Routing with Blinkers: Online Throughput Maximization without Queue Length Information

Georgios S. Paschos, Mathieu Leconte, and Apostolos Destounis
Mathematical and Algorithmic Sciences Lab, Huawei Technologies Ltd, Paris, France

***Abstract*—We study a service provisioning system where arriving jobs are routed in an online fashion to any of the available servers; typical applications include datacenters, Internet switches, and cloud computing infrastructures. A common goal in these scenarios is to balance the load across the servers and achieve maximum throughput. For example, the classical online policy Join-the-Shortest-Queue (JSQ) routes an arriving job to the server with the shortest instantaneous queue length. Although JSQ has desirable properties, it requires coordination between the routers and the servers in the form of queue length reports, which prohibits its practical usability in many scenarios.**

In this paper we study the practical case of "routing with blinkers", where no coordination is allowed between the routers and the service provisioning system, and the routers act in an individual manner with limited view of the system state. Every router keeps a log of delays of all jobs it has routed in the past; these are delayed estimates of the actual server queue length. Although easy to acquire, such information is a highly inaccurate depiction of the system state and hence it is unclear whether it is enough to achieve maximum performance. Motivated by the fact that a reasonable policy such as Join-the-Shortest-Delay fails to achieve maximum throughput, we propose a novel routing policy that "samples" the servers periodically and achieves maximum throughput, subject to a condition for the service discipline of the server.

I. Introduction

We focus on a service provisioning system that can be abstracted as a bipartite network graph, see figure 1. Each router is connected to a subset of servers and should choose among them where to route the arriving jobs. Such a model covers a variety of interesting applications like data centers [1], input-queued switches [2], and cloud computing installations [3], where the router may play the role of the load balancer.

In this setup, we assume that the future job arrivals are unknown and our goal is *to achieve maximum throughput*. Clearly, static allocation policies like Round-Robin or Randomized Routing with fixed probabilities fail to achieve this goal [4] since by design they only work for specific arrival rate vectors. A well known policy that solves this problem is the Join-the-Shortest-Queue policy (JSQ). Upon arrival of a job request, JSQ examines the backlogs of the reachable servers and routes the job to the one with the shortest queue. This policy is shown to achieve maximum throughput in the underload [5], and the best load balancing in overload [6], hence JSQ has become the reference policy in this setup. When the set of servers is very large, prior work shows that consulting only two [7], or even less [8], provides most of the system performance. Moreover, queue length reports need not be instantaneous, since delayed queue length estimates are also enough to guarantee maximum throughput [9].

However, JSQ approaches are rarely adopted in the practical systems, mainly because they require the servers to frequently report their backlogs to the routers. This might be difficult for many reasons, (i) the routers may be geographically located in remote areas, (ii) the devices involved may not cooperate, (iii) frequent messaging is not desirable in high-speed systems, (iv) security reasons may prohibit this reporting, etc. We are motivated to study the throughput performance of an online scheduler with no queue length reports.

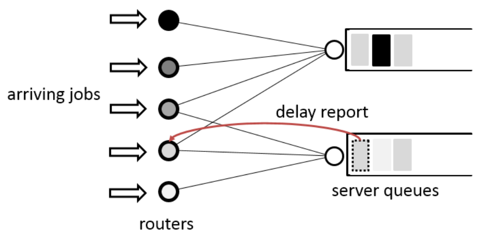

Fig. 1. When a job completes service, it reports to its corresponding router the delay. Arriving jobs are then routed to reachable servers using a routing policy that bases the decisions on reported delays.

In this paper, we assume that the service provisioning system is not willing to provide any information to the routers, except for *acknowledging the completion of service for the routed jobs*. Hence, the server queue lengths are not available to the routers. Instead, the routers receive acknowledgments for past routed jobs with *job delay* information, i.e., the time that the past routed job spent in the server queue, see Figure 1. Since the router action is decided based on partial information about the server congestion, with only implied knowledge of what other routers are deciding, we parallel the way this routing works to the use of blinkers.

