not happen, and it is relatively straightforward to upper bound the increase in the potential in this case. (Most pessimistically, no packets is cleared during the time window and for each time step, there is a packet arriving at each source node, then the queue length of each source node grows by w.)

Lemma 14. *If event B does not happen, then* $\Phi(\mathbf{Q}_{(\ell+\mathbf{1})\cdot\mathbf{w}}) - \Phi(\mathbf{Q}_{\ell\cdot\mathbf{w}}) \leq \sum_{i\in S_1} Q^i_{\ell\cdot w} w + w^2$.

Lemma 9 follows from combining Lemma 12, 13 and 14.

5 Patient Queueing Model

In this section we extend a model introduced in [4], where queues do not vary their routing policies from step to step, but evaluate the utility of a fixed routing policy over a long period of time. On complete bipartite graphs, Gaitonde and

Tardos [4] showed that Nash equilibria always exist in the resulting game, and that a system is stable under *any* Nash as long as it can be stable under a central policy with $\frac{e}{e-1}$ times as much arrival rates. We obtain a similar result for incomplete bipartite graphs, but the factor in our bicriteria result worsens to 2. We leave for future work to decide whether this factor can be improved.

Below we first describe the model in more detail, before presenting our result.

5.1 Model Description

A bipartite *Patient Queueing Model* has the same packet arrival, routing and processing procedures as in a Queue-B model described in Sect. 2; the servers' priority rule is to pick the oldest packet. The main difference is that the queues are "patient": each queue fixes a routing strategy in the form of a distribution over the servers it can reach, and evaluates its utility/cost over a long time period. Formally, the strategy space of queue i is the simplex over the servers i can send requests to: $\Delta_i \coloneqq \Delta(N^{\text{out}}(i))$. By adopting a strategy $p_i \in \Delta_i$, queue i in each round samples a server according to the distribution given by p_i, independently of all history and other happenings in the system, and sends a request to the sampled server if its queue is non-empty. Let $\boldsymbol{p}_{\text{-}i}$ be the strategies of the queues other than i, then the *cost* of queue i for using strategy p_i is $c_i(p_i, \boldsymbol{p}_{\text{-}i}) \coloneqq \lim_{t\to\infty} \frac{T_t^i}{t}$, where T_t^i is the age of the oldest packet in queue i at time step t. Each queue aims to minimize its cost. A strategy profile $\mathbf{p}$ is a *Nash equilibrium* if for each queue i, $p_i \in \operatorname{argmin}_{p'\in\Delta_i} c_i(p', \boldsymbol{p}_{\text{-}i})$.

Gaitonde and Tardos [4] considered a special case of the Patient queueing Model, where the underlying bipartite graph is complete, i.e., $E = S_1 \times S_3$. They gave an algorithm that, given a strategy profile $\mathbf{p}$, computes an $r_i(\mathbf{p})$ for each queue i, with $r_i(\mathbf{p})$ equal to $c_i(\mathbf{p})$ almost surely. This algorithm played a crucial role in the derivation of their main result. The algorithm extends directly to our more general setting, and is also a key step in our result in this section. We present this algorithm next.

5.2 Gaitonde and Tardos's Algorithm for Computing Costs

Algorithm 1 is a straightforward generalization (to general bipartite graphs) of Gaitonde and Tardo's [4] algorithm for computing the queues' costs. We give some rough intuition here. Thanks to the service priority rule, in the long run, the queues are tiered according to the rates of growth of their lengths: the faster growing ones have higher priority over the slower growing ones. Determining which queues grow the fastest is like a self-fulfilling prophecy: a group of queues grow the fastest even when they have the highest priority. The algorithm enumerates all possible "first tier" queues, and finds the one that fulfills the "prophecy." It then continues to find lower tiers, assuming that all higher tiers have priority. Nailing down this intuition involves intricate probabilistic arguments, and is a major technical accomplishment of [4].

In the step that picks Q_k, there is no ambiguity because the union of minimizers of $f(Q)$ can be shown to be another minimizer. The following results are

Algorithm 1: Computing the queues' costs given their strategies

Input: Queueing system $(S_1 \cup S_3, E, \boldsymbol{\lambda}, \boldsymbol{\mu})$, strategy profile $\mathbf{p}$
$I \leftarrow S_1, k \leftarrow 1$
while $I \neq \emptyset$ **do**
  Compute for each $Q \subseteq I$, $f(Q) = \frac{\sum_{j=1}^{m} \mu_j (1 - \prod_{i \in Q} (1 - p_{ij}))}{\sum_{i \in Q} \lambda_i}$.
  Let Q_k be the minimizer of $f(Q)$, breaking ties in favor of larger cardinality.
  if $f(Q_k) \geq 1$ **then**
    For any $i \in I$, output $r_i(\mathbf{p}) = 0$, terminate.
  else
    For each $i \in Q_k$, $r_i(\mathbf{p}) = 1 - f(Q_k)$;
    Update $\mu_j \leftarrow \mu_j \prod_{i \in Q_k} (1 - p_{ij}), I \leftarrow I \setminus Q_k, k \leftarrow k + 1$.
  end
end
Output: a sequence of queueing groups $Q_1, \cdots, Q_K$ and $r_i(\mathbf{p})$ for each queue i.

