MapTask Scheduling in MapReduce with Data Locality: Throughput and Heavy-Traffic Optimality

Weina Wang, Kai Zhu, Lei Ying, *Jian Tan, and Li Zhang †

June 2, 2014

Abstract

MapReduce/Hadoop framework has been widely used to process large-scale datasets on computing clusters. Scheduling map tasks with data locality consideration is crucial to the performance of MapReduce. Many works have been devoted to increasing data locality for better efficiency. However, to the best of our knowledge, fundamental limits of MapReduce computing clusters with data locality, including the capacity region and theoretical bounds on the delay performance, have not been well studied. In this paper, we address these problems from a stochastic network perspective. Our focus is to strike the right balance between data-locality and load-balancing to simultaneously maximize throughput and minimize delay. We present a new queueing architecture and propose a map task scheduling algorithm constituted by the Join the Shortest Queue policy together with the MaxWeight policy. We identify an outer bound on the capacity region, and then prove that the proposed algorithm can stabilize any arrival rate vector strictly within this outer bound. It shows that the outer bound coincides with the actual capacity region and the proposed algorithm is throughput optimal. Further, we study the number of backlogged tasks under the proposed algorithm, which is directly related to the delay performance based on Little's law. We prove that the proposed algorithm is heavy-traffic optimal, i.e., it asymptotically minimizes the number of backlogged tasks as the arrival rate vector approaches the boundary of the capacity region. Therefore, the proposed algorithm is also delay optimal in the heavy-traffic regime. The proofs in this paper deal with random processing times with heterogeneous parameters and nonpreemptive task execution, which differentiate our work from many existing works on MaxWeight-type algorithms, so the proof techniques themselves for the stability analysis and the heavy-traffic analysis are also novel contributions.

1 Introduction

Processing large-scale datasets has become an increasingly important and challenging problem as the amount of data created by online social networks, healthcare industry, scientific research, etc., explodes. MapReduce [1] is a simple yet powerful framework for processing large-scale datasets in a distributed and parallel fashion. Originally developed by Google, and later on popularized by its open source implementation Hadoop [2],

*W. Wang, K. Zhu and L. Ying are with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281, USA (e-mail: {weina.wang, kzhu17, lei.ying.2}@asu.edu).

†J. Tan and L. Zhang are with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA (e-mail: {tanji, zhangli}@us.ibm.com).

MapReduce has been widely used in practice by companies including Google, Yahoo!, Facebook, Amazon and IBM.

A production MapReduce cluster may consist of tens of thousands of machines [3]. The stored data are typically organized on distributed file systems (e.g., Google File System (GFS) [4], Hadoop Distributed File System (HDFS) [5]), which divide a large dataset into data chunks (e.g., 128 MB) and store multiple replicas (by default 3 in GFS and HDFS) of each chunk on different machines. A data processing request under the MapReduce framework, called a *job*, consists of two types of tasks: *map* and *reduce*. A map task reads one data chunk and processes it to produce intermediate results (key-value pairs). Then reduce tasks fetch the intermediate results and carry out further computations to produce the final result. Map and reduce tasks are assigned to the machines in the computing cluster by a master node which keeps track of the status of these tasks to manage the computation process. In assigning map tasks, a critical consideration is to place map tasks on or close to machines that store the input data chunks, a problem called *data locality*.

For each task, we call a machine a *local machine* for the task if the data chunk associated with the task is stored locally, and we call this task a *local task* on the machine; otherwise, the machine is called a *remote machine* for the task and correspondingly this task is called a *remote task* on the machine. *Locality* also refers to the fraction of tasks that run on local machines. Improving locality can reduce both the processing time of map tasks and the network traffic load since fewer map tasks need to fetch data remotely. However, assigning all tasks to local machines may lead to an uneven distribution of tasks among machines, i.e., some machines may be heavily congested while others may be idle. Therefore, we need to strike the right balance between data-locality and load-balancing in MapReduce.

We call the algorithm that assigns map tasks to machines a *map-scheduling algorithm* or simply a *scheduling algorithm*. Most existing scheduling algorithms put great effort on increasing data locality for performance improvement [6–11]. Among these scheduling algorithms, the Hadoop Fair Scheduler [7] is the de facto industry standard. It deploys a technique called delay scheduling, which delays some map tasks for a small amount of time to attain higher locality. More discussions on the related works are provided in Section 2.

While the data locality issue has received a lot of attention and scheduling algorithms that improve data locality have been proposed in the literature and implemented in practice, to the best of our knowledge, none of the existing works have studied the fundamental limits of MapReduce computing clusters with data locality. Basic questions such as *what is the capacity region of a MapReduce computing cluster with data locality*, *which scheduling algorithm can achieve the full capacity region*, and *how to minimize the waiting time and congestion in a MapReduce computing cluster with data locality*, remain open.

In this paper, we will address these basic questions from a stochastic network perspective. Trace analysis on some production MapReduce clusters show that a large portion of jobs are map-intensive, and many of them only require map tasks [12] [13]. Therefore, we focus on map-scheduling algorithms and assume that reduce tasks are either not required or not the bottleneck of the job processing. We simply use *task* to refer to *map task* in the rest of this paper. We assume that the data have been divided into chunks, and each chunk has three replicas stored at three different machines. The computing cluster is modeled as a time-slotted system, in which jobs consisting of a number of map tasks arrive at the beginning of each time slot according to some stochastic process. Each map task processes one data chunk and map tasks are *nonpreemptive*. The service (processing) time of a map task is a random variable. The cumulative distribution function (CDF) of this random variable is F_L with mean $1/\alpha$ if the task is served at a local machine, and is F_R with mean

$1/\gamma$ ($\gamma < \alpha$) if the task is served at a remote machine. Both two CDFs have finite variances. Based on this model, we establish the following fundamental results:

- First, we present an outer bound on the capacity region of a MapReduce computing cluster with data locality, where the capacity region consists of all the arrival rate vectors for which there exists a scheduling algorithm that stabilizes the system.
- We propose a new queueing architecture with one local queue for each machine, storing local tasks associated with the machine, and a common queue for all machines. Based on this new queueing architecture, we propose a two-stage scheduling algorithm under which a newly arrived task is routed to one of the three local queues associated with the three local machines or the common queue using the Join the Shortest Queue (JSQ) policy; and when a machine is available, it selects a task from the associated local queue or the common queue using the MaxWeight policy.
- We prove that the joint JSQ and MaxWeight scheduling algorithm is throughput optimal, i.e., it stabilizes any arrival rate vector strictly within the outer bound of the capacity region, which also shows that the outer bound coincides with the actual capacity region. We remark that most existing results on MaxWeight-type scheduling algorithms assume deterministic service time or geometrically distributed service time with preemptive tasks. To the best of our knowledge, the stability of MaxWeight scheduling with nonpreemptive task execution and random processing time has not been established before. So the proof technique itself is a novel contribution of this paper, and may be extended to prove the stability of MaxWeight-type scheduling for other applications.
- In addition to throughput optimality, we further study the number of backlogged tasks, which is directly related to the delay performance based on Little's law. We consider the case that the service times have geometric distributions and assume a heavy local traffic condition. Then the joint JSQ and MaxWeight scheduling algorithm is proved to be heavy-traffic optimal, i.e., it asymptotically minimizes the number of backlogged tasks as the arrival rate vector approaches the boundary of the capacity region. Therefore, the proposed algorithm strikes the right balance between data-locality and load-balancing and is both throughput and delay optimal in the heavy-traffic regime.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 describes the system model and presents the scheduling algorithm for map tasks. Section 4 characterizes the capacity region and shows the throughput optimality. Section 5 analyzes the delay performance under heavy-traffic regime and establishes the heavy-traffic optimality. Simulation results are given in Section 6 and the paper is concluded in Section 7.

2 Related Work

There is a huge body of work (see, e.g. [6–10,14–17]) devoted to the scheduling problem of MapReduce. The Fair Scheduler in Hadoop is the de facto standard [7], in which the delay scheduling technique is used to improve locality. When a machine requests a new task, if the job that should be scheduled next according to fairness does not have available local tasks for this machine, the job is temporarily skipped and the machine

checks the next job in the list. Since machines free up quickly, more tasks are served locally. However, machine idling could be introduced since an available machine may skip all the jobs when it cannot find a local task, and the trade-off between the idling time and locality is not clear.

Many other scheduling algorithms are also commonly used in practice. The default scheduler of Hadoop is a FIFO scheduler [6], which schedules an available machine to serve the map task from the head-of-line job with data closest to the machine. Although some locality optimization is performed, the head-of-line blocking problem results in limited locality and throughput performance. The scheduling algorithm Quincy [11] is designed for Dryad, which is a distributed computational model that allows more complicated data flow than MapReduce. Quincy uses the amount of data transfer as the measure of locality and encodes it into the price model. Then scheduling decisions are made by solving a min-cost flow problem. Joint scheduling of multiple phases of MapReduce have also been proposed [15–17]. In this paper we focus on map-intensive jobs.

There are numerous works on the scheduling problem in MapReduce with data locality and we cannot list them all. Most of these existing scheduling algorithms put great effort on increasing locality for performance improvement. However, locality is not a direct performance measure of the MapReduce cluster. More concerned performance measures are throughput and delay. To the best of our knowledge, none of the existing scheduling algorithms provide direct theoretical guarantee on the throughput performance or the delay performance of the MapReduce cluster.

Since introduced, the idea of the MaxWeight scheduling algorithm [18] is widely applied in the design of scheduling algorithms. However, most existing results on MaxWeight-type scheduling algorithms assume deterministic service time or geometrically distributed service time with preemptive tasks. In the MapReduce system, the service (processing) time of a map task is not deterministic, and it depends on whether the task is executed on a local machine or not. Besides, map tasks are nonpreemptive. Recently in [19], the authors studied MaxWeight scheduling for resource allocation in clouds and independently established a similar result with general service time distributions.

In terms of heavy-traffic analysis, our proof of heavy-traffic optimality follows the Lyapunov drift based technique recently developed in [20], where heavy-traffic optimality of JSQ only and MaxWeight only have been proved. The result has been extended to a joint JSQ and MaxWeight algorithm in [21] (in a different context), where jobs are preemptive and servers are homogeneous. Again, for the map task scheduling problem in MapReduce, tasks are nonpreemptive and machines are heterogeneous due to data locality, which imposes challenges to the state space collapse analysis and makes the proof of heavy-traffic optimality a non-trivial application of the drift-based analysis.

3 System Model and the Scheduling Algorithm

MapReduce is a framework for processing large-scale datasets in a distributed and parallel fashion [1]. A data processing request under the MapReduce framework, called a *job*, is expressed as two functions: *map* and *reduce*. The input data for a job are split into data chunks and stored distributedly across the computing cluster by the distributed file system. Users submit jobs, and the MapReduce framework breaks a job into multiple map tasks and reduce tasks as follows. Each data chunk of the input data is associated with one map task, which reads the data chunk, applies the map function to the data chunk, and then produces the

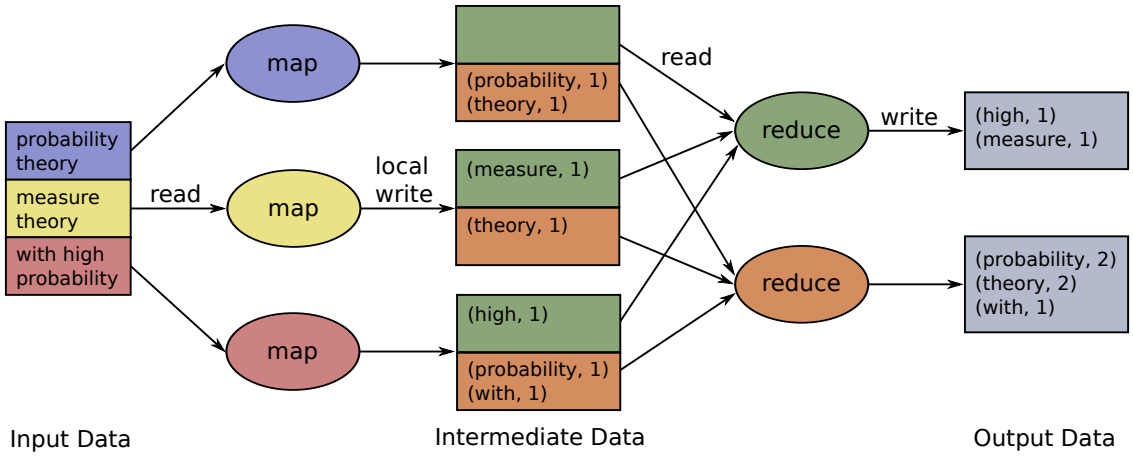


Figure 1: Example of MapReduce: word count.

intermediate data. The intermediate data are in the form of key-value pairs and stored on the local disks of the machines that execute the map tasks. The set of all possible keys of the intermediate data is divided into several subsets, each of which corresponds to a reduce task. The reduce task fetches the intermediate key-value pairs with keys in its assigned subset from all map tasks and applies the reduce function. By doing these, reduce tasks combine the intermediate data and produce the final results.

Now we use a toy example as shown in Figure 1 to demonstrate how MapReduce works. In this example, the objective of the job is to count the number of occurrences of each word. The input data consists of a set of documents and it is split into data chunks with contents shown in Figure 1. Whenever the map function sees a word in the data chunk, it generates a key-value pair using the word as the key and 1 as the value. The set of all the possible keys comprises of all the words. It is divided into two subsets: one containing all the words starting with letters a–m, and another one containing all the words starting with letters n–z. Therefore we have two reduce tasks in the job. For each key, the reduce function sums up all the values belonging to the same key and then gets the number of occurrences of each word.

In this paper, we assume reduce tasks are not the bottleneck of the job processing, and consider the map task scheduling. We consider a time-slotted model for a computing cluster with M machines, indexed $1, \cdots, M$. Let $\mathcal{M} = \{1, 2, \cdots, M\}$ be the set of indices of all the machines. Jobs come in stochastically and when a job comes in, it brings a random number of map tasks, which need to be served by the machines. We assume that each data chunk is replicated and placed at three different machines. Therefore, each task is associated with three local machines. On average, it takes longer time for a machine to process a task if the required data chunk is not stored locally since the machine needs to retrieve the data first. According to the associated local machines, tasks can be classified into *types* denoted by

$$\vec{L} \in \big\{(m_1, m_2, m_3) \in \mathcal{M}^3 \colon m_1 < m_2 < m_3\big\}, \tag{1}$$

where m_1, m_2, m_3 are the indices of the three local machines. We use the notation $m \in \vec{L}$ to indicate machine m is a local machine for type $\vec{L}$ tasks.

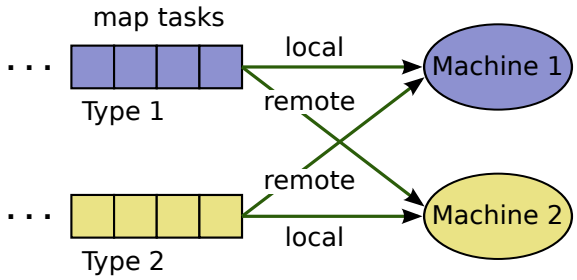


Figure 2: Example showing the impact of task scheduling algorithms with data locality. Machine 1 is local for type 1 tasks and remote for type 2 tasks; machine 2 is local for type 2 tasks and remote for type 1 tasks.