The setup of routing with blinkers is very realistic; for example assuming the servers respond to TCP, it is possible to combine TCP acknowledgments with a time stamping mechanism and obtain the required information without any modifications of the service provisioning system. Hence, as opposed to the server queue length, the job delay information can be assumed readily available at the sender of the job. Note that this is different from knowing the queue lengths, since the queue length depends on jobs that have arrived from all routers. *In this paper we study the question whether this job delay information is also enough to achieve maximum throughput,* and we find that the answer is positive.

A. Join the Shortest Delay Policy

Consider the example of two servers with unit capacity in Figure 2-left. Jobs arrive at routers 1 and 3 with rate 0.8 and can only be routed to a single server as indicated by the links. Hence only router 2 with rate 0.3 needs to decide where to route the arriving jobs. If JSQ is used, the arrivals of router 2 will be equally split to the two servers, and each server

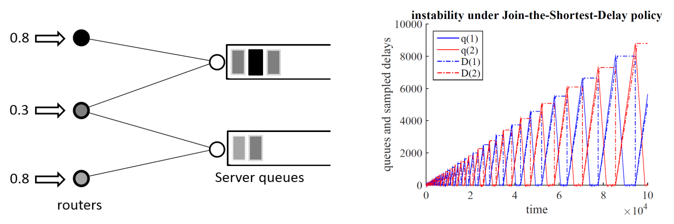

Fig. 2. (left) An example with three routers and two servers. (right) Oscillations of queue lengths under the Join-the-Shortest-Delay policy for the example on the left.

will end up having total job load 0.95, which is less than the server capacity and hence we say that the throughput vector $(0.8, 0.3, 0.8)$ is achievable.

We now define the *Join-the-Shortest-Delay* (JSD) policy which falls into the framework of routing with blinkers. Whenever a new job arrives at stream 2, the most recent delay reports from server 1 and 2 are compared, and the new job is routed to the server with the smallest delay. In other words, router 2 remembers the quality of service received from the last completed job at each server and chooses the server with the best quality (smallest delay). Interestingly, such a simple policy fails to balance the server load properly. In Figure 2-right we observe that the server backlogs perpetually increase in an oscillating fashion. This implies that the throughput vector $(0.8, 0.3, 0.8)$ is not achievable by the JSD policy. Since the point is achievable by JSQ, we conclude that JSD does not achieve maximum throughput.

The unwanted oscillations can be justified by the fact that the delay measurements can be an arbitrarily bad estimate of current server queue lengths. In fact, when a server is measured to have large delay, it is never again used unless another server reaches a point of having even larger delay; by that time the former server may be empty. This observation motivates frequent "sampling" (by routing jobs) of the servers that appear to be congested in order to better track the server congestion status.

B. Strict Priority

The discipline used by server determines the order in which the jobs are served (examples include First-Come First-Serve (FCFS), or priority disciplines). In past JSQ-related work, the order in which the jobs are served does not affect the stability of the server. This is different in our case. Since the service discipline relates the available information (e.g. the job delay measurements) to the actual queue length, it becomes crucial to the system performance. In this example we stress the importance of the service discipline.

In Figure 2-left, assume that stream 2 jobs are prioritized at both servers. From classical queueing theory, we know that the delays of stream 2 jobs do not "see" the low priority traffic [10]. Hence the information available at stream 2 router is not related to the congestion of each server. Thus, the routing decisions become entirely blind to the server queue lengths.

To verify the problem with a numerical example, consider two different arrival vectors $(1, 0.3, 0.5)$ and $(0.5, 0.3, 1)$. They are both stabilizable under JSQ. However, any routing policy that bases the decisions only on atomic past delay measurements, due to job prioritization cannot distinguish the two cases, and will not be able to avoid overloading one of the two servers. Hence under strict priority, the throughput region of routing with blinkers is reduced.