straightforward generalizations of corresponding results in [4]. We state them without proofs. Theorem 6 and 7 correspond to Theorem 4.1 and 3.3 in [4], respectively. Theorem 8 is a generalization of Lemma 3.3 and Theorem 4.4 in [4].

Theorem 6. *For any strategy profile* $\mathbf{p}$, $c_i(\mathbf{p}) = \lim_{t \to \infty} \frac{T_t^i}{t} = r_i(\mathbf{p})$ *almost surely.*

Theorem 7. *If the cost function of each queue* i *is defined to be* $r_i(\boldsymbol{p})$*, then every system of the Patient Queueing Model admits a Nash Equilibrium.*

Theorem 8. *Given a queueing system and a strategy profile* $\mathbf{p}$*, let* Q_1 *be the first group of queues output by Algorithm 1. If* $f(Q_1) > 1$*, then the queueing system is stable under* $\boldsymbol{p}$*.*

5.3 Price of Anarchy in Patient Queueing Model

Our bicriteria result for general bipartite graphs is presented again in a form more comparable to the dual form of conditions for centralized stability (see Lemma 1).

Theorem 9. *Given a bipartite queueing system* $(V, E, \boldsymbol{\lambda}, \boldsymbol{\mu})$*, if for any* $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_n) \in \{0, 1\}^n$*, there is a matching matrix* M*, such that* $\frac{1}{2}\boldsymbol{\alpha}^\top M \boldsymbol{\mu} > \boldsymbol{\alpha}^\top \boldsymbol{\lambda}$*, then the system is stable in any Nash Equilibrium in the patient queue model.*

The proofs missing due to lack of space are deferred to the full version of this paper.

We remark that repeatedly playing a Nash Equilibrium strategy profile $\mathbf{p}$ may not be no-regret strategies (see an example in the full version), therefore Theorem 9 is not implied by our results in Sect. 3.

6 Conclusion

In this work generalize the decentralized queueing systems proposed by Gaitonde and Tardos [3,4]. In the full version of this paper, we also consider two more variants of the model with multiple layers, when packet arrivals are adversarial instead of probabilistic, and when packets themselves (rather than the queues they are from) determine which servers can process them. We show that the bicriteria results under no regret strategies are robust against these model modifications. Lastly, we provide a slightly tighter analysis for the Queue-CB model of [3] in the full version of the paper.

Since the bicriteria result in [3] for queues using no regret strategies is tight even for complete bipartite graphs, our results for the more general cases are also tight. On the other hand, we do not know if the factor 2 is tight in our result for patient queues in general bipartite graphs. It seems challenging to directly apply the deformation technique developed in [4] in this more general setting; we leave for future work to investigate the tight bound of this problem.

References

1. Borodin, A., Kleinberg, J.M., Raghavan, P., Sudan, M., Williamson, D.P.: Adversarial queuing theory. J. ACM **48**(1), 13–38 (2001)
2. Christodoulou, G., Kovács, A., Schapira, M.: Bayesian combinatorial auctions. J. ACM **63**(2), 11:1–11:19 (2016)
3. Gaitonde, J., Tardos, É.: Stability and learning in strategic queuing systems. In: Proceedings of the 21st ACM Conference on Economics and Computation, pp. 319–347 (2020)
4. Gaitonde, J., Tardos, E.: Virtues of patience in strategic queuing systems. In: Proceedings of the 22nd ACM Conference on Economics and Computation, pp. 520–540 (2021)
5. Johari, R., Tsitsiklis, J.N.: Efficiency loss in a network resource allocation game. Math. Oper. Res. **29**(3), 407–435 (2004)
6. Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: Meinel, C., Tison, S. (eds.) STACS 1999. LNCS, vol. 1563, pp. 404–413. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-49116-3_38
7. Pemantle, R., Rosenthal, J.S.: Moment conditions for a sequence with negative drift to be uniformly bounded in LR. Stoch. Process. Appl. **82**(1), 143–155 (1999)
8. Roughgarden, T., Tardos, É.: How bad is selfish routing? J. ACM **49**(2), 236–259 (2002)
9. Sentenac, F., Boursier, E., Perchet, V.: Decentralized learning in online queuing systems. Adv. Neural. Inf. Process. Syst. **34**, 18501–18512 (2021)