3.1 Arrival and Service

Let $A_{\vec{L}}(t)$ denote the number of type $\vec{L}$ tasks arriving at the beginning of time slot t. We assume that the arrival process is temporally i.i.d. with arrival rate $\lambda_{\vec{L}}$. We further assume that the number of total arrivals in each time slot is bounded by a constant C_A and has a positive probability to be zero. Let $\lambda = \left(\lambda_{\vec{L}} \colon \vec{L} \in \mathcal{L}\right)$ be the arrival rate vector, where $\mathcal{L}$ is the set of task types with arrival rates greater than zero; i.e., $\mathcal{L} = \left\{\vec{L} \colon \lambda_{\vec{L}} > 0\right\}$.

The execution of tasks is *nonpreemptive* and the service time of each task is a random variable with positive integer value. At each machine, we assume that the service times for local tasks have the same CDF F_L with expectation $1/\alpha$ and the service times for remote tasks have the same CDF F_R with expectation $1/\gamma$. Then the service rate is α for local tasks and γ for remote tasks. We assume $\alpha > \gamma$ since processing local tasks is more efficient than processing remote tasks. By making this assumption data locality is integrated into our service model. We also assume that F_L and F_R have finite variances. Note that for now we do not impose any requirements on the shapes of F_L and F_R except the assumptions on the expectations and variances. Later on in the heavy-traffic analysis, we assume F_L and F_R to be geometric distributions with means $1/\alpha$ and $1/\gamma$, respectively. This service model is motivated by [10], where the service times for tasks follow geometric distributions with different means due to data locality.

3.2 Task Scheduling Algorithm

The task scheduling problem is to assign the incoming tasks to the machines. Due to data locality, the task scheduling algorithm can significantly affect the efficiency of the system. As a simple example, consider a system with two machines and two task types as shown in Figure 2. Each data chunk has one copy stored on one of the two machines. Then there are two task types: type 1 with machine 1 as the local machine, and type 2 with machine 2 as the local machine. Suppose both the arrival processes of the two types are Bernoulli processes with parameter λ, and the service processes are Bernoulli processes with parameter α for local tasks and parameter γ for remote tasks. Recall that $\alpha > \gamma$. Algorithm 1 always assigns tasks to remote machines. Then the system can be stable only when λ is less than γ, resulting in a throughput less than 2γ. However, algorithm 2 always assigns tasks to local machines, under which the system can achieve a throughput close to 2α ($> 2\gamma$). Further, to compare the delay performance, consider the scenario when $\lambda < \gamma$. Under both algorithms, two Geo/Geo/1 queues are formed. Under algorithm 1 the average waiting time of a task is $\frac{1}{\gamma-\lambda}$, while under algorithm 2, the average waiting time of a task becomes $\frac{1}{\alpha-\lambda}$ ($< \frac{1}{\gamma-\lambda}$).

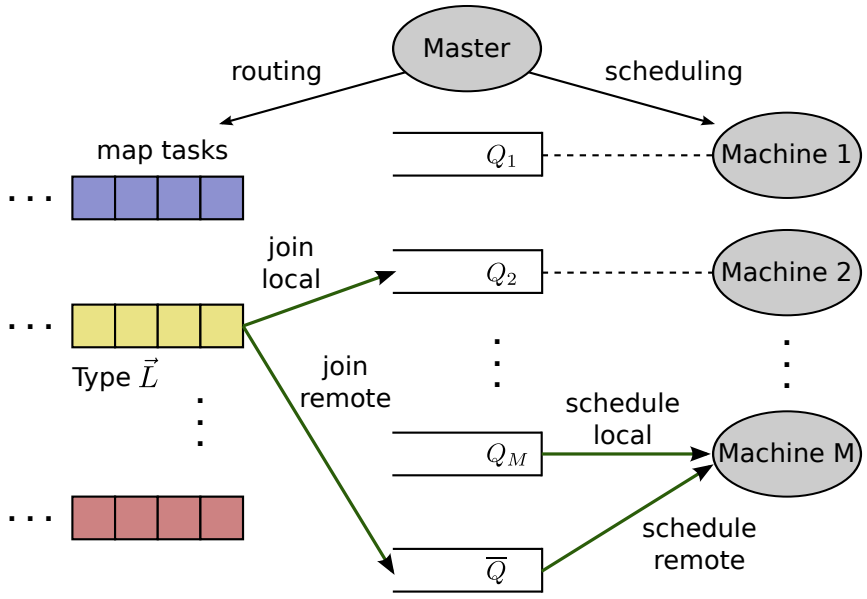


Figure 3: The queue architecture: $Q_1, Q_2, \cdots, Q_M$ are local queues associated with machines $1, 2, \cdots, M$, respectively, and $\overline{Q}$ is the common remote queue. The scheduling algorithm has two steps: routing and scheduling.

In this paper, we consider a task scheduling algorithm consisting of two parts: routing and scheduling. The task scheduling algorithm is performed by the master, where the data locality information is also available. We present a new queue architecture as illustrated in Figure 3. For each machine m, the master maintains a *local queue* to buffer local tasks, denoted by Q_m; for all the machines, the master maintains a *common remote queue*, denoted by $\overline{Q}$ (or sometimes Q_{M+1}). The queue length vector

$$Q(t) = \big(Q_1(t), Q_2(t), \cdots, Q_M(t), \overline{Q}(t)\big) \tag{2}$$

denotes the queue lengths at the beginning of time slot t. When a task comes in, the master routes the task to some queue in the queueing system; when a machine is available, the master schedules the machine to serve a task from the corresponding local queue or the common remote queue. These two steps are illustrated in Figure 3. We call the first step *routing*, and with a slight abuse of terminology we call the second step *scheduling*. It should be clear from the context that whether we are referring to the whole task scheduling problem or to this service scheduling step. Based on our queue architecture, we propose the following joint routing and scheduling algorithm.

- **Join the Shortest Queue (JSQ) Routing.** When a task comes in, the master compares the queue lengths of the three local queues of the local machines and the common remote queue, and then routes the task to the shortest one. Thus for an incoming task with type $\vec{L}$, the routing destination is the shortest queue in $\{Q_m : m \in \vec{L}\} \cup \{\overline{Q}\}$. Ties are broken randomly.

- **MaxWeight Scheduling.** If a machine just finished a task at time slot $t-1$, we say the machine is *available* at time slot t, and otherwise the machine is *busy* since it must be serving some task. Available machines are scheduled according to the MaxWeight algorithm. Let machine m be an available machine

at time slot t. Then it is scheduled to serve a task from the local queue if $\alpha Q_m(t) \geq \gamma \overline{Q}(t)$ and a task from the common remote queue otherwise. Note that there can be multiple available machines that are scheduled to serve the tasks from the common remote queue at the same time slot. We let these machines fetch the tasks in the common remote queue in a random order. Busy machines are scheduled to continue to serve their unfinished tasks, i.e., the execution of tasks is *nonpreemptive*.

The proposed algorithm has a very low computational complexity. In the routing step, the master compares four queue lengths for each arriving task. In the scheduling step, the master compares $\alpha Q_m(t)$ and $\gamma \overline{Q}(t)$ for each available machine. Therefore for each arriving task and each available machine, the computational complexity is a constant.

Remark. The proposed scheduling algorithm, with a slight modification, works in heterogeneous environments, where machines may process tasks at different rates [14]. For each machine m, let α_m and γ_m denote the service rates for local and remote tasks, respectively. Then we modify the MaxWeight scheduling step as follows: an available machine is scheduled to serve the local queue if $\alpha_m Q_m(t) \geq \gamma_m \overline{Q}(t)$ and the common remote queue otherwise. After this modification, the throughput optimality result also holds under heterogeneity. The proof is a direct extension of the current proof, so we focus on the current model for simplicity.

Remark. Here we note that the proposed map task scheduling algorithm can cooperate with job-level strategies. Under the proposed queue architecture, each queue can be further divided into multiple subqueues according to the job that the task comes from, i.e., into per job subqueues. Then in the scheduling step, after being scheduled to serve some queue, an available machine can further choose from the subqueues according to the job-level strategy. However, this change will not affect our analysis throughout this paper, so we only consider this structure in the simulation part.

3.3 Queue Dynamics

In this section, we derive the dynamics of the queue length vector $Q(t)$. We use a working status vector

$$\Psi(t) = \big(\Psi_1(t), \Psi_2(t), \ldots, \Psi_M(t)\big) \tag{3}$$

to denote the working status of the machines at the beginning of time slot t, where $\Psi_m(t)$ is the amount of time that machine m has spent on the currently in-service task, and it is set back to 0 if machine m just finished a task at the previous time slot. By this definition, machine m is available if and only if $\Psi_m(t) = 0$.

At the beginning of time slot t, the master checks the working status of machines and the queue length vector. Then tasks arrive and the master routes the incoming tasks according to the queue length vector. Let $A_{\vec{L},m}(t)$ and $\overline{A}_{\vec{L}}(t)$ denote the number of type $\vec{L}$ task arrivals allocated to Q_m and $\overline{Q}$, respectively. Then $A_{\vec{L}}(t) = \sum_{m \in \vec{L}} A_{\vec{L},m}(t) + \overline{A}_{\vec{L}}(t)$. The number of arrivals allocated to each queue is expressed by the following arrival vector

$$A(t) = \big(A_1(t), \ldots, A_M(t), \overline{A}(t)\big), \tag{4}$$

where

$$A_m(t) = \sum_{\vec{L}\colon\, m \in \vec{L}} A_{\vec{L},m}(t), m \in \mathcal{M}, \;\; \overline{A}(t) = \sum_{\vec{L} \in \mathcal{L}} \overline{A}_{\vec{L}}(t). \tag{5}$$

After the routing step, the master schedules the machines according to their working status and the queue length vector. If a machine is available and is scheduled to serve the local queue, or if the machine is busy serving the local queue, we say the machine is *scheduled to* the local queue. Otherwise we say the machine is *scheduled to* the common remote queue. For both cases, an available machine may remain available if the assigned queue is empty when the machine tries to fetch a task. If this happens we say this machine is *idling*. The scheduling decisions are recorded by the following scheduling decision vector

$$\eta(t) = \big(\eta_1(t), \eta_2(t), \ldots, \eta_M(t)\big), \tag{6}$$

where

$$\eta_m(t) = \begin{cases} 0 & \text{machine } m \text{ is scheduled to } Q_m \text{ and nonidling,} \\ 1 & \text{machine } m \text{ is scheduled to } Q_m \text{ and idling,} \\ 2 & \text{machine } m \text{ is scheduled to } \overline{Q} \text{ and nonidling,} \\ 3 & \text{machine } m \text{ is scheduled to } \overline{Q} \text{ and idling.} \end{cases} \tag{7}$$

At the end of time slot t, finished tasks depart the queueing system. We use random variables $S_m^l(t)$ and $S_m^r(t)$ defined as follows to describe the service performed by machine m. If machine m is scheduled to the local queue and finishes a task at the end of time slot t, let $S_m^l(t) = 1$; if machine m is scheduled to the local queue and is idling at time slot t, let $S_m^l(t) = \alpha$; otherwise let $S_m^l(t) = 0$. Similarly, if machine m is scheduled to the common remote queue and finishes a task at the end of time slot t, let $S_m^r(t) = 1$; if machine m is scheduled to the common remote queue and is idling at time slot t, let $S_m^r(t) = \gamma$; otherwise let $S_m^r(t) = 0$. Let

$$S_m(t) = S_m^l(t), \;\; \overline{S}(t) = \sum_{m=1}^{M} S_m^r(t), \tag{8}$$

which we call the *service* performed to the queues. Note that the service is not the departure of the queue. To obtain the departures, we also consider the following quantities, which we call the *unused service*:

$$U_m(t) = \begin{cases} 0 & \text{if } Q_m(t) + A_m(t) \geq 1, \\ S_m^l(t) & \text{if } Q_m(t) + A_m(t) = 0, \end{cases} \tag{9}$$

$$\overline{U}(t) = \sum_{m=1}^{M} S_m^r(t) - \sum_{m \in \mathcal{R}(t)} S_m^r(t), \tag{10}$$

where $\mathcal{R}(t)$ is the set of machines that are scheduled to the common remote queue and nonidling at time slot t. Then the departures of the queues are

$$S_m(t) - U_m(t), m \in \mathcal{M}, \;\; \overline{S}(t) - \overline{U}(t). \tag{11}$$

Let the service vector be denoted by

$$S(t) = \big(S_1(t), \ldots, S_M(t), \overline{S}(t)\big), \tag{12}$$

and the unused service vector be denoted by

$$U(t) = \big(U_1(t), \ldots, U_M(t), \overline{U}(t)\big). \tag{13}$$

Then by the above notation, the queue dynamics can be finally expressed as

$$Q(t+1) = Q(t) + A(t) - S(t) + U(t). \tag{14}$$

The following lemma presents a property of the unused service vector $U(t)$, which will be used in the analysis of the throughput performance. The proof of this lemma is provided in our technical report [22].

Lemma 1. (Approximate orthogonality) *Let $Q(t)$ be the queue length vector and $U(t)$ be the unused service vector at time slot t. Then for any time slot t,*

$$\langle Q(t), U(t) \rangle \leq M^2, \tag{15}$$

where M is the number of machines in the system.

In this queueing system, since the execution of tasks is nonpreemptive and the distributions of service times are not assumed to follow some particular distribution, the queueing process $\big\{Q(t), t \geq 0\big\}$ itself is not a Markov chain. We consider the working status vector $\Psi(t)$, the scheduling decision vector $\eta(t)$ and the queue length vector $Q(t)$, and they together form a Markov chain $\big\{Z(t) = \big(\Psi(t), \eta(t), Q(t)\big), t \geq 0\big\}$. We assume that the initial state of this Markov chain is $Z(0) = \big(0_{M\times 1}, 0_{M\times 1}, 0_{(M+1)\times 1}\big)$ and the state space $\mathcal{S} \subseteq \mathbb{N}^M \times \{0,1,2,3\}^M \times \mathbb{N}^{M+1}$ consists of all the states that can be reached from the initial state, where $\mathbb{N}$ is the set of nonnegative integers.

Remark. The Markov chain $\big\{Z(t), t \geq 0\big\}$ is irreducible and aperiodic under our assumptions. For any state $Z = \big(\Psi, \eta, Q\big)$ in the state space, the queue lengths in the system are finite. Let τ be an integer such that $F_L(\tau) > 0$ and $F_R(\tau) > 0$. Such τ exists since F_L and F_R are CDFs. Consider the event that each task in the system is finished within τ time slots and there is no arrivals for the next $\tau \sum_{m=1}^{M+1} Q_m$ time slots. If this event happens, the Markov chain reaches the initial state after $n = \tau \sum_{m=1}^{M+1} Q_m$ time slots. Since this event happens with a positive probability, the n-step transition probability from Z to the initial state is positive. Therefore any state in the state space can reach the initial state, and hence the Markov chain is irreducible. The Markov chain is also aperiodic since the transition probability from the initial state to itself is positive.

4 Throughput Optimality

In this section, the throughput optimality of the map-scheduling algorithm proposed in Section 3.2 is established. Let λ be the arrival rate vector. Consider a scheduling algorithm and let $\Phi(t)$ be the number of unfinished tasks in the system at time slot t under this scheduling algorithm. Then we say that this scheduling algorithm *stabilizes* the system for λ if

$$\lim_{C\to\infty} \lim_{t\to\infty} \Pr(\Phi(t) > C) = 0, \tag{16}$$

i.e., if the number of unfinished tasks does not explode. For the proposed scheduling algorithm, the number of unfinished tasks is $\Phi(t) = \sum_{m=1}^{M+1} Q_m(t)$. Therefore the existence of a stationary distribution of the Markov chain $\{Z(t), t \geq 0\}$ is a sufficient condition for the stability of the system.