In conclusion, for optimal routing with blinkers, it is required for the servers to satisfy certain conditions regarding the service discipline used. In this paper we analyze such conditions.

II. System Model

A set of routers $\mathcal{U}$ is connected to a set of servers $\mathcal{S}$ using a bipartite graph. A link (u, s), $u \in \mathcal{U}, s \in \mathcal{S}$ in this graph indicates that router u is connected to server s. Let $\mathcal{E}$ be the set of such available connections.

The time is slotted, and $A_u(t)$ jobs arrive to router $u \in \mathcal{U}$ to be routed to one of the servers. The mean arrivals are denoted with $a_u = \mathbb{E}[A_u(t)]$. We also assume that the arrivals in every slot are bounded above by some constant, $A_u(t) \leq A_{\max} < \infty$. The routing decision must be taken instantaneously. We denote with $A^{\pi}_{us}(t)$ the routing variables decided by the router under policy π. Evidently we must have at each slot

$$\sum_{s \in \mathcal{S}(u)} A^{\pi}_{us}(t) = A_u(t), \quad \forall u \in \mathcal{U}, \forall t.$$

where $\mathcal{S}(u) \doteq \{s : (u, s) \in \mathcal{E}\} \subseteq \mathcal{S}$ is the set of servers reachable by router u.

Server s maintains a queue with backlog denoted by $Q_s(t)$, which evolves over time as

$$Q_s(t+1) = (Q_s(t) - \mu_s)^+ + \sum_u A^{\pi}_{us}(t), \tag{1}$$

where μ_s is the service rate of the server s,[1] and $\sum_u A^{\pi}_{us}(t)$ is the total number of routed jobs to s.

We define the process $\mathcal{Q}(t) = [\mathcal{Q}_1(t), ..., \mathcal{Q}_{|\mathcal{S}|}(t)]$, where $\mathcal{Q}_s(t)$ is a list of the packets at server s, tagged by the ID of the router that sent them. The order of the list represents the order by which these packets would be served.[2] In addition, denote by $W_{ms}(t)$ the time that the $m-th$ packet on the list has already spent at the queue by time t. The process $\mathcal{X}(t) = (\mathcal{Q}(t), \mathbf{W}(t))$ is a Markov chain on a countable state space and describes the state of the system. Notice that the queue lengths $\mathbf{Q}(t)$ can be retrieved from $\mathcal{X}(t)$.

In this paper we assume that the processes $\mathcal{X}(t)$ and $\mathbf{Q}(t)$ are not observable by the routers. What can be observed by the routers are the delay measurements of their finished jobs. Consider the matrix $\mathbf{D}(t) = (D_{us}(t))_{u,s}$ where the element $D_{us}(t)$ is the delay experienced by the last job of class u served by server s. By convention, $D_{us}(t) = \infty$ if u is not connected to s, and $D_{us}(t) = 0$ if s has not yet served any job of class u. In addition, consider the matrix $\mathbf{F}(t)$ containing how many jobs of router u are in the queue of each server s.

[1]For simplicity we assume here that the service rate is fixed and integral, but the model can be easily extended to include random service with any positive mean.

[2]For example, under FCFS discipline the packets that arrived earlier are first, under Last-Come First-Serve the packets that arrived later go first, under strict priority the packets of the router with highest priority go first, under round robin the order is randomly changed each time slot, etc.

Note that each router can readily have this information about its own jobs.

Definition 1 (Routing with Blinkers (RwB) Class)**.** *A policy is admissible in the class RwB if the observations available to router u at time t is restricted to $((D_{us}(t))_s, (F_{us}(t))_s)$.*

We define stability of the system as rate stability for the queue length process $\mathbf{Q}(t)$ [11].