The *capacity region* is defined to be the set of arrival rate vectors for which there exists a scheduling algorithm that stabilizes the system. We will first identify an outer bound of the capacity region, and then prove that the proposed map-scheduling algorithm stabilizes any arrival rate vector strictly within this outer bound, which means that the capacity region coincides with the outer bound, and the proposed algorithm is *throughput optimal*.

4.1 Outer Bound of the Capacity Region

Recall that $\lambda = (\lambda_{\vec{L}} \colon \vec{L} \in \mathcal{L})$ is the arrival rate vector. For any task type $\vec{L} \in \mathcal{L}$, we assume that the number of type $\vec{L}$ tasks that are allocated to machine m has a rate $\lambda_{\vec{L},m} \geq 0$; then $\lambda_{\vec{L}} = \sum_{m=1}^{M} \lambda_{\vec{L},m}$. Such a rate vector $(\lambda_{\vec{L},m} \colon \vec{L} \in \mathcal{L}, m \in \mathcal{M})$ is called a *decomposition* of the arrival rate vector $\lambda = (\lambda_{\vec{L}} \colon \vec{L} \in \mathcal{L})$, and the index range may be omitted for conciseness in the rest of this paper. A necessary condition for the arrival rate vector λ to be supportable is that the average arrivals allocated to each machine in one time slot must be served within one time slot. Therefore, for any machine m, λ needs to satisfy

$$\sum_{\vec{L} \colon m \in \vec{L}} \frac{\lambda_{\vec{L},m}}{\alpha} + \sum_{\vec{L} \colon m \notin \vec{L}} \frac{\lambda_{\vec{L},m}}{\gamma} \leq 1, \tag{17}$$

where the left hand side is the amount of time that machine m needs to serve the arrivals allocated to it in one time slot on average, since the service rate is α for local tasks and γ for remote tasks.

Let Λ be the set of arrival rate vectors such that each element of it has a decomposition satisfying the necessary condition above. Formally,

$$\begin{aligned} \Lambda = \Bigg\{ \lambda = (\lambda_{\vec{L}} \colon \vec{L} \in \mathcal{L}) \colon\ & \text{there exists } (\lambda_{\vec{L},m}) \text{ such that} \\ & \text{for any } \vec{L} \in \mathcal{L} \text{ and } m \in \mathcal{M},\ \lambda_{\vec{L},m} \geq 0, \\ & \text{for any } \vec{L} \in \mathcal{L},\ \lambda_{\vec{L}} = \sum_{m=1}^{M} \lambda_{\vec{L},m}, \text{ and} \\ & \text{for any } m \in \mathcal{M},\ \sum_{\vec{L} \colon m \in \vec{L}} \frac{\lambda_{\vec{L},m}}{\alpha} + \sum_{\vec{L} \colon m \notin \vec{L}} \frac{\lambda_{\vec{L},m}}{\gamma} \leq 1. \Bigg\} \end{aligned} \tag{18}$$

Then Λ gives an outer bound of the capacity region.

4.2 Achievability

Theorem 1. (Throughput Optimality) *The map-scheduling algorithm proposed in Section 3.2 stabilizes any arrival rate vector strictly within Λ. Hence, Λ is the capacity region of the system, and the proposed algorithm is throughput optimal.*

For any arrival rate vector $\lambda \in \Lambda^o$, $\{Z(t) = (\Psi(t), \eta(t), Q(t)), t \geq 0\}$ is an irreducible and aperiodic

Markov chain, and the number of unfinished tasks $\Phi(t) = \sum_{m=1}^{M+1} Q_m(t)$. If this Markov chain is positive recurrent, then the condition (16) is satisfied since the distribution of $Z(t)$ will converge to the stationary distribution as $t \to \infty$. Thus the stability can be obtained by proving the positive recurrence.

However, since the execution of tasks is nonpreemptive and the service times are random, there are two aspects in our model that make the proof difficult and differentiate our proof from the standard Lyapunov analysis. First, in the Markov chain, both $\Psi(t)$ and $Q(t)$ can be unbounded, so they both need to be considered in the Lyapunov function. Second, scheduling decisions are made only when some tasks are just completed. Then the expected departures at time slot t not only depend on the state of the Markov chain at time slot t but also depend on the state at some previous time slot, which makes it difficult to find a Lyapunov function whose one time slot drift is negative outside a finite set. Therefore we consider a period of T time slots and analyze the T time slot drift of a Lyapunov function. By the extension of the Foster-Lyapunov theorem, it suffices to find a Lyapunov function and a positive integer T for each arrival rate vector $\lambda \in \Lambda^o$ such that the T time slot Lyapunov drift is bounded within a finite subset of the state space and negative outside the subset. The intuition for throughput optimality is that for large enough T, the average service time of departed tasks in the same category (local or remote) concentrates on its mean value. Then the system behaves like a system with deterministic service times ($1/\alpha$ for a local task and $1/\gamma$ for a remote task), for which the MaxWeight scheduling is throughput optimal.

Consider any arrival rate vector $\lambda \in \Lambda^o$. Since Λ^o is an open set, there exists $\epsilon > 0$ such that $\lambda' = (1+\epsilon)\lambda \in \Lambda$. Suppose the decomposition of λ' that satisfies (17) is $\left(\lambda'_{\vec{L},m}\right)$. Then

$$\left(\lambda_{\vec{L},m} \colon \vec{L} \in \mathcal{L}, m \in \mathcal{M}\right) := \left(\frac{\lambda'_{\vec{L},m}}{1+\epsilon} \colon \vec{L} \in \mathcal{L}, m \in \mathcal{M}\right) \tag{19}$$

is a decomposition of λ, and for any $m \in \mathcal{M}$,

$$\sum_{\vec{L}\colon m \in \vec{L}} \frac{\lambda_{\vec{L},m}}{\alpha} + \sum_{\vec{L}\colon m \notin \vec{L}} \frac{\lambda_{\vec{L},m}}{\gamma} \le \frac{1}{1+\epsilon}. \tag{20}$$

Let $\tilde{\lambda} = \left(\tilde{\lambda}_1, \tilde{\lambda}_2, \cdots, \tilde{\lambda}_{M+1}\right)$ be defined as

$$\tilde{\lambda}_m = \sum_{\vec{L}\colon m \in \vec{L}} \lambda_{\vec{L},m}, \ \ m \in \mathcal{M}, \quad \tilde{\lambda}_{M+1} = \sum_{m=1}^{M} \sum_{\vec{L}\colon m \notin \vec{L}} \lambda_{\vec{L},m}. \tag{21}$$

Then $\tilde{\lambda}$ can be thought of as one possibility of arrival allocation so that each $\tilde{\lambda}_m$ is the arrival rate allocated to the queue Q_m. The following two lemmas reveal more properties of $\tilde{\lambda}$ and lead to the proof of the throughput optimality theorem. The proofs of these two lemmas are provided in Appendix B.

Lemma 2. *Consider any arrival rate vector $\lambda \in \Lambda^o$ and the corresponding $\tilde{\lambda}$ defined in* (21)*. Let $A(t)$ be the arrival vector to the queues and $Q(t)$ be the queue length vector at time slot t. Then under the JSQ routing algorithm, for any t_0 and any $t \ge t_0$,*

$$\mathbb{E}\left[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\lambda}\rangle \,\middle|\, Z(t_0)\right] \le 0. \tag{22}$$

Lemma 3. *Consider any arrival rate vector $\lambda \in \Lambda^o$ and the corresponding $\tilde{\lambda}$ defined in (21). Let $S(t)$ be the service vector and $Q(t)$ be the queue length vector at time slot t. Then under the MaxWeight scheduling algorithm, there exists $T_0 > 0$ such that for any $T \geq T_0$ and any t_0,*

$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1}\Big(\langle Q(t), \tilde{\lambda}\rangle - \langle Q(t), S(t)\rangle\Big) \;\Big|\; Z(t_0)\right] \leq -\theta T\|Q(t_0)\|_1 + C_1, \tag{23}$$

where θ and C_1 are positive constants independent of $Z(t_0)$ and $\|\cdot\|_1$ is the L^1-norm.

Proof of Theorem 1. Consider the following Lyapunov function

$$W(Z(t)) = \|Q(t)\|^2 + \|\Psi(t)\|_1 = \sum_{m=1}^{M} Q_m^2(t) + \overline{Q}^2(t) + \sum_{m=1}^{M}\Psi_m(t), \tag{24}$$

where $\|\cdot\|$ is the L^2-norm and $\|\cdot\|_1$ is the L^1-norm. In the rest of the proof we use $W(t)$ as an abbreviation of $W(Z(t))$. Then for any arrival rate vector $\lambda \in \Lambda^o$, it suffices to find a finite set $\mathcal{B} \subseteq \mathcal{S}$ and two constants δ and C with $\delta > 0$ such that for some positive integer T and any time slot t_0,

$$\mathbb{E}\big[W(t_0+T) - W(t_0) \,\big|\, Z(t_0) = Z\big] \leq -\delta \;\text{ if } Z \in \mathcal{B}^c, \tag{25}$$

$$\mathbb{E}\big[W(t_0+T) - W(t_0) \,\big|\, Z(t_0) = Z\big] \leq C \quad \text{if } Z \in \mathcal{B}. \tag{26}$$

Let $D_Q(t_0) = \mathbb{E}\big[\|Q(t_0+T)\|^2 - \|Q(t_0)\|^2 \;\big|\; Z(t_0)\big]$ and $D_\Psi(t_0) = \mathbb{E}\big[\|\Psi(t_0+T)\|_1 - \|\Psi(t_0)\|_1 \;\big|\; Z(t_0)\big]$. Then for any time slot t_0, the T time slot Lyapunov drift can be calculated as

$$D(Z(t_0)) := \mathbb{E}\big[W(t_0+T) - W(t_0) \;\big|\; Z(t_0)\big] = D_Q(t_0) + D_\Psi(t_0). \tag{27}$$

First let us consider $D_Q(t_0)$. By the queue dynamics (14),

$$\begin{aligned} D_Q(t_0) &= \mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1}\Big(\|Q(t+1)\|^2 - \|Q(t)\|^2\Big) \;\Big|\; Z(t_0)\right] \\ &= \mathbb{E}\left[\sum_{t}\Big(2\langle Q(t), A(t) - S(t)\rangle + 2\langle Q(t), U(t)\rangle + \|A(t) - S(t) + U(t)\|^2\Big) \;\Big|\; Z(t_0)\right]. \end{aligned}$$

The summation range is usually omitted for conciseness when it is clear from the context. By Lemma 1, the second term $2\langle Q(t), U(t)\rangle$ is bounded by a constant. Meanwhile, by our assumptions, the arrival vector $A(t)$ and the service vector $S(t)$ are bounded, and thus the unused service vector $U(t)$ is also bounded since each entry of $U(t)$ is no greater than the corresponding entry of $S(t)$ by definition. Therefore, the drift $D_Q(t_0)$ can be bounded as

$$D_Q(t_0) \leq 2\mathbb{E}\left[\sum_{t}\langle Q(t), A(t) - S(t)\rangle \;\Big|\; Z(t_0)\right] + C_2,$$

where $C_2 > 0$ is a constant independent of $Z(t_0)$. Utilizing the vector $\tilde{\lambda}$ defined in (21) we have

$$\mathbb{E}\left[\sum_{t}\langle Q(t), A(t) - S(t)\rangle \;\Big|\; Z(t_0)\right]$$

$$= \mathbb{E}\Big[\sum_t \Big(\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\lambda}\rangle\Big) \;\Big|\; Z(t_0)\Big] + \mathbb{E}\Big[\sum_t \Big(\langle Q(t), \tilde{\lambda}\rangle - \langle Q(t), S(t)\rangle\Big) \;\Big|\; Z(t_0)\Big].$$

Since the JSQ routing algorithm and MaxWeight scheduling algorithm are used, by Lemma 2 and Lemma 3, there exists $T_0 > 0$ such that for any $T \geq T_0$,

$$\mathbb{E}\Bigg[\sum_{t=t_0}^{t_0+T-1} \langle Q(t), A(t) - S(t)\rangle \;\Bigg|\; Z(t_0)\Bigg] \leq -\theta T \|Q(t_0)\|_1 + C_1,$$

where $\theta > 0$ and $C_1 > 0$ are two constants independent of $Z(t_0)$. Hence

$$D_Q(t_0) \leq -2\theta T \|Q(t_0)\|_1 + 2C_1 + C_2.$$

Next let us consider $D_\Psi(t_0)$, which can be calculated as

$$D_\Psi(t_0) = \sum_{m=1}^{M} \mathbb{E}\big[\Psi_m(t_0+T) - \Psi_m(t_0) \;\big|\; Z(t_0)\big].$$

For any $m \in \mathcal{M}$, let t_m^* be the first time slot at which machine m is available; i.e.,

$$t_m^* = \min\big\{\tau \colon \tau \geq t_0, \Psi_m(\tau) = 0\big\}. \tag{28}$$

Note that $\Psi_m(t_m^*) = 0$. Since $\Psi_m(t)$ indicates the amount of time that machine m has spent on the currently in-service task, for any t we have $\Psi_m(t+1) \leq \Psi_m(t) + 1$. Thus if $t_m^* \leq t_0 + T$, $\Psi_m(t_0+T) \leq \Psi_m(t_m^*) + (t_0 + T - t_m^*) \leq T$. Then we use t_m^* to decompose the probability space and get the following bound

$$\begin{aligned}
&\mathbb{E}\big[\Psi_m(t_0+T) - \Psi_m(t_0) \;\big|\; Z(t_0)\big] \\
&= \mathbb{E}\big[\Psi_m(t_0+T) - \Psi_m(t_0) \;\big|\; Z(t_0), t_m^* \leq t_0+T\big] \cdot \Pr\big(t_m^* \leq t_0+T \;\big|\; Z(t_0)\big) \\
&\quad + \mathbb{E}\big[\Psi_m(t_0+T) - \Psi_m(t_0) \;\big|\; Z(t_0), t_m^* > t_0+T\big] \cdot \Pr\big(t_m^* > t_0+T \;\big|\; Z(t_0)\big) \\
&\leq \big(T - \Psi_m(t_0)\big) \Pr\big(t_m^* \leq t_0+T \;\big|\; Z(t_0)\big) + T \Pr\big(t_m^* > t_0+T \;\big|\; Z(t_0)\big) \\
&= -\Psi_m(t_0) \Pr\big(t_m^* \leq t_0+T \;\big|\; Z(t_0)\big) + T.
\end{aligned}$$

Since

$$\lim_{T\to\infty} \Pr\big(t_m^* \leq t_0+T \;\big|\; Z(t_0)\big) = 1,$$

for any ξ with $0 < \xi < 1$, there exists T_m such that for any $T \geq T_m$, $\Pr\big(t_m^* \leq t_0+T \;\big|\; Z(t_0)\big) \geq \xi$. Therefore

$$\mathbb{E}\big[\Psi_m(t_0+T) - \Psi_m(t_0) \;\big|\; Z(t_0)\big] \leq -\xi\Psi_m(t_0) + T.$$

To combine $D_Q(t_0)$ and $D_\Psi(t_0)$, pick any $T \geq \max\big\{T_0, T_1, \cdots, T_M, \frac{\xi}{2\theta}\big\}$. Let $C = 2C_1 + C_2 + MT$. Then by (27), the T time slot Lyapunov drift satisfies

$$\mathbb{E}\big[W(Z(t_0+T)) - W(Z(t_0)) \;\big|\; Z(t_0)\big] \leq -2\theta T \|Q(t_0)\|_1 - \xi\|\Psi(t_0)\|_1 + 2C_1 + C_2 + MT$$

$$\leq -\xi\big(\|Q(t_0)\|_1 + \|\Psi(t_0)\|_1\big) + C.$$

Pick any $\delta > 0$ and let $\mathcal{B} = \Big\{Z = (\Psi, \eta, Q) \in \mathcal{S}\colon \|Q\|_1 + \|\Psi\|_1 \leq \frac{C+\delta}{\xi}\Big\}$. Then $\mathcal{B}$ is a finite subset of $\mathcal{S}$. For any $Z \in \mathcal{B}^c$, $D(Z) \leq -\delta$ and for any $Z \in \mathcal{B}$, $D(Z) \leq C$. This finishes the proof for positive recurrence. Hence, Λ is the capacity region of the system, and the proposed algorithm is throughput optimal. □

Remark. The common Lyapunov function for proving the throughput optimality of MaxWeight-type algorithms is $\|Q\|^2$, which does not work in our case because $\big\{Z\colon \|Q\|_1 \leq K\big\}$ is not a finite set for any $K > 0$. Note that for a fixed Q, we have infinitely many (Ψ, η, Q), which is a result of nonpreemptive tasks and general service time distributions. We therefore have to use a modified Lyapunov function defined in (24), which includes the $\|\Psi(t)\|_1$ and guarantees that the drift is negative when either $Q(t)$ or $\Psi(t)$ is large.

5 Heavy-Traffic Optimality

In this section, we analyze the performance of the proposed algorithm beyond throughput. We consider the case that the service times have geometric distributions, i.e., the CDFs F_L and F_R are geometric distributions with parameters $1/\alpha$ and $1/\gamma$, respectively. We will show that in the *heavy-traffic* regime, the proposed algorithm asymptotically minimizes the number of backlogged tasks.

5.1 Simplified Markov Chain

When the service times are geometrically distributed, the Markov chain that represents the queueing system can be simplified due to the memoryless property of geometric distributions. Instead of keeping track of both the amount of time that a machine has spent on the currently in-service task and the scheduling decision, we only need to maintain the following working status vector

$$f(t) = \big(f_1(t), f_2(t), \cdots, f_M(t)\big), \tag{29}$$

where $f_m(t)$ is the working status of machine m at time slot t, and $f_m(t) = 0, 1, 2$ denotes available, working on a local task, and working on a remote task, respectively.

We still use the arrival vector $A(t) = \big(A_1(t), \cdots, A_M(t), \overline{A}(t)\big)$ to denote the arrivals to each queue. In contrast to the general case, we do not need to differentiate between whether a machine is idling or not. The scheduling decisions are recorded by the following scheduling decision vector

$$\sigma(t) = \big(\sigma_1(t), \sigma_2(t), \cdots, \sigma_M(t)\big), \tag{30}$$

where

$$\sigma_m(t) = \begin{cases} 1 & \text{if machine } m \text{ is scheduled to } Q_m, \\ 2 & \text{if machine } m \text{ is scheduled to } \overline{Q}. \end{cases} \tag{31}$$

We still use random variables $S_m^l(t)$ and $S_m^r(t)$ to denote the service performed by machine m. Let $\text{Bern}(p)$ denote the Bernoulli distribution with parameter p. If machine m is scheduled to the local queue and finishes a task at the end of time slot t, let $S_m^l(t) = 1$; if machine m is scheduled to the local queue and is idling

at time slot t, let $S_m^l(t) \sim \text{Bern}(\alpha)$; otherwise let $S_m^l(t) = 0$. Similarly, if machine m is scheduled to the common remote queue and finishes a task at the end of time slot t, let $S_m^r(t) = 1$; if machine m is scheduled to the common remote queue and is idling at time slot t, let $S_m^r(t) \sim \text{Bern}(\gamma)$; otherwise let $S_m^r(t) = 0$. Therefore, since we assume geometrically distributed service times, $S_m^l(t)$ is a Bernoulli random variable with parameter α as long as machine m is scheduled to the local queue, and $S_m^r(t)$ is a Bernoulli random variable with parameter γ as long as machine m is scheduled to the common remote queue. Consider

$$\mu_m^l(t) = \begin{cases} \alpha & \text{if } \sigma_m(t) = 1, \\ 0 & \text{if } \sigma_m(t) = 2, \end{cases} \qquad \mu_m^r(t) = \begin{cases} 0 & \text{if } \sigma_m(t) = 1, \\ \gamma & \text{if } \sigma_m(t) = 2. \end{cases} \tag{32}$$

Then $S_m^l(t) \sim \text{Bern}\big(\mu_m^l(t)\big)$ and $S_m^r(t) \sim \text{Bern}\big(\mu_m^r(t)\big)$. We still consider the service vector

$$S(t) = \big(S_1(t), \cdots, S_M(t), \overline{S}(t)\big), \tag{33}$$

where

$$S_m(t) = S_m^l(t), \ \ m \in \mathcal{M}, \quad \overline{S}(t) = \sum_{m=1}^{M} S_m^r(t). \tag{34}$$

Then the queue dynamics can still be described by (14).

Now the queueing system can be expressed by the simplified Markov chain $\big\{Z(t) = \big(Q(t), f(t)\big), t \geq 0\big\}$. We still assume that the system starts from the state that all the queues are empty and all the machines are available, so the initial state of the Markov chain is $Z(0) = \big(0_{(M+1)\times 1}, 0_{M\times 1}\big)$. The state space $\mathcal{S} \subseteq \mathbb{N}^{M+1} \times \{0,1,2\}^M$ consists of all the states that can be reached from the initial state. Using similar arguments we can see, this Markov chain is irreducible, aperiodic, and positive recurrent.

5.2 Heavy-Traffic Regime

Suppose the set of existing task types $\mathcal{L}$ is such that there are M_l machines, each of which is considered as a local machine by some task type, and the other $M_r = M - M_l$ machines are remote machines for all the task types. Denote the set of the machines which can have local tasks as $\mathcal{M}_l$ and the set of the machines which only have remote tasks as $\mathcal{M}_r$. Then

$$\begin{aligned} \mathcal{M}_l &= \big\{m \in \mathcal{M} \colon \exists\ \vec{L} \in \mathcal{L},\ \text{s.t. } m \in \vec{L}\big\}, \\ \mathcal{M}_r &= \mathcal{M} - \mathcal{M}_l, \end{aligned}$$

and $|\mathcal{M}_l| = M_l$, $|\mathcal{M}_r| = M_r$. Without loss of generality, we assume $\mathcal{M}_l = \big\{1, \cdots, M_l\big\}$ and $\mathcal{M}_r = \big\{M_l + 1, \cdots, M\big\}$.

We consider the heavy-traffic regime such that for any subset $\mathcal{H}$ of $\mathcal{M}_l$, the sum arrival rate of tasks with local machines in $\mathcal{H}$ is greater than the processing capability of the machines in $\mathcal{H}$. Formally, consider any subset of machines $\mathcal{H} \subseteq \mathcal{M}_l$ and let $\mathcal{N}(\mathcal{H}) = \big\{\vec{L} \in \mathcal{L} \colon \exists\ m \in \mathcal{H},\ \text{s.t. } m \in \vec{L}\big\}$ be the set of task types with local machines in $\mathcal{H}$. Let $\lambda = \big(\lambda_{\vec{L}} \colon \vec{L} \in \mathcal{L}\big) \in \Lambda^o$ be the arrival rate vector. Then the considered heavy-traffic

regime assumes that for any $\mathcal{H} \subseteq \mathcal{M}_l$,

$$\sum_{\vec{L} \in \mathcal{N}(\mathcal{H})} \lambda_{\vec{L}} > |\mathcal{H}|\alpha, \tag{35}$$

which is referred to as the *heavy local traffic assumption*. In this regime, the machines in $\mathcal{M}_l$ cannot accommodate all the arrivals, so we assume $\mathcal{M}_r \neq \emptyset$ to stabilize the system. A legitimate application scenario for this heavy-traffic regime is node leasing. It is a common practice, (e.g., for IBM Platform Computing), to support node leasing for MapReduce groups. Consider a MapReduce cluster that is shared by several groups, and each group has a dedicated pool of machines. There are agreements among these groups that they can lease some nodes to other groups when needed. In this case, it is therefore very common for a busy pool to borrow nodes from another group for executing its MapReduce jobs.

When the arrival rate vector λ satisfies the heavy local traffic assumption, there exists a decomposition of λ as described in the following lemma. This lemma is a crucial step to derive the state space collapse result, and the proof is provided in Appendix C.

Lemma 4. *Let the arrival rate vector λ satisfy the heavy local traffic assumption, and consider any ϵ with $0 \leq \epsilon \leq \epsilon_{\max}$, where $\epsilon_{\max}$ is a positive constant. Then there exists a decomposition $\left(\lambda_{\vec{L},m} \colon \vec{L} \in \mathcal{L}, m \in \mathcal{M}\right)$ of λ and a positive constant κ not depending on ϵ such that:*

1. *For any $\vec{L} \in \mathcal{L}$ and any $m \in \mathcal{M}_l$, $\lambda_{\vec{L},m} \geq \kappa$.*
2. *For any $\vec{L} \in \mathcal{L}$, $\overline{\lambda}_{\vec{L}} = \sum_{m \notin \vec{L}} \lambda_{\vec{L},m} \geq \kappa$.*
3. *For any $m \in \mathcal{M}_l$, $\sum_{\vec{L}\colon m \in \vec{L}} \lambda_{\vec{L},m} = \alpha - \frac{\epsilon}{M_l+1}$.*

When λ is in the capacity region, it is easy to see $\sum_{\vec{L} \in \mathcal{L}} \lambda_{\vec{L}} \leq M_l\alpha + M_r\gamma$. We assume that

$$\sum_{\vec{L} \in \mathcal{L}} \lambda_{\vec{L}} = M_l\alpha + M_r\gamma - \epsilon, \tag{36}$$

where $\epsilon > 0$ characterizes the distance between the arrival rate vector and the capacity boundary. The superscript $^{(\epsilon)}$ will be used in this section to indicate the heavy-traffic parameter ϵ. Consider any arrival processes $\left\{\left(A^{(\epsilon)}_{\vec{L}}(t) \colon \vec{L} \in \mathcal{L}\right), t \geq 0\right\}$ with arrival rate vector $\lambda^{(\epsilon)} = \left(\lambda^{(\epsilon)}_{\vec{L}} \colon \vec{L} \in \mathcal{L}\right)$ satisfying (36). Then

$$\mathbb{E}\left[\sum_{\vec{L}} A^{(\epsilon)}_{\vec{L}}(t)\right] = \sum_{\vec{L}} \lambda^{(\epsilon)}_{\vec{L}} = M_l\alpha + M_r\gamma - \epsilon. \tag{37}$$

The variance of the number of overall arrivals is given by

$$\mathrm{Var}\left(\sum_{\vec{L}} A^{(\epsilon)}_{\vec{L}}(t)\right) = (\sigma^{(\epsilon)})^2. \tag{38}$$

Let $\left\{Z^{(\epsilon)}(t) = \left(Q^{(\epsilon)}(t), f^{(\epsilon)}(t)\right), t \geq 0\right\}$ denote the Markov chain consisting of the queue length vector and the working status vector. Later we will let ϵ go to zero to get *heavy-traffic limits*.

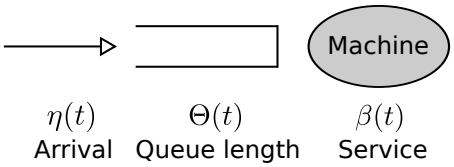


Figure 4: The lower bounding system.

5.3 Lower Bound

In this subsection, we derive a lower bound on the expectation of the number of backlogged tasks in the system under any stabilizing scheduling algorithm.

Recall that the arrival processes in the system are $\big\{\big(A_{\vec{L}}^{(\epsilon)}(t)\colon \vec{L}\in\mathcal{L}\big), t\geq 0\big\}$. Consider any stabilizing scheduling algorithm and let $\Phi^{(\epsilon)}(t)$ be the number of backlogged tasks at time slot t in the system under this scheduling algorithm. Assume that the process $\{\Phi^{(\epsilon)}(t), t\geq 0\}$ starts from $\Phi^{(\epsilon)}(0)=0$ and converges in distribution to a random variable $\widehat{\Phi}^{(\epsilon)}$ as $t\to\infty$.

Consider a hypothetical single server queueing system as depicted in Figure 4. The arrival process $\big\{\eta^{(\epsilon)}(t), t\geq 0\big\}$ and service process $\big\{\beta(t), t\geq 0\big\}$ of this system is defined as follows:

$$\eta^{(\epsilon)}(t)=\sum_{\vec{L}} A_{\vec{L}}^{(\epsilon)}(t), \quad \beta(t)=\sum_{i=1}^{M_l} X_i(t)+\sum_{j=1}^{M_r} Y_j(t), \tag{39}$$

where all the processes $\big\{X_i(t), t\geq 0\big\}, i=1,2\cdots,M_l$ and $\big\{Y_j(t), t\geq 0\big\}, j=1,2,\cdots,M_r$ are independent and each process is temporally i.i.d. For any $i=1,\cdots,M_l$ and any t, let $X_i(t)\sim \text{Bern}(\alpha)$, and for any $j=1,\cdots,M_r$ and any t, let $Y_j(t)\sim\text{Bern}(\gamma)$. The queueing process in this system is denoted by $\big\{\Theta^{(\epsilon)}(t), t\geq 0\big\}$ and we let it start from $\Theta^{(\epsilon)}(0)=0$.

For any time slot t, the overall arrivals to the map task scheduling system in MapReduce and to the hypothetical single server queueing system are the same, and the overall service in the former system is stochastically less than the overall service in the latter system. Thus the queue length $\Phi^{(\epsilon)}(t)$ is stochastically greater than $\Theta^{(\epsilon)}(t)$ and $\mathbb{E}\big[\Phi^{(\epsilon)}(t)\big]\geq\mathbb{E}\big[\Theta^{(\epsilon)}(t)\big]$. Since $\big\{\Theta^{(\epsilon)}(t), t\geq 0\big\}$ is an irreducible, aperiodic, positive recurrent Markov chain, it will converge to a random variable $\widehat{\Theta}^{(\epsilon)}$ in distribution as $t\to\infty$. Then $\mathbb{E}\big[\widehat{\Phi}^{(\epsilon)}\big]\geq\mathbb{E}\big[\widehat{\Theta}^{(\epsilon)}\big]$.

Lemma 4 in [20] provides a lower bound on $\mathbb{E}\big[\widehat{\Theta}^{(\epsilon)}\big]$, which is also a lower bound on $\mathbb{E}\big[\widehat{\Phi}^{(\epsilon)}\big]$. The arrival process and service process in the hypothetical system satisfy that

$$\mathbb{E}\big[\eta^{(\epsilon)}(t)\big]=M_l\alpha+M_r\gamma-\epsilon,\ \ \text{Var}\big(\eta^{(\epsilon)}(t)\big)=(\sigma^{(\epsilon)})^2,$$

$$\mathbb{E}\big[\beta(t)\big]=M_l\alpha+M_r\gamma,\ \ \text{Var}\big(\beta(t)\big)=M_l\alpha(1-\alpha)+M_r\gamma(1-\gamma).$$

Denote $\text{Var}\big(\beta(t)\big)$ by ν^2. Then the lower bound is given by $\mathbb{E}\big[\widehat{\Theta}^{(\epsilon)}\big]\geq\frac{(\sigma^{(\epsilon)})^2+\nu^2+\epsilon^2}{2\epsilon}-\frac{M}{2}$, and the following theorem holds.