Definition 2 (Stability)**.** *The system is (rate) stable if*

$$\lim_{t\to\infty} \frac{Q_s(t)}{t} = 0, \forall s \in \mathcal{S}.$$

We are interested in determining the stability region of the RwB class, and finding a single RwB policy that achieves the stability region.

III. Outer Bound of Unrestricted Routing

Since the RwB class is a subset of routing policies, it follows that their performance is dominated by unrestricted routing, hence the unrestricted routing feasibility region is an outer bound for RwB. In this section we present this bound, and we also give a useful offline randomized policy.

Consider the following *offline bipartite routing problem.* Let a_u denote the amount flow arriving at u, and $\mathbf{a} = (a_u)$ the arrival vector. We decompose the arrivals to a flow matrix $\mathbf{f} = (f_{us})$, where the element f_{us} corresponds to the portion of a_u that is routed to server s. Hence, it must be

$$a_u = \sum_{s\in\mathcal{S}(u)} f_{us}, \quad \forall u \in \mathcal{U}. \quad \text{(routing conservation)} \qquad (2)$$

Additionally, we would like the total flow reaching a server to be smaller than the capacity of the server μ_s, i.e.,

$$\sum_{u:s\in\mathcal{S}(u)} f_{us} \le \mu_s, \quad \forall s \in \mathcal{S}. \quad \text{(server capacity)} \qquad (3)$$

Then we consider the following set (also termed region) of arrivals:

$$\Lambda = \{\mathbf{a} \ge \mathbf{0} : \exists \mathbf{f} \ge \mathbf{0}, (2) - (3) \text{ are satisfied}\}. \qquad (4)$$

The region Λ is also known as the *feasibility region* of the offline bipartite routing. The next lemma makes a (standard) connection between the offline flow problem and the online problem with random job arrivals.

Lemma 1 (Λ is a superset of the RwB region)**.** *Let $a_u = \mathbb{E}[A_u(t)]$ be the mean number of jobs arriving at router u, and $\mathbf{a} = (a_u)$. Suppose that $\mathbf{a} \notin \Lambda$, then any RwB policy is unstable.*

By a trivial adoption of [11] we may prove the lemma for the class of unrestricted routing, hence this completes the proof also for the RwB subclass. Furthermore, consider the following offline randomized policy.

Stationary Randomized Routing (STAT(a)).

- Fix $\mathbf{a}$ in the interior of Λ, and find a flow matrix $\mathbf{f}^*$ that satisfies (2), and (3) less an extra ϵ positive term:

$$\sum_{\mathcal{S}(u)\ni s} f^*_{us} \le \mu_s - \epsilon, \quad \forall s \in \mathcal{S}. \qquad (5)$$

  Since $\mathbf{a}$ is in the interior of Λ, this is possible for some small $\epsilon > 0$.
- At each slot, and each arrived job $j \in \{1, \dots, A_u(t)\}$, select a server with probability $\frac{f^*_{us}}{\sum_{k\in\mathcal{S}(u)} f^*_{uk}}$ and route j to the selected server, so that we finally have

$$\mathbb{E}\big[A^{\text{STAT}}_{us}(t)|\mathcal{X}(t)\big] = \mathbb{E}\big[A^{\text{STAT}}_{us}(t)\big] = f^*_{us}. \qquad (6)$$

Corollary 2 (STAT($\mathbf{a}$) optimality)**.** *Suppose $\mathbf{a}$ is in the interior of Λ, then STAT($\mathbf{a}$) stabilizes the system.*

Thus, given $\mathbf{a}$, STAT($\mathbf{a}$) suffices for optimal routing. Next, we consider the case where $\mathbf{a}$ is unknown.

IV. A Maximum Throughput RwB Policy

In this section we propose an online RwB policy (deprived of server queue length information) that achieves maximum throughput agnostically to arrivals. The result is obtained under a condition for the service discipline.