Theorem 2. (Lower Bound) *For the map task scheduling system in MapReduce, consider an arrival process such that the number of total arrivals at each time slot has expectation $M_l\alpha+M_r\gamma-\epsilon$ and variance $(\sigma^{(\epsilon)})^2$. Let $\{\Phi^{(\epsilon)}(t), t\geq 0\}$ be the process of the number of backlogged tasks in the system under any stabilizing*

scheduling algorithm. Assume that it starts from $\Phi^{(\epsilon)}(0) = 0$ *and converges in distribution to a random variable* $\widehat{\Phi}^{(\epsilon)}$ *as* $t \to \infty$*. Then for any* ϵ *with* $0 < \epsilon < M_l\alpha + M_r\gamma$*, the expectation of the number of backlogged tasks in steady state can be lower-bounded as*

$$\mathbb{E}\big[\widehat{\Phi}^{(\epsilon)}\big] \geq \frac{(\sigma^{(\epsilon)})^2 + \nu^2 + \epsilon^2}{2\epsilon} - \frac{M}{2}. \tag{40}$$

Therefore, in the heavy-traffic limit as the arrival rate approaches the service rate from below, assuming the variance $(\sigma^{(\epsilon)})^2$ *converges to a constant* σ^2*, the lower bound becomes*

$$\liminf_{\epsilon \to 0^+} \epsilon \mathbb{E}\big[\widehat{\Phi}^{(\epsilon)}\big] \geq \frac{\sigma^2 + \nu^2}{2}. \tag{41}$$

Especially, assuming that the Markov chain $\big\{Z^{(\epsilon)}(t) = \big(Q^{(\epsilon)}(t), f^{(\epsilon)}(t)\big), t \geq 0\big\}$ *is in steady state under the proposed map-scheduling algorithm, the lower bound becomes*

$$\liminf_{\epsilon \to 0^+} \epsilon \mathbb{E}\left[\sum_{m=1}^{M+1} Q_m^{(\epsilon)}(t)\right] \geq \frac{\sigma^2 + \nu^2}{2}. \tag{42}$$

5.4 State Space Collapse

For a single server queueing system like the one in Figure 4, the discrete-time Kingman's bound [23] gives an upper bound on the expectation of the queue length in steady state, which is derived by studying the drift of an appropriate Lyapunov function in steady state. The task scheduling system in MapReduce is a more complicated queueing system, which consists of multiple queues and thus has a multi-dimensional state space. However, in the heavy-traffic scenario, we will show that the multi-dimensional state description of the system reduces to a single dimension in the sense that the deviation from a particular direction has bounded moments, independent of the heavy-traffic parameter. This behavior of the queueing system in heavy-traffic scenario is called *state space collapse* [20]. When the state space collapse happens, the system can be analyzed by the similar techniques as used for the single-dimensional system. In this subsection, we will establish the state space collapse for the task scheduling system in MapReduce.

For the Markov chain $\big\{Z^{(\epsilon)}(t) = \big(Q^{(\epsilon)}(t), f^{(\epsilon)}(t)\big), t \geq 0\big\}$ with state space $\mathcal{S}$, since our aim is to analyze the queue lengths, we focus on the subspace of $\mathcal{S}$ consisting of the queue length vectors. In this paper, the norms are L^2 norms and denoted by $\|\cdot\|$ unless otherwise specified. Let $c \in \mathbb{R}_+^{M+1}$ be a vector with unit norm, where $\mathbb{R}_+$ is the set of nonnegative real numbers. Then the corresponding parallel and perpendicular components of a queue length vector Q are

$$Q_\parallel = \langle Q, c\rangle c, \quad Q_\perp = Q - Q_\parallel. \tag{43}$$

If all the moments of $\|Q_\perp\|$, which represent the deviation of the queue length vector from the direction c, are bounded by constants not depending on the heavy-traffic parameter ϵ, the state space is said to collapse to the direction of c.

Let

$$c = \frac{1}{\sqrt{M_l + 1}}(\underbrace{1, \cdots, 1}_{M_l}, \underbrace{0, \cdots, 0}_{M_r}, 1) \tag{44}$$

be the direction that we will prove the state space collapses to, where the first M_l entries are ones and the following M_r entries are zeros. Consider the Lyapunov function

$$V_\perp(Z) = \|Q_\perp\|.$$

We will prove that the drift of $V_\perp$ satisfies condition (C1) and (C2) in the following lemma and the resulted constants do not depend on the heavy traffic parameter ϵ. Then by this lemma, $V_\perp\big(Z^{(\epsilon)}(t)\big) = \|Q_\perp^{(\epsilon)}(t)\|$ has bounded moments in steady state not depending on ϵ, i.e., the state space collapses. This lemma is an extension of Lemma 1 in [20] and can be derived from [24].

Lemma 5. *For an irreducible, aperiodic and positive recurrent Markov chain $\big\{X(t), t \geq 0\big\}$ over a countable state space $\mathcal{X}$, suppose $V\colon \mathcal{X} \to \mathbb{R}_+$ is a Lyapunov function. For some positive integer T, the T time slot drift of V at X defined as follows*

$$\Delta V(X) := \big[V(X(t_0+T)) - V(X(t_0))\big] I(X(t_0) = X),$$

where $I(X(t_0) = X)$ is the indicator function, satisfies the following conditions

(C1) There exist constants δ and ζ with $\delta > 0$ such that

$$\mathbb{E}\big[\Delta V(X) \;\big|\; X(t_0) = X\big] \leq -\delta, \text{ for all } X \in \mathcal{X} \text{ with } V(X) \geq \zeta.$$

(C2) There exists a constant C such that

$$\Pr\big(|\Delta V(X)| \leq C\big) = 1, \text{ for all } X \in \mathcal{X}.$$

Then $\big\{V(X(t)), t \geq 0\big\}$ converges in distribution to a random variable $\widehat{V}$, and there exist constants θ^* and C^* with $\theta^* > 0$ such that $\mathbb{E}\Big[e^{\theta^* \widehat{V}}\Big] \leq C^*$, which directly implies that all moments of $\widehat{V}$ exist and are finite.

Theorem 3. (State Space Collapse) *For the map-scheduling system in MapReduce, consider an arrival process such that the number of total arrivals at each time slot has expectation $M_l\alpha + M_r\gamma - \epsilon$ and variance $(\sigma^{(\epsilon)})^2$. Let the arrival rate vector be strictly within the capacity region and satisfy the heavy local traffic assumption. Suppose the Markov chain $\big\{Z^{(\epsilon)}(t) = \big(Q^{(\epsilon)}(t), f^{(\epsilon)}(t)\big), t \geq 0\big\}$ is in steady state under the proposed map-scheduling algorithm. Then for any t and any ϵ with*

$$0 < \epsilon < \min\big\{\epsilon_{\max}, M_l\alpha + M_r\gamma, M_r(M_l+1)\gamma\big\}, \tag{45}$$

there exist constants $\big\{C_r\colon r \in \mathbb{N}\big\}$ not depending on ϵ such that for each positive integer r,

$$\mathbb{E}\big[\|Q_\perp^{(\epsilon)}(t)\|^r\big] \leq C_r,$$

where the $\perp$ component is w.r.t. the direction c defined in (44).

Proof. It suffices to verify that the drift of $V_\perp$ satisfies condition (C1) and (C2) in Lemma 5 and the constants do not depend on the heavy traffic parameter ϵ. Fix an ϵ within the range (45) specified in the

theorem. Then the superscript $^{(\epsilon)}$ is temporarily omitted for simplicity and it will be revived at the end of the proof. Following the notation in the lemma, the T time slot drift of $V_\perp$ is denoted by $\Delta V_\perp(Z) = \big[V_\perp(Z(t_0+T)) - V_\perp(Z(t_0))\big] I(Z(t_0) = Z)$. Before analyzing $\Delta V_\perp(Z)$, we list the following lemmas to provide some useful basic facts about the queueing process. The proofs of these lemmas are in Appendix D. Recall that the queues in the system evolve according to the dynamics $Q(t+1) = Q(t) + A(t) - S(t) + U(t)$.

Lemma 6. *Let c be a vector with unit norm in $\mathbb{R}^{M+1}$. Then for any $t \geq 0$,*

$$\|Q_\parallel(t+1)\|^2 - \|Q_\parallel(t)\|^2 \geq 2\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle.$$

Lemma 7. *Consider a time slot t_0 and a positive integer T. Then for any t with $t_0 \leq t < t_0 + T$,*

$$\big|\|Q_\perp(t)\| - \|Q_\perp(t_0)\|\big| \leq T\sqrt{M+1}\max\{M, NA_{\max}\}.$$

Lemma 8. *Consider a time slot t_0 and a positive integer T. For any t with $t_0 \leq t < t_0 + T$, let $G(t) = \langle Q(t), A(t) - S(t)\rangle - \langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle$. Then $G(t) \leq h\|Q_\perp(t_0)\| + F_1$, where $h = \sqrt{M+1}\max\{M, NA_{\max}\}$ and $F_1 = (M+1)T\left(\max\{M, NA_{\max}\}\right)^2$ are constants.*

Now we start analyzing the drift $\Delta V_\perp(Z)$. First notice that the condition (C2) follows directly from Lemma 7 when we choose the constant $C = T\sqrt{M+1}\max\{M, NA_{\max}\}$. Then we are left with the condition (C1). Consider the following two Lyapunov functions

$$W(Z) = \|Q\|^2, \quad W_\parallel(Z) = \|Q_\parallel\|^2. \tag{46}$$

Then $V_\perp(Z) = \sqrt{W(Z) - W_\parallel(Z)}$, and the drift of $V_\perp$ satisfies the following inequality due to the concavity of the square root function (Lemma 7 in [20]),

$$\Delta V_\perp(Z) \leq \frac{1}{2\|Q_\perp\|}\big(\Delta W(Z) - \Delta W_\parallel(Z)\big), \tag{47}$$

where the drifts $\Delta W(Z)$ and $\Delta W_\parallel(Z)$ also follow the notation of the T time slot drift as defined in Lemma 5. For the drift $\Delta W(Z)$, by the boundedness of arrivals and service and the property of the unused service in Lemma 1,

$$\begin{aligned}\mathbb{E}\big[\Delta W(Z(t_0)) \;\big|\; Z(t_0)\big] &= \mathbb{E}\big[\|Q(t_0+T)\|^2 - \|Q(t_0)\|^2 \;\big|\; Z(t_0)\big] \\ &\leq 2\mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1}\langle Q(t), A(t) - S(t)\rangle \;\middle|\; Z(t_0)\right] + B_2,\end{aligned}$$

where $B_2 > 0$ is a constant. For the drift $\Delta W_\parallel(Z)$, by Lemma 6,

$$\begin{aligned}\mathbb{E}\big[\Delta W_\parallel(Z(t_0)) \;\big|\; Z(t_0)\big] &= \mathbb{E}\big[\|Q_\parallel(t_0+T)\|^2 - \|Q_\parallel(t_0)\|^2 \;\big|\; Z(t_0)\big] \\ &\geq 2\mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1}\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle \;\middle|\; Z(t_0)\right].\end{aligned}$$

Combining these two inequalities let us focus on the term $G(t) := \langle Q(t), A(t) - S(t)\rangle - \langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle$.

Consider the following random variables

$$\begin{aligned} t_m^* &= \min\{\tau\colon \tau \geq t_0, f_m(\tau) = 0\}, \ \ m \in \mathcal{M}, \\ t^* &= \max_{1\leq m\leq M} t_m^*. \end{aligned} \tag{48}$$

By the time slot t^* all the machines have been available at least once. Let $T = JK$, where J and K are positive integers. Then

$$\mathbb{E}\big[\textstyle\sum_t G(t) \bigm| Z(t_0)\big] = \mathbb{E}\big[\textstyle\sum_t G(t) \bigm| Z(t_0), t^* \geq t_0 + K\big] \cdot \Pr\big(t^* \geq t_0 + K \bigm| Z(t_0)\big) \tag{49}$$

$$+ \mathbb{E}\big[\textstyle\sum_t G(t) \bigm| Z(t_0), t^* < t_0 + K\big] \cdot \Pr\big(t^* < t_0 + K \bigm| Z(t_0)\big). \tag{50}$$

For the term (49), by Lemma 8 we have $\mathbb{E}\big[\sum_t G(t) \bigm| Z(t_0), t^* \geq t_0 + K\big] \leq hT\|Q_\perp(t_0)\| + F_1 T$. For the term (50), taking conditional expectation yields

$$\begin{aligned} &\mathbb{E}\big[\textstyle\sum_t G(t) \bigm| Z(t_0), t^* < t_0 + K\big] \\ &= \mathbb{E}\left[\sum_{t=t_0}^{t^*} \mathbb{E}\big[G(t) \bigm| t^*, Z(t_0)\big] \,\middle|\, Z(t_0), t^* < t_0 + K\right] \qquad (51) \\ &\quad + \mathbb{E}\left[\sum_{t=t^*+1}^{t_0+T-1} \mathbb{E}\big[G(t) \bigm| t^*, Z(t_0)\big] \,\middle|\, Z(t_0), t^* < t_0 + K\right]. \qquad (52) \end{aligned}$$

Still by Lemma 8, $\mathbb{E}\Big[\sum_{t=t_0}^{t^*} \mathbb{E}\big[G(t) \bigm| t^*, Z(t_0)\big] \,\Big|\, Z(t_0), t^* < t_0 + K\Big] \leq Kh\|Q_\perp(t_0)\| + KF_1$. The key step in this proof is to bound $\sum_{t=t^*+1}^{t_0+T-1} \mathbb{E}\big[G(t) \bigm| t^*, Z(t_0)\big]$. Consider the following random variables. For any $m \in \mathcal{M}_l$, let

$$\tilde{\mu}_m^l(t) = \begin{cases} \alpha\Big(1 - \frac{\epsilon}{\alpha(M_l+1)}\Big) & \text{if } \sigma_m(t) = 1, \\ 0 & \text{if } \sigma_m(t) = 2, \end{cases} \qquad \tilde{\mu}_m^r(t) = \begin{cases} 0 & \text{if } \sigma_m(t) = 1, \\ \gamma\big(1 - \frac{\epsilon}{\alpha(M_l+1)}\big) & \text{if } \sigma_m(t) = 2. \end{cases}$$

and for any $m \in \mathcal{M}_r$, let

$$\tilde{\mu}_m^l(t) = 0, \ \ \tilde{\mu}_m^r(t) = \gamma - \frac{\epsilon}{M_r(M_l+1)},$$

where ϵ is the heavy-traffic parameter. Note that by the range of ϵ in (45), all the $\tilde{\mu}_m^l(t), \tilde{\mu}_m^r(t)$ with $m \in \mathcal{M}$ are nonnegative. Then we define $\tilde{\mu}(t) = \big(\tilde{\mu}_1(t), \tilde{\mu}_2(t), \cdots, \tilde{\mu}_{M+1}(t)\big)$ as

$$\tilde{\mu}_m(t) = \tilde{\mu}_m^l(t), \ \ m \in \mathcal{M}, \quad \tilde{\mu}_{M+1}(t) = \sum_{m=1}^{M} \tilde{\mu}_m^r(t).$$

For each t with $t^* + 1 \leq t < t_0 + T$, we have the following inequalities based on this auxiliary vector $\tilde{\mu}$. The proofs are provided in Appendix E.

Lemma 9. $\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\mu}(t)\rangle \bigm| t^*, Z(t_0)\big] \leq -\lambda_{\min}\|Q_\perp(t_0)\| + F_2$, *where* $\lambda_{\min}$ *are* F_2 *are positive constants not depending on* ϵ.

Lemma 10. $\mathbb{E}\big[\langle Q(t),\tilde{\mu}(t)\rangle - \langle Q(t),S(t)\rangle \;\big|\; t^*, Z(t_0)\big] \le -\frac{\epsilon}{\sqrt{M_l+1}}\|Q_{\|}(t_0)\| + F_3$, *where* F_3 *is a positive constant not depending on* ϵ.

Lemma 11. $\mathbb{E}\big[\langle c,Q(t)\rangle\langle c,A(t)-S(t)\rangle \;\big|\; t^*, Z(t_0)\big] \ge -\frac{\epsilon}{\sqrt{M_l+1}}\|Q_{\|}(t_0)\| - F_4$, *where* F_4 *is a positive constant not depending on* ϵ.

Utilizing these inequalities yields

$$\mathbb{E}\left[\sum_{t=t^*+1}^{t_0+T-1}\mathbb{E}\big[G(t)\;\big|\;t^*,Z(t_0)\big]\;\middle|\;Z(t_0),t^*<t_0+K\right] \le -\lambda_{\min}(J-1)K\|Q_{\perp}(t_0)\| + T(F_2+F_3+F_4).$$

Combining the bounds and denoting $\Pr\big(t^*\ge t_0+K \;\big|\; Z(t_0)=Z\big)$ by P_K we have

$$\mathbb{E}\big[\Delta V_{\perp}(Z)\;\big|\;Z(t_0)=Z\big] \le \frac{1}{\|Q_{\perp}\|}\Big(\mathbb{E}\big[\textstyle\sum_t G(t)\;\big|\;Z(t_0)=Z\big]+B_2\Big) \le -F_5 + \frac{F_6}{\|Q_{\perp}\|},$$

where $F_5 = P_K hT + (1-P_K)hK - \lambda_{\min}(1-P_K)(J-1)K$ and $F_6 = B_2 + P_K F_1 T + (1-P_K)F_1 K + (1-P_K)(F_2+F_3+F_4)T$. Since $\lim_{K,J\to\infty} F_5 = +\infty$, for any $\theta>0$ there exist large enough K and J such that $-F_5 < -\theta$. Pick any δ with $0<\delta<\theta$ and let $\zeta = \frac{F_6}{\theta-\delta}$. Then $\mathbb{E}\big[\Delta V_{\perp}(Z)\;\big|\;Z(t_0)=Z\big] \le -\delta$ for all Z with $V_{\perp}(Z)\ge\zeta$. This completes the proof that the drift of $V_{\perp}$ satisfies the condition (C1), and the constants δ and ζ do not depend on ϵ.

We revive the superscript $^{(\epsilon)}$ now. Since the Markov chain $\big\{Z^{(\epsilon)}(t), t\ge 0\big\}$ is positive recurrent, Lemma 5 implies that all moments of $V_{\perp}\big(Z^{(\epsilon)}(t)\big) = \|Q_{\perp}^{(\epsilon)}(t)\|$ are bounded in steady state, i.e., there exists a sequence of real numbers $\big\{C_r\colon r\in\mathbb{N}\big\}$ such that for each positive integer r, $\mathbb{E}\big[\|Q_{\perp}^{(\epsilon)}(t)\|^r\big] \le C_r$. The numbers $\big\{C_r\colon r\in\mathbb{N}\big\}$ can be determined from δ,ζ and C. Therefore they do not depend on ϵ. □

5.5 Upper Bound and Heavy-Traffic Optimality

In this subsection, we derive an upper bound on the expectation of the sum of the queue lengths in steady state based on the Lyapunov drift-based moment bounding techniques developed in [20], and we show that this upper bound is asymptotically tight under the heavy-traffic regime.

We have established the state space collapse for the task scheduling system in MapReduce under the proposed algorithm, so the queue length vector in steady state concentrates along a single direction. Enlightened by the way how the queue length in the single server queueing system is bounded, we view the multi-dimensional state space in our problem as a one-dimensional state space along the collapse direction and then set the drift of the Lyapunov function $W_{\|}(Z) = \|Q_{\|}\|^2$ to zero in steady state to obtain an upper bound for the expected queue length along this direction.

Due to the multi service rates in our system, the terms related to service in the Lyapunov drift can not be bounded directly. We consider the *ideal service process* that makes the best use of every machine. This

service process $\big\{S'(t) = \big(S'_1(t), S'_2(t), \cdots, S'_{M+1}(t)\big), t \geq 0\big\}$ is defined by

$$S'_m(t) = \begin{cases} X^l_m(t) & \text{if } m \in \mathcal{M}_l, \\ 0 & \text{if } m \in \mathcal{M}_r, \\ \sum_{m \in \mathcal{M}_r} X^r_m(t) & \text{if } m = M+1, \end{cases}$$

which is coupled with $\{S(t), t \geq 0\}$ in the following way. For each $m \in \mathcal{M}_l$, in the case that $\sigma_m(t) = 1$, $X^l_m(t) = S^l_m(t)$; in the other case that $\sigma_m(t) = 2$, $X^l_m(t) = 1$ if $S^r_m(t) = 1$, and $X^l_m(t) \sim \text{Bern}\big(\frac{\alpha-\gamma}{1-\gamma}\big)$ if $S^r_m(t) = 0$. For each $m \in \mathcal{M}_r$, $X^r_m(t) = S^r_m(t)$. Then it can be seen that for each $m \in \mathcal{M}_l$, $\big\{X^l_m(t), t \geq 0\big\}$ is temporally i.i.d. with $X^l_m(t) \sim \text{Bern}(\alpha)$, and for each $m \in \mathcal{M}_r$, $\big\{X^r_m(t), t \geq 0\big\}$ is temporally i.i.d. with $X^r_m(t) \sim \text{Bern}(\gamma)$. Further, all the processes $\big\{X^l_m(t), t \geq 0\big\}, m \in \mathcal{M}_l$ and $\big\{X^r_m(t), t \geq 0\big\}, m \in \mathcal{M}_r$ are independent, and they are independent from $\{Q(t), t \geq 0\}$ and $\{A(t), t \geq 0\}$. Utilizing this service process, the queue dynamics (14) can be rewritten as

$$Q(t+1) = Q(t) + A(t) - S'(t) + U'(t), \tag{53}$$

where $U'(t) = S'(t) - S(t) + U(t)$. Since the distribution of $S'(t)$ is known, we will use this equivalent queue dynamics to express the Lyapunov drift. Then setting the Lyapunov drift to zero gives the following lemma.

Lemma 12. *For the map task scheduling system in MapReduce, consider any arrival process with an arrival rate vector strictly within the capacity region. Suppose the queueing process is in steady state under the proposed map-scheduling algorithm. Then for any direction c,*

$$2\mathbb{E}\left[\langle c, Q(t)\rangle\langle c, S'(t) - A(t)\rangle\right] = \mathbb{E}\left[\langle c, A(t) - S'(t)\rangle^2\right] + \mathbb{E}\left[\langle c, U'(t)\rangle^2\right] \tag{54}$$

$$+ 2\mathbb{E}\left[\langle c, Q(t) + A(t) - S'(t)\rangle\langle c, U'(t)\rangle\right]. \tag{55}$$

The proof of this lemma is standard, so we omit it due to space limit. Analyzing each term gives the following upper bound, which is asymptotically tight under the heavy-traffic limit.

Theorem 4. (Upper Bound) *For the map-scheduling system in MapReduce, consider an arrival process such that the number of total arrivals at each time slot has expectation $M_l\alpha + M_r\gamma - \epsilon$ and variance $(\sigma^{(\epsilon)})^2$. Let the arrival rate vector be strictly within the capacity region and satisfy the heavy local traffic assumption. Suppose the Markov chain $\big\{Z^{(\epsilon)}(t) = \big(Q^{(\epsilon)}(t), f^{(\epsilon)}(t)\big), t \geq 0\big\}$ is in steady state under the proposed map-scheduling algorithm. Then for any t and any ϵ with*

$$0 < \epsilon < \min\big\{\epsilon_{\max}, M_l\alpha + M_r\gamma, M_r(M_l+1)\gamma\big\},$$

the expectation of the sum of the queue lengths in steady state can be upper bounded as

$$\mathbb{E}\left[\sum_{m=1}^{M+1} Q^{(\epsilon)}_m(t)\right] \leq \frac{(\sigma^{(\epsilon)})^2 + \nu^2}{2\epsilon} + B^{(\epsilon)}, \tag{56}$$

where $B^{(\epsilon)} = o\big(\frac{1}{\epsilon}\big)$, i.e., $\lim_{\epsilon \to 0^+} \epsilon B^{(\epsilon)} = 0$.

Therefore, in the heavy-traffic limit as the arrival rate approaches the service rate from below, assuming

the variance $(\sigma^{(\epsilon)})^2$ *converges to a constant* σ^2 *the upper bound becomes*

$$\limsup_{\epsilon\to 0^+} \epsilon\mathbb{E}\left[\sum_{m=1}^{M+1} Q_m^{(\epsilon)}(t)\right] \le \frac{\sigma^2+\nu^2}{2}. \tag{57}$$

This upper bound under heavy-traffic limit coincides with the lower bound (42)*, which establishes the first moment heavy-traffic optimality of the proposed algorithm.*

Proof. The superscript $^{(\epsilon)}$ is still temporarily omitted. For any time slot t, we analyze each term in Lemma 12 with respect to the collapse direction c defined in (44).

First, the term on the left side of (54) satisfies

$$\begin{aligned}
&\mathbb{E}\big[\langle c, Q(t)\rangle\langle c, S'(t)-A(t)\rangle\big] \\
\overset{(a)}{=}\ &\frac{1}{M_l+1}\mathbb{E}\left[\sum_{m=1}^{M} Q_m(t)+\overline{Q}(t)\right]\cdot\mathbb{E}\left[\sum_{m\in\mathcal{M}_l} X_m^l(t)+\sum_{m\in\mathcal{M}_r} X_m^r(t)-\sum_{\vec{L}\in\mathcal{L}} A_{\vec{L}}(t)\right] \\
\overset{(b)}{=}\ &\frac{\epsilon}{M_l+1}\mathbb{E}\left[\sum_{m=1}^{M} Q_m(t)+\overline{Q}(t)\right],
\end{aligned}$$

where (a) follows from the fact that the sum of the arrivals, the ideal service process and the queue lengths are independent; (b) follows from our assumption about the arrivals and $S'(t)$.

Next, we study the two terms on the right side in (54). Recall that we use ν^2 to denote the variance of $\beta(t)$ in (39). In fact, $\mathrm{Var}\big(\langle c, S'(t)\rangle\big) = \nu^2$ since this random variable has the same distribution as $\beta(t)$. Then

$$\mathbb{E}\big[\langle c, A(t)-S'(t)\rangle^2\big] = \frac{1}{M_l+1}\Big((\sigma^{(\epsilon)})^2+\nu^2+\epsilon^2\Big).$$

Since $Q(t)$ is in steady state, $\mathbb{E}\big[\langle c, A(t)-S'(t)+U'(t)\rangle\big] = \mathbb{E}\big[\langle c, Q(t+1)\rangle - \langle c, Q(t)\rangle\big] = 0$. Thus $\mathbb{E}\big[\langle c, U'(t)\rangle\big] = \mathbb{E}\big[\langle c, S'(t)-A(t)\rangle\big] = \frac{\epsilon}{\sqrt{M_l+1}}$. Meanwhile,

$$\langle c, U'(t)\rangle = \langle c, S'(t)-S(t)+U(t)\rangle \le \langle c, S'(t)\rangle \le \frac{M}{\sqrt{M_l+1}}.$$

Note that by the coupling of $S'(t)$ and $S(t)$, $\langle c, S'(t)-S(t)\rangle \ge 0$. Therefore $\langle c, U'(t)\rangle \ge 0$ and thus

$$\mathbb{E}\left[\langle c, U'(t)\rangle^2\right] \le \frac{M}{\sqrt{M_l+1}}\mathbb{E}\left[\langle c, U'(t)\rangle\right] = \frac{\epsilon M}{M_l+1}.$$

Finally we bound the term (55). By the boundedness of arrivals,

$$\mathbb{E}\left[\langle c, Q(t)+A(t)-S'(t)\rangle\langle c, U'(t)\rangle\right] \le \mathbb{E}\left[\langle c, Q(t)\rangle\langle c, U'(t)\rangle\right] + \frac{\epsilon N A_{\max}}{M_l+1}.$$

To bound the expectation of $\langle c, Q(t)\rangle\langle c, U'(t)\rangle$, we write it as

$$\begin{aligned}
\langle c, Q(t)\rangle\langle c, U'(t)\rangle &= \langle Q(t), U'(t)\rangle - \langle Q_\perp(t), U'_\perp(t)\rangle \\
&= \langle Q(t), S'(t)-S(t)\rangle + \langle Q(t), U(t)\rangle - \langle Q_\perp(t), U'_\perp(t)\rangle.
\end{aligned}$$

The following lemma bounds $\langle Q(t), S'(t) - S(t)\rangle$. The proof is provided in Appendix F.

Lemma 13. $\mathbb{E}\left[\langle Q(t), S'(t) - S(t)\rangle\right] \leq R_1\sqrt{M_l+1}\mathbb{E}\left[\langle c, S'(t) - S(t)\rangle\right]$, *where* $R_1 > 0$ *is a constant.*

This bound indicates that on average, the queue length vector $Q(t)$ only has finite projection in the direction of the difference between $S'(t)$ and $S(t)$. Similarly as in the proof of Lemma 1, $\langle Q(t), U(t)\rangle = \overline{Q}(t)\overline{U}(t) \leq M\overline{U}(t) \leq M\sqrt{M_l+1}\langle c, U(t)\rangle$. Let $R_2 = \max\{R_1, M\}$. Then

$$\begin{aligned}\mathbb{E}\left[\langle Q(t), S'(t) - S(t)\rangle + \langle Q(t), U(t)\rangle\right] &\leq R_2\sqrt{M_l+1}\mathbb{E}\left[\langle c, S'(t) - S(t)\rangle + \langle c, U(t)\rangle\right] \\ &= R_2\sqrt{M_l+1}\mathbb{E}\left[\langle c, U'(t)\rangle\right] \\ &= \epsilon R_2,\end{aligned}$$

To bound the term $-\langle Q_\perp(t), U'_\perp(t)\rangle$, we use the state space collapse result. By the state space collapse theorem, there exists a constant C_2 such that $\mathbb{E}\left[\|Q_\perp(t)\|^2\right] \leq C_2$. Then using the Cauchy-Schwartz inequality and this bound yields,

$$\begin{aligned}\mathbb{E}\left[-\langle Q_\perp(t), U'_\perp(t)\rangle\right] &\leq \sqrt{\mathbb{E}\left[\|Q_\perp(t)\|^2\right]\mathbb{E}\left[\|U'_\perp(t)\|^2\right]} \\ &\leq \sqrt{C_2\mathbb{E}\left[\|U'(t)\|^2\right]}.\end{aligned}$$

The following lemma bounds $\mathbb{E}\left[\|U'(t)\|^2\right]$ in the asymptotic regime $\epsilon \to 0^+$. The proof is provided in Appendix G. By this lemma, $\mathbb{E}\left[-\langle Q_\perp(t), U'_\perp(t)\rangle\right] = o(1)$ as $\epsilon \to 0^+$.