A. Periodic Probing

Let us assume that each router u periodically sends a *probing* job to each of its reachable servers. If no real job is available to be used as probe, then the probing job is a fake job; it only serves as a way of collecting delay information from the servers.[3] Note, however, that the probing job has to wait in the server queue as the normal jobs, and hence the information obtained in this way is not immediate.

The fake job adds extra load on the servers. In particular the total arrivals to a server at time t become

$$A_{us}(t) = A^{\pi}_{us}(t) + \mathbb{1}[t|T], \qquad (7)$$

where $A^{\pi}_{us}(t)$ are the routed real jobs under policy π, and $\mathbb{1}[t|T]$ is a periodic indicator function equal to 1 once every T slots, and zero otherwise. Note, that the number of probing jobs over a probing period T is less than the maximum in-degree of the server. Hence, (i) $A_{us}(t)$ is also upper bounded by a constant, and (ii) we can keep the incurred overhead small by choosing T large enough.

Let $F_{us}(t)$ be the number of probe packets of router u that have not yet been served by server s. Inspired by prior work on dynamic overlay routing [12], we call these the *probes in flight.* Keeping the number of probes in flight stable would also keep the queue sizes stable, as the total number of packets including probes that may arrive during a period T is bounded. In what follows we will develop a routing scheme based on probes in flight.

B. Server Discipline

Motivated by the example of strict priority in the introduction, we proceed to define a condition for the employed discipline by the server, which guarantees optimality of the RwB class. The development below will prove that this condition is a *sufficient* condition for RwB optimality. A necessary condition is left as future work.

[3]In a real implementation of the probing system, it is possible to adapt the probing scheme to remove entirely the fake jobs. However, we omit this consideration in this paper to simplify exposition.

Condition 1 (Service discipline)**.** *Suppose that the service discipline is such, that we have*

$$Q_s(t) \leq c_1 \min_{(u,s)} F_{us}(t) + c_2, \tag{8}$$

where c_1, c_2 are positive constants.

In words, we require that the probes in flight at server s sent by any router are not arbitrarily far from the server backlog. An example of service discipline that satisfies the above condition is the FCFS where for any two users $u \neq v$ we have

$$|F_{us}(t) - F_{vs}(t)| \leq 1, \quad \forall u, v \in \mathcal{U}, (u,s), (v,s) \in E,$$

from which it follows

$$Q_s(t) \leq \left(1 + \min_{(u,s)} F_{us}(t)\right) T A_{\max} |\mathcal{U}(s)|,$$

where $\mathcal{U}(s)$ is the set of routers that can reach s. Hence (8) is satisfied. On the contrary, strict priority does not satisfy (8).

C. Proposed Policy & Analysis

We propose a dynamic routing policy in the class RwB.

T-probing RwB Policy (T-RwB) of router u.
T-probing. Every T slots, send a fake job at each reachable server in the set $\mathcal{S}(u)$.
Server selection. At each slot, observe $F_{us}(t)$ for all reachable servers and *Route the jobs to the server with the minimum* $F_{us}(t)$, i.e., choose $s^\pi(t) \in \arg\min_{s \in \mathcal{S}(u)} F_{us}(t)$, where the ties are broken arbitrarily, and then select

$$A^\pi_{us}(t) = \begin{cases} A_u(t) & \text{if } s = s^\pi(t) \\ 0 & \text{otherwise} \end{cases}$$

In order to reflect the added workload due to periodic probing, we need the following notion:
Definition 3 ($1/T$–interior of the feasibility region)**.**

$$\Lambda_T = \left\{ \mathbf{a} \geq \mathbf{0} : \exists \mathbf{f} \textit{ s.t. for some } \epsilon' > 0 \right. \\ \left. \frac{|\mathcal{U}(s)|}{T} + \sum_u f_{us} \leq \mu_s - \epsilon', \forall s \in \mathcal{S} \textit{ and } (3) \right\}$$

Terms $\frac{|\mathcal{U}(s)|}{T}$ reflect that the probing packets are injected at a rate of $1/T$ per connected user at every queue. Note that Λ_T is achievable by a random stationary policy. We have the following main result of the paper:
Theorem 3 (Optimality of T-RwB)**.** *Suppose* $\mathbf{a} \in \Lambda_T$. *Further, suppose that all routers operate individually with T-RwB, and the servers use a discipline that satisfies C.1. Then the system is rate stable.*

Note that for large T, Λ_T approaches the feasibility region, therefore policy $T-$RwB is close to optimal.