Lemma 14. $\lim_{\epsilon\to 0^+}\mathbb{E}\left[\|U'^{(\epsilon)}(t)\|^2\right] = 0$.

Therefore the term (55) can be bounded as

$$\mathbb{E}\left[\langle c, Q(t) + A(t) - S'(t)\rangle\langle c, U'(t)\rangle\right] \leq \epsilon\left(R_2 + \frac{NA_{\max}}{M_l+1}\right) + o(1).$$

We revive the superscript $^{(\epsilon)}$ now. Combining the inequalities yields

$$\frac{2\epsilon}{M_l+1}\mathbb{E}\left[\sum_{m=1}^{M+1} Q_m^{(\epsilon)}(t)\right] \leq \frac{1}{M_l+1}\left((\sigma^{(\epsilon)})^2 + \nu^2 + \epsilon^2\right) + \frac{\epsilon M}{M_l+1} + 2\epsilon\left(R_2 + \frac{NA_{\max}}{M_l+1}\right) + o(1).$$

Consequently,

$$\mathbb{E}\left[\sum_{m=1}^{M+1} Q_m^{(\epsilon)}(t)\right] \leq \frac{(\sigma^{(\epsilon)})^2 + \nu^2}{2\epsilon} + B^{(\epsilon)},$$

where

$$B^{(\epsilon)} = \frac{\epsilon}{2} + \frac{M}{2} + (M_l+1)R_2 + NA_{\max} + o\left(\frac{1}{\epsilon}\right).$$

Obviously $\lim\limits_{\epsilon\to 0^+}\epsilon B^{(\epsilon)} = 0$, so $B^{(\epsilon)} = o\left(\frac{1}{\epsilon}\right)$. Then the bound (57) for the heavy-traffic limit follows immediately by taking limits on both sides. □

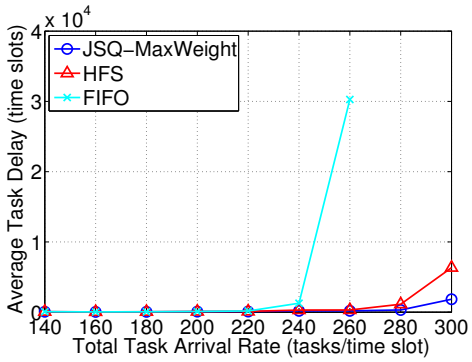


Figure 5: Throughput performance.

6 Simulations

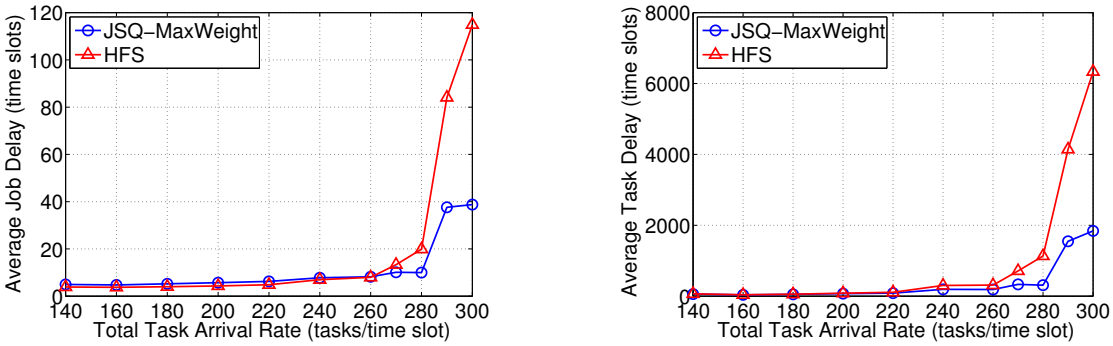
In this section, we use simulations to compare performance of the proposed algorithm (JSQ-MaxWeight) with Hadoop's default FIFO scheduler (FIFO) and the Hadoop Fair Scheduler (HFS) [7], where HFS is the de facto standard. Beyond these comparisons, we also demonstrate the robustness of JSQ-MaxWeight to estimation errors of parameters.

We consider a computing cluster with 400 machines and a dataset distributed uniformly in 360 of the them. Each data chunk has three replicas stored on three different machines uniformly. This simulation models the scenario that the dateset is a database like the user profile database of Facebook, and each job is some manipulation of the data such as searching.

The related simulation parameters are from mimicking real workload analyzed in [25]. According to this analysis, the number of tasks in a job follows a power-law distribution, so we use a truncated Pareto distribution ranging from 10 to $100,000$ with shape parameter 1.9 to generate the number of tasks for each job. Each task processes one data chunk uniformly selected from the dataset. The service time of a task follows a geometric distribution with parameter $\alpha = 0.8$ at a local machine, and with parameter $\gamma = 0.4$ at a remote machine. With this processing capability, the total task arrival rate λ_Σ should be no larger than $360\alpha + 40\gamma = 304$ per time slot. We run the simulations for λ_Σ from 140 to 300 per time slot with 20 interval to evaluate performance.

Throughput Performance Figure 5 compares the throughput region of the three algorithms. The x-axis shows the total task arrival rate λ_Σ, and the y-axis shows the average task delay. A turning point on the curve indicates that λ_Σ is close to the throughput region boundary of the algorithm. Figure5 shows that under FIFO, the system becomes unstable at some λ_Σ between $240 \sim 260$, whereas under JSQ-MaxWeight and HFS the system stays stable for all the considered arrival rates.

Delay Performance Figure 6 compares the delay performance of JSQ-MaxWeight and HFS. FIFO is not included since the job-level policy of FIFO and these two algorithms follow different principle, thus not directly comparable. Also FIFO becomes unstable at a medium load. For each λ_Σ, we calculate the average delay for jobs and tasks departing in steady state. Figure 6 shows that for heavy traffic, JSQ-MaxWeight outperforms HFS on both job delay and task delay, and for light to medium traffic, the two algorithms



(a) Average job delay in steady state.

(b) Average task delay in steady state.

Figure 6: Delay performance.

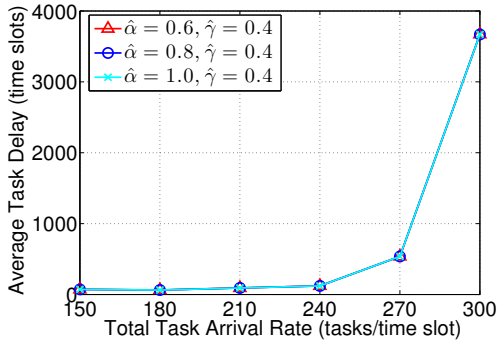


Figure 7: Robustness to estimation errors.

perform similar. Note that the average job delay can be smaller than the average task delay. Most jobs are small jobs, each of which consists of up to several hundreds of tasks, whereas a great portion of tasks come from large jobs. Therefore the average job delay is determined by small jobs, but the average task delay is more related to the tasks in large jobs.

Robustness The MaxWeight scheduling step of JSQ-MaxWeight weights each local queue and the common remote queue by the service rates α and γ, respectively. Therefore, errors in estimating these two parameters may affect the performance. We conducted simulations to evaluate the performance of JSQ-MaxWeight with estimation errors. The actual service rates are $\alpha = 0.8$ and $\gamma = 0.4$. We tested three different settings with different α's in the MaxWeight scheduling step. As shown in Figure 7, the performance under these three sets of estimates are very close, which indicates that the algorithm is robust to estimation errors. The theoretical analysis of the robustness of the system is one of the future research problems we would like to study.

7 Conclusion

In this paper, we studied the map task scheduling problem in MapReduce with data locality. We derived an outer bound on the capacity region and a lower bound on the expected number of backlogged tasks in

steady state. Then we developed a scheduling algorithm constituted by the Join the Shortest Queue policy and the MaxWeight policy. This algorithm achieves the full capacity region and minimizes the expected number of backlogged tasks in the considered heavy-traffic regime. Therefore, the proposed algorithm is both throughput optimal and heavy-traffic optimal. The proof technique was novel since nonpreemptive task execution and random service times were involved. Simulation results were given to evaluate the throughput performance and the delay performance for a large range of total arrival rates.

A Proof of Lemma 1

For any time slot t, $\langle Q(t), U(t)\rangle = \sum_{m=1}^{M} Q_m(t)U_m(t) + \overline{Q}(t)\overline{U}(t)$. By definitions, for any $m \in \mathcal{M}$, $Q_m(t)U_m(t) = 0$. When $\overline{U}(t) > 0$, $\overline{Q}(t)$ must be less than the number of machines scheduled to it, which implies that $\overline{Q}(t) < M$. We also have $\overline{U}(t) \leq M$. Therefore $\overline{Q}(t)\overline{U}(t) < M^2$ always holds. Combing the two parts yields the conclusion. □

B Proof of Lemma 2–Lemma 3

Proof of Lemma 2. Taking conditional expectation yields

$$\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\lambda}\rangle \mid Z(t_0)\big] = \mathbb{E}\Big[\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\lambda}\rangle \mid Z(t)\big] \,\Big|\, Z(t_0)\Big], \tag{58}$$

where (58) follows from the fact that $Q(t)$ and $A(t)$ are conditionally independent from $Z(t_0)$ given $Z(t)$. Using the definitions of the notations and changing the order of summations yield

$$\begin{aligned}
&\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\lambda}\rangle \mid Z(t)\big] \\
&= \mathbb{E}\Bigg[\bigg(\sum_m Q_m(t) \sum_{\vec{L}\colon m\in\vec{L}} A_{\vec{L},m}(t) + \overline{Q}(t)\sum_{\vec{L}\in\mathcal{L}} \overline{A}_{\vec{L}}(t)\bigg) \\
&\quad - \bigg(\sum_m Q_m(t) \sum_{\vec{L}\colon m\in\vec{L}} \lambda_{\vec{L},m} + \overline{Q}(t)\sum_m \sum_{\vec{L}\colon m\notin\vec{L}} \lambda_{\vec{L},m}\bigg) \,\Bigg|\, Z(t)\Bigg] \\
&= \sum_{\vec{L}\in\mathcal{L}} \Bigg\{\mathbb{E}\Bigg[\sum_{m\in\vec{L}} Q_m(t)A_{\vec{L},m}(t) + \overline{Q}(t)\overline{A}_{\vec{L}}(t) \,\Bigg|\, Z(t)\Bigg] \\
&\qquad - \bigg(\sum_{m\in\vec{L}} Q_m(t)\lambda_{\vec{L},m} + \overline{Q}(t)\sum_{m\notin\vec{L}} \lambda_{\vec{L},m}\bigg)\Bigg\}.
\end{aligned} \tag{59}$$

For each type $\vec{L}$, denote

$$Q^*_{\vec{L}}(t) = \min\Big\{\min_{m\in\vec{L}} Q_m(t), \overline{Q}(t)\Big\}. \tag{60}$$

According to the JSQ routing algorithm, $A_{\vec{L}}(t)$ is routed to the shortest one among its three local queues and the common remote queue, so the arrival of type $\vec{L}$ tasks to this queue equals $A_{\vec{L}}(t)$ and the arrivals of

type $\vec{L}$ tasks to the other queues are zero. Thus for each $\vec{L} \in \mathcal{L}$,

$$\sum_{m \in \vec{L}} Q_m(t) A_{\vec{L},m}(t) + \overline{Q}(t) \overline{A}_{\vec{L}}(t) = Q^*_{\vec{L}}(t) A_{\vec{L}}(t),$$

and then

$$\mathbb{E}\left[\sum_{m \in \vec{L}} Q_m(t) A_{\vec{L},m}(t) + \overline{Q}(t) \overline{A}_{\vec{L}}(t) \;\middle|\; Z(t) \right] = Q^*_{\vec{L}}(t) \mathbb{E}\big[A_{\vec{L}}(t) \mid Z(t) \big] = Q^*_{\vec{L}}(t) \lambda_{\vec{L}}.$$

It is easy to see that

$$\sum_{m \in \vec{L}} Q_m(t) \lambda_{\vec{L},m} + \overline{Q}(t) \sum_{m \notin \vec{L}} \lambda_{\vec{L},m} \geq Q^*_{\vec{L}}(t) \left(\sum_{m \in \vec{L}} \lambda_{\vec{L},m} + \sum_{m \notin \vec{L}} \lambda_{\vec{L},m} \right) = Q^*_{\vec{L}}(t) \lambda_{\vec{L}}.$$

Therefore by (59), $\mathbb{E}\big[\langle Q(t), A(t) \rangle - \langle Q(t), \tilde{\lambda} \rangle \mid Z(t) \big] \leq 0$, and then by (58) we can conclude that

$$\mathbb{E}\big[\langle Q(t), A(t) \rangle - \langle Q(t), \tilde{\lambda} \rangle \mid Z(t_0) \big] \leq 0.$$

□

Proof of Lemma 3. By the boundedness of arrivals and service, for any t_0, t and T with $t_0 \leq t < t_0 + T$,

$$\begin{aligned} &Q_m(t_0) - T \leq Q_m(t) \leq Q_m(t_0) + C_A T, \;\; m \in \mathcal{M}, \\ &\overline{Q}(t_0) - MT \leq \overline{Q}(t) \leq \overline{Q}(t_0) + C_A T. \end{aligned} \tag{61}$$

We call this set of inequalities as *the boundedness inequalities*, and they will be used repeatedly in this proof.

Fix a $t_0 \geq 0$ and pick a $T > 0$. By definitions,

$$\begin{aligned} &\mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1} \Big(\langle Q(t), \tilde{\lambda} \rangle - \langle Q(t), S(t) \rangle \Big) \;\middle|\; Z(t_0) \right] \\ &= \sum_{m=1}^{M} \mathbb{E}\left[\sum_t \Big(Q_m(t) \sum_{\vec{L}\colon m \in \vec{L}} \lambda_{\vec{L},m} + \overline{Q}(t) \sum_{\vec{L}\colon m \notin \vec{L}} \lambda_{\vec{L},m} \Big) \;\middle|\; Z(t_0) \right] \\ &\quad - \sum_{m=1}^{M} \mathbb{E}\left[\sum_t \Big(Q_m(t) S^l_m(t) + \overline{Q}(t) S^r_m(t) \Big) \;\middle|\; Z(t_0) \right]. \end{aligned} \tag{62}$$

We refer to the two expectations on the right hand side of the above equation as the arrival term and the service term, respectively. We are going to calculate these two terms for each $m \in \mathcal{M}$.

First let us consider the arrival term. By the boundedness inequalities (61), for each $m \in \mathcal{M}$,

$$\begin{aligned} &\mathbb{E}\left[\sum_t \Big(Q_m(t) \sum_{\vec{L}\colon m \in \vec{L}} \lambda_{\vec{L},m} + \overline{Q}(t) \sum_{\vec{L}\colon m \notin \vec{L}} \lambda_{\vec{L},m} \Big) \;\middle|\; Z(t_0) \right] \\ &\leq T \Big(Q_m(t_0) \sum_{\vec{L}\colon m \in \vec{L}} \lambda_{\vec{L},m} + \overline{Q}(t_0) \sum_{m\colon m \notin \vec{L}} \lambda_{\vec{L},m} \Big) + 2 C_A T^2 \end{aligned}$$

$$\leq T\max\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\}\left(\sum_{\vec{L}\colon m\in\vec{L}}\frac{\lambda_{\vec{L},m}}{\alpha}+\sum_{m\colon m\notin\vec{L}}\frac{\lambda_{\vec{L},m}}{\gamma}\right)+2C_AT^2$$
$$\leq \frac{T}{1+\epsilon}\max\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\}+2C_AT^2,$$

where the last inequality follows from the property (20) of the selected decomposition $\{\lambda_{\vec{L},m}\}$.