Proof of Theorem 3:
We use the candidate Lyapunov function $L(t) = \sum_s Q_s(t) \log(1 + Q_s(t))$. The rational for using this non-standard Lyapunov function instead of the classical quadratic one is that we cannot have an accurate enough estimation of the queue lengths at each time instant. In fact, our estimation may be off by a multiplicative factor. This implies that we need to use a Lyapunov function which does not grow too fast to infinity. First, we prove a technical lemma that relates to this function.

To simplify notations, write $\Delta_t X := X(t+1) - X((t)$, for any quantity X. Let $B \geq \max_s \max\{|\mathcal{U}(s)| A_{\max}, \mu_s\}$ be a constant that bounds the queue arrivals and departures in a queue in a time slot.
Lemma 4. $\Delta_t L \leq \sum_s \log\left(1 + Q_s(t)\right)\left(\sum_u A^\pi_{us}(t) - \mu_s\right) +$ *C, where* $C = |\mathcal{S}| B^2 + B \sum_s \log(1 + \mu_s)$.

Proof. For each server s, we have

$$\begin{aligned} &\Delta_t \left(Q_s \log(1 + Q_s)\right) \\ &\quad = Q_s(t+1) \Delta_t \left(\log(1 + Q_s)\right) + \left(\Delta_t Q_s\right) \log(1 + Q_s(t)) \\ &\quad \leq \frac{Q_s(t+1)}{1 + Q_s(t)} \Delta_t Q_s + \left(\Delta_t Q_s\right) \log(1 + Q_s(t)), \end{aligned}$$

where we used $\log x \leq x - 1$ for $x = \frac{1 + Q_s(t+1)}{1 + Q_s(t)}$. As arrivals and departures are bounded by B, the term $\frac{Q_s(t+1)}{1 + Q_s(t)}$ is also bounded by B. Also, adding a bounded constant $B \log(1 + \mu_s)$ to the overall value, we can replace $\Delta_t Q_s$ by $\sum_u A^\pi_{us}(t) - \mu_s$, as the two terms are equal for $Q_s(t) > B$. Then,

$$\Delta_t \left(Q_s \log(1 + Q_s)\right) \leq B^2 + B \log(1 + \mu_s) \\ + \left(\sum_u A^\pi_{us}(t) - \mu_s\right) \log(1 + Q_s(t)).$$

The lemma follows by summing over all servers. □

Next, for policy π=T-RwB we consider the one-slot Lyapunov drift $\mathbb{E}[\Delta_t L | \mathcal{Q}(t)] = \mathbb{E}[L(t+1) - L(t) | \mathcal{Q}(t)]$, where the expectation is taken over random realizations of the arrivals, services, and possibly routing decisions. Applying lemma 4 and using (7)

$$\mathbb{E}[\Delta_t L | \mathcal{Q}(t)] \leq C + \sum_s \log\left(1 + Q_s(t)\right) \\ \times \mathbb{E}\left[\sum_u A^{T\text{-RwB}}_{us}(t) + \mathbb{1}[t|T] - \mu_s \middle| \mathcal{Q}(t)\right].$$

Applying (8) to the pair (u, s), and using convexity of the logarithm we get

$$\begin{aligned} \log(1 + Q_s(t)) &\leq \log(1 + c_1 F_{us}(t) + c_2) \\ &\leq \log(1 + F_{us}(t)) + \log c_1 + (1 + c_2 - c_1)^+ \end{aligned}$$

Plugging this into the drift, we get

$$\mathbb{E}[\Delta_t L | \mathcal{Q}(t)] \leq C' + \sum_s \log\left(1 + Q_s(t)\right)\left(\mathbb{1}[t|T] - \mu_s\right) \\ + \sum_s \sum_u \mathbb{E}\left[A^{T\text{-RwB}}_{us}(t) | \mathcal{Q}(t)\right] \log\left(1 + F_{us}(t)\right)$$

and C' is an appropriate constant. Note that our policy T-RwB routes the jobs to minimize at each slot the term

$$\sum_s \sum_u A^\pi_{us}(t) \log(1 + F_{us}(t)),$$

hence we may replace the decisions $A_{us}^{T\text{-RwB}}(t)$ of our policy with those of STAT($\mathbf{a}$), in which way we get

$$\mathbb{E}[\Delta_t L|\mathcal{Q}(t)] \leq C' + \sum_s \log\left(1+Q_s(t)\right)\left(\mathbb{1}[t|T] - \mu_s\right)$$
$$+ \sum_s \sum_u \mathbb{E}\left[A_{us}^{\text{STAT}}(t)|\mathcal{Q}(t)\right] \log\left(1+F_{us}(t)\right)$$

In addition, note that the fake jobs in flight are also part of the queue length, thus we always have $F_{us}(t) \leq \max_{(u,s)} F_{us}(t) \leq Q_s(t)$. By plugging this into the drift and using (5)-(6) we get

$$\mathbb{E}[\Delta_t L|\mathcal{Q}(t)] \leq C' + \sum_{s\in\mathcal{S}} (|\mathcal{U}(s)|\mathbb{1}[t|T] - \epsilon_s) \log\left(1+Q_s(t)\right), \tag{9}$$

where we have defined $\epsilon_s = \mu_s - \sum_u f_{us}^*$.

Now we are going to examine the process $\mathcal{Q}(kT)$. We take its T-slot drift $\mathbb{E}\left[\Delta_{kT}^T|\mathcal{Q}(kT)\right] = \mathbb{E}[L((k+1)T) - L(kT)|\mathcal{Q}(kT)]$ by using telescoping sums for (9). The idea is to split the queues into two sets: set $\mathcal{S}_+$ containing those that cannot get empty in slots $\{kT+1,...,kT+T\}$ and another set containing the rest. For all queues in the latter it holds $Q_s(kT) \leq \mu_s T$, therefore since T and the maximum change to a queue in a slot are finite, the contribution of all queues in $\mathcal{S}\setminus\mathcal{S}_+$ in the $T-$slot drift can be bounded above by a constant $T\hat{B}$. Assuming that $\max_s Q_s(kT) > T\max_s \mu_s$, i.e. that set $\mathcal{S}_+$ is nonempty (note that the opposite case corresponds to a finite set), then

$$\mathbb{E}\left[\Delta_{kT}^T|\mathcal{Q}(kT)\right] \leq C'' + \sum_{s\in\mathcal{S}_+} \sum_{t=kT}^{kT+T-1} \left(|\mathcal{U}|(s)\mathbb{1}[t|T] + \epsilon_s\right)$$
$$\times\, \mathbb{E}[\log\left(1+Q_s(t)\right)|\mathcal{Q}(kT)]$$

However, for every $t \in \{kT,\dots kT+T-1\}$ we have $Q_s(t) \leq Q_s(kT) + TB$, therefore ($\tilde{C}$ is a finite constant)

$$\mathbb{E}\left[\Delta_{kT}^T|\mathcal{Q}(kT)\right] \leq \tilde{C} + \sum_{s\in\mathcal{S}_+} \left(|\mathcal{U}(s)| - T\epsilon_s\right) \log\left(1+Q_s(kT)\right)$$
$$\leq \tilde{C} - T\epsilon' \sum_{s\in\mathcal{S}_+} \log\left(1+Q_s(kT)\right)$$
$$< \tilde{C} - T\epsilon' \log\left(1+\max_s Q_s(kT)\right),$$

for some $\epsilon' > 0$ since $\mathbf{a} \in \Lambda_T$. The above implies that the process $\mathbf{Q}(kT)$ is positive recurrent (since it is aperiodic, irreducible and the drift of a Lyapunov function is negative outside a finite set), therefore $\lim_{k\to\infty} \frac{Q_s(kT)}{k} = 0, \forall s \in \mathcal{S}$ and we have

$$\lim_{t\to\infty} \frac{Q_s(t)}{t} \leq \lim_{k\to\infty} \frac{Q_s(kT)+BT}{kT} = 0, \forall s \in \mathcal{S},$$

finishing the proof. ■

D. Numerical Examples

We study the example of Figure 1-left, when $a_1 = a_3 = 0.8$, and we vary a_2. From Figure 3 we conclude that T can be appropriately chosen to yield high throughput (the maximum in this case is 0.4) and low delay (small backlogs).

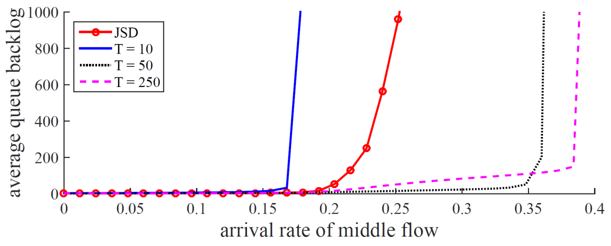

Fig. 3. Average server backlog in simulation runs of one million slots, under JSD, and T-RwB for $T = 10, 50, 250$.

V. Conclusions

We propose *Routing with Blinkers*, a new dynamic routing paradigm, where the routing decisions are based on individual job delay information. Although these reports are easy to obtain with no cooperation effort from the service provisioning system, they represent an inaccurate depiction of the system state. Reasonable policies like *join the server with the shortest delay* are shown to be suboptimal. Also, strict priority rules may hinder the dynamic routing operation. Thus, we propose a probing framework and a condition for the service discipline which guarantees that the congestion of competing users can be detected. In this context, we show that a simple dynamic policy that balances the probes in-flight is throughput optimal. As future work, it is interesting to establish the necessary condition that a service discipline needs to satisfy in order to allow the RwB class to be optimal. The proposed framework can find numerous practical applications in systems where cooperation between router (or load balancer) and server is not feasible.

References

[1] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *SIGCOMM Comput. Commun. Rev.*, 2008.

[2] N. McKeown, V. Anantharam, and J. Walrand, "Achieving 100input-queued switch," in *INFOCOM*, 1996.

[3] S. T. Maguluri, R. Srikant, and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing," in *IEEE INFOCOM*, 2012.

[4] K. Katsalis, G. Paschos, Y. Viniotis, and L. Tassiulas, "CPU provisioning algorithms for service differentiation in cloud-based environments," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 1, pp. 61–74, March 2015.

[5] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[6] C.-P. Li, G. S. Paschos, E. Modiano, and L. Tassiulas, "Dynamic overload balancing in multi-server systems," *IFIP Networking*, 2014.

[7] M. Mitzenmacher, "The power of two choices in randomized load balancing," Ph.D. dissertation, University of California, Berkeley, 1995.

[8] L. Ying, R. Srikant, and X. Kang, "The power of slightly more than one sample in randomized load balancing," in *INFOCOM*, 2015.

[9] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *INFOCOM*, 1998.

[10] D. Bertsekas and R. Gallagher, *Data Networks*. Prentice-Hall, Inc., 1987.

[11] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–147, 2006.

[12] N. M. Jones, G. S. Paschos, B. Shrader, and E. Modiano, "An overlay architecture for throughput optimal multipath routing," in *Proc. of ACM Mobihoc*, 2014.