Next let us consider the service term. Still by the boundedness inequalities (61), for each $m\in\mathcal{M}$,

$$\mathbb{E}\left[\sum_t\Big(Q_m(t)S_m^l(t)+\overline{Q}(t)S_m^r(t)\Big)\;\Big|\;Z(t_0)\right]$$
$$\geq \mathbb{E}\left[Q_m(t_0)\sum_t S_m^l(t)+\overline{Q}(t_0)\sum_t S_m^r(t)\;\Big|\;Z(t_0)\right]-(M+1)T^2. \tag{63}$$

For each nonnegative integer τ, let

$$N_m(\tau)=\sum_{u=t_0}^{t_0+\tau-1}\Big(I\big(S_m^l(u)>0\big)+I\big(S_m^r(u)>0\big)\Big), \tag{64}$$

where I is the indicator function, and $N_m(0)=0$ by convention. Then $\{N_m(\tau),\tau\geq 0\}$ is a discrete-time counting process. Let $X_m(1)$ denote the occurrence time of the first event of this counting process, and $X_m(n)$ the holding time of the $(n-1)$-th event for $n\geq 2$. The process $\{X_m(n), n\geq 1\}$ is called the *interval time process*. Let $Y_m(n)=\sum_{i=1}^n X_m(i)$ for $n\geq 1$ and $Y_m(0)=0$ by convention. Then $Y_m(n)$ is the occurrence time of the n-th event and satisfies

$$N_m(\tau)=\sup\{n\geq 0\mid Y_m(n)\leq\tau\}.$$

Note that $\{t_0+Y_m(n), n\geq 1\}$ are the time slots that machine m is available after t_0. Consider the scheduling decisions at these time slots and let $H_m(t_0)=\big(\eta_m(t_0+Y_m(n))\colon n\geq 1, n\in\mathbb{N}\big)$. Then

$$\mathbb{E}\left[Q_m(t_0)\sum_t S_m^l(t)+\overline{Q}(t_0)\sum_t S_m^r(t)\;\Big|\;Z(t_0)\right]$$
$$=\mathbb{E}\left[\mathbb{E}\left[Q_m(t_0)\sum_t S_m^l(t)+\overline{Q}(t_0)\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\right]\;\Big|\;Z(t_0)\right]. \tag{65}$$

According to the queue dynamics (14), $\{X_m(n), n\geq 1\}$ are independent given $\{H_m(t_0), Z(t_0)\}$. Let $C_R=\alpha C_AT+\gamma MT$ and $C_L=\alpha T+\gamma C_AT$. Then we consider the following three cases: (1) $\gamma\overline{Q}(t_0)>\alpha Q_m(t_0)+C_R$, (2) $\gamma\overline{Q}(t_0)<\alpha Q_m(t_0)-C_L$ and (3) $\alpha Q_m(t_0)-C_L\leq\gamma\overline{Q}(t_0)\leq\alpha Q_m(t_0)+C_R$. Recall that F_L and F_R are the CDFs of the service times for local tasks and remote tasks respectively in our model. In the following analysis, we will show that

$$\mathbb{E}\left[\frac{1}{\alpha}\sum_t S_m^l(t)+\frac{1}{\gamma}\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\right]\approx T.$$

The intuition is that $\sum_t S_m^l(t)$ and $\sum_t S_m^r(t)$ are the approximate numbers of local tasks and remote tasks

that are served from t_0 to t_0+T-1, respectively, and $1/\alpha$ is the average service time for a local task and $1/\gamma$ is the average service time for a remote task. The challenge of proving this result lies in the fact that $S_m^l(t)$ and $S_m^r(t)$ are not independent. We prove the result by considering large T and Chebyshev's inequality to obtain a concentration result.

- **Case 1.** If $\gamma\overline{Q}(t_0) > \alpha Q_m(t_0) + C_R$, then for any t with $t_0 \le t < t_0 + T$ we have

$$\gamma\overline{Q}(t) - \alpha Q_m(t) \ge \big(\gamma\overline{Q}(t_0) - \gamma MT\big) - \big(\alpha Q_m(t_0) + \alpha C_A T\big) > 0.$$

Therefore, for any t with $t_0 + Y_m(1) \le t < t_0 + T$, machine m is scheduled to the common remote queue and $\overline{Q}(t) > 0$. Then $\eta_m(t_0 + Y_m(n)) = 2$ for $n \ge 1$ and $Y_m(n) \le T-1$, and

$$\begin{aligned}
&\mathbb{E}\bigg[Q_m(t_0)\sum_t S_m^l(t) + \overline{Q}(t_0)\sum_t S_m^r(t) \;\bigg|\; H_m(t_0), Z(t_0)\bigg] \\
&\ge \mathbb{E}\bigg[Q_m(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^l(t) + \overline{Q}(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^r(t) \;\bigg|\; H_m(t_0), Z(t_0)\bigg] \\
&= \mathbb{E}\bigg[\overline{Q}(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^r(t) \;\bigg|\; H_m(t_0), Z(t_0)\bigg] \\
&\ge \overline{Q}(t_0)\mathbb{E}\big[N_m(T) - 1 \;\big|\; H_m(t_0), Z(t_0)\big]. \qquad (66)
\end{aligned}$$

Given $\big\{H_m(t_0), Z(t_0)\big\}$, $\big\{X_m(n), n \ge 1\big\}$ are independent, and $\big\{X_m(n), 1 < n \le N_m(T)+1\big\}$ are i.i.d. with CDF F_R. The CDF of $X_m(1)$ is denoted as F_1 and it satisfies that $F_1(x) = \Pr\big(X \le x \;\big|\; X > \Psi_m(t_0)\big)$ for $x \ge 0$, where X is a random variable with CDF F_L if machine m is scheduled to the local queue, or with CDF F_R if machine m is scheduled to the common remote queue. A *delayed renewal process* is almost a renewal process except that the initial time to the first renewal can have a distribution different from that of other interoccurence times [26]. Consider a delayed renewal process $\big\{\tilde{N}(\tau), \tau \ge 0\big\}$ with the interval time process $\big\{\tilde{X}(n), n \ge 1\big\}$. Suppose that this delayed renewal process is independent from the queueing system. Let $\tilde{X}(1)$ have CDF F_1 and $\tilde{X}(n)$ with $n \ge 2$ have CDF F_R. Then $\tilde{N}(T)$ and $N_m(T)$ have the same distribution given $\big\{H_m(t_0), Z(t_0)\big\}$. By the elementary renewal theorem for delayed renewal processes [26],

$$\lim_{\tau\to\infty} \frac{\mathbb{E}\big[\tilde{N}(\tau)\big]}{\tau} = \gamma. \qquad (67)$$

Therefore, for any ϵ_1 with $0 < \epsilon_1 < \gamma$ there exists $\tau_m^{(1)} > 0$ such that for any $\tau \ge \tau_m^{(1)}$,

$$\mathbb{E}\big[\tilde{N}(\tau)\big] \ge (\gamma - \epsilon_1)\tau.$$

Consider any ϵ_1' with $0 < \epsilon_1' < 1 - \frac{\epsilon_1}{\gamma}$ and let $T \ge \max\Big\{\tau_m^{(1)}, \frac{1}{\gamma\epsilon_1'}\Big\}$. Then

$$\mathbb{E}\big[N_m(T) \;\big|\; H_m(t_0), Z(t_0)\big] = \mathbb{E}\big[\tilde{N}(T)\big] \ge (\gamma - \epsilon_1)T.$$

Hence by (66),

$$\begin{aligned}
&\mathbb{E}\Big[Q_m(t_0)\sum_t S_m^l(t) + \overline{Q}(t_0)\sum_t S_m^r(t) \;\Big|\; H_m(t_0), Z(t_0)\Big] \\
&\geq \big((\gamma-\epsilon_1)T-1\big)\overline{Q}(t_0) \\
&\geq \Big(1-\frac{\epsilon_1}{\gamma}-\epsilon_1'\Big)T\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\},
\end{aligned}$$

where the last equality follows from the choice of T and the assumption of this case.

- **Case 2.** If $\gamma\overline{Q}(t_0) < \alpha Q_m(t_0) - C_L$, then for any t with $t_0 \leq t < t_0 + T$ we have

$$\alpha Q_m(t) - \gamma\overline{Q}(t) \geq \big(\alpha Q_m(t_0) - \alpha T\big) - \big(\gamma\overline{Q}(t_0) + \gamma C_A T\big) > 0.$$

Therefore, for any t with $t_0+Y_m(1) \leq t < t_0+T$, machine m is scheduled to the local queue and $Q_m(t) > 0$. Then $\eta_m(t_0 + Y_m(n)) = 0$ for $n \geq 1$ and $Y_m(n) \leq T-1$, and

$$\begin{aligned}
&\mathbb{E}\Big[Q_m(t_0)\sum_t S_m^l(t) + \overline{Q}(t_0)\sum_t S_m^r(t) \;\Big|\; H_m(t_0), Z(t_0)\Big] \\
&\geq \mathbb{E}\Bigg[Q_m(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^l(t) + \overline{Q}(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^r(t) \;\Bigg|\; H_m(t_0), Z(t_0)\Bigg] \\
&= \mathbb{E}\Bigg[Q_m(t_0)\sum_{t=t_0+Y_m(1)}^{t_0+T-1} S_m^l(t) \;\Bigg|\; H_m(t_0), Z(t_0)\Bigg] \\
&\geq Q_m(t_0)\mathbb{E}\big[N_m(T)-1 \;\big|\; H_m(t_0), Z(t_0)\big]. \qquad (68)
\end{aligned}$$

Similarly as in Case 1, for any ϵ_2 with $0 < \epsilon_2 < \alpha$ and any ϵ_2' with $0 < \epsilon_2' < 1 - \frac{\epsilon_2}{\alpha}$, there exists $\tau_m^{(2)} > 0$ such that for any $T \geq \max\Big\{\tau_m^{(2)}, \frac{1}{\alpha\epsilon_2'}\Big\}$,

$$\mathbb{E}\big[N_m(T) \;\big|\; H_m(t_0), Z(t_0)\big] \geq (\alpha-\epsilon_2)T.$$

Hence by (68),

$$\begin{aligned}
&\mathbb{E}\Big[Q_m(t_0)\sum_t S_m^l(t) + \overline{Q}(t_0)\sum_t S_m^r(t) \;\Big|\; H_m(t_0), Z(t_0)\Big] \\
&\geq \big((\alpha-\epsilon_2)T-1\big)Q_m(t_0) \\
&\geq \Big(1-\frac{\epsilon_2}{\alpha}-\epsilon_2'\Big)T\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\},
\end{aligned}$$

where the last equality follows from the choice of T and the assumption of this case.

- **Case 3.** If $\alpha Q_m(t_0) - C_L \leq \gamma\overline{Q}(t_0) \leq \alpha Q_m(t_0) + C_R$, then

$$\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\} \leq \gamma\overline{Q}(t_0) + C_L,$$

and thus

$$\begin{aligned}
&\mathbb{E}\Big[Q_m(t_0)\sum_t S_m^l(t)+\overline{Q}(t_0)\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\Big]\\
&\geq \mathbb{E}\Big[\Big(\frac{\gamma}{\alpha}\overline{Q}(t_0)-\frac{C_R}{\alpha}\Big)\sum_t S_m^l(t)+\overline{Q}(t_0)\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\Big]\\
&\geq \max\big\{\alpha Q_m(t_0),\gamma\overline{Q}(t_0)\big\}\mathbb{E}\Big[\frac{1}{\alpha}\sum_t S_m^l(t)+\frac{1}{\gamma}\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\Big]-T\Big(\frac{C_L}{\gamma}+\frac{C_R}{\alpha}\Big). \qquad (69)
\end{aligned}$$

Given $\big\{H_m(t_0),Z(t_0)\big\}$, we consider a function $g\colon \mathbb{N}\to\mathbb{R}$ defined as follows

$$\begin{aligned}
g(n)=\sum_{i=1}^{n}\Big(&\frac{1}{\alpha}I\big(\eta_m(t_0+Y_m(i-1))=0\big)+I\big(\eta_m(t_0+Y_m(i-1))=1\big)\\
&+\frac{1}{\gamma}I\big(\eta_m(t_0+Y_m(i-1))=2\big)+I\big(\eta_m(t_0+Y_m(i-1))=3\big)\Big), \qquad (70)
\end{aligned}$$

where I is the indicator function and $g(0)=0$ by convention. Then $g(\cdot)$ is an increasing function and $n\leq g(n)\leq \frac{n}{\gamma}$. We first derive two properties of $g(\cdot)$, which will be used to further bound (69).

First we note that $g(n)-g(1)=\sum_{i=2}^{n}\mathbb{E}\big[X_m(i)\;\big|\;H_m(t_0),Z(t_0)\big]$. Let random variables X_L and X_R have CDFs F_L and F_R respectively. Let $C_3=\max\big\{E_L,E_R\big\}$, where $E_L=\mathbb{E}\big[X_L\;\big|\;X_L>\Psi_m(t_0)\big]$ and $E_R=\mathbb{E}\big[X_R\;\big|\;X_R>\Psi_m(t_0)\big]$. Then C_3 is a constant such that $\mathbb{E}\big[X_m(1)\;\big|\;H_m(t_0),Z(t_0)\big]\leq C_3$, and thus for any $n\geq 1$,

$$g(n)+C_3\geq\sum_{i=1}^{n}\mathbb{E}\big[X_m(i)\;\big|\;H_m(t_0),Z(t_0)\big]. \qquad (71)$$

Next we note that

$$g(n)=\mathbb{E}\Big[\frac{1}{\alpha}\sum_t S_m^l(t)+\frac{1}{\gamma}\sum_t S_m^r(t)\;\Big|\;N_m(T)=n,H_m(t_0),Z(t_0)\Big].$$

Therefore,

$$\mathbb{E}\Big[\frac{1}{\alpha}\sum_t S_m^l(t)+\frac{1}{\gamma}\sum_t S_m^r(t)\;\Big|H_m(t_0),Z(t_0)\Big]=\sum_{n=1}^{\infty}g(n)\Pr\big(N_m(T)=n\;\big|H_m(t_0),Z(t_0)\big).$$

Now we bound the expectation in (69). For any ϵ_3 with $0<\epsilon_3<1$, let

$$T\geq\frac{1}{1+\gamma(1-\epsilon_3)}\Big(\frac{1}{\gamma\epsilon_3}+\frac{C_3}{\epsilon_3}\Big)$$

and choose n_0 such that

$$g(n_0-1)<(1-\epsilon_3)T\leq g(n_0). \qquad (72)$$

Then

$$\mathbb{E}\Big[\frac{1}{\alpha}\sum_t S_m^l(t)+\frac{1}{\gamma}\sum_t S_m^r(t)\;\Big|\;H_m(t_0),Z(t_0)\Big]$$

$$\geq \sum_{n=n_0}^{\infty} g(n) \Pr\big(N_m(T) = n \;\big|\; H_m(t_0), Z(t_0)\big)$$
$$\geq (1-\epsilon_3)T \Pr\big(N_m(T) \geq n_0 \;\big|\; H_m(t_0), Z(t_0)\big), \tag{73}$$

where for the last inequality we have used the fact that $g(\cdot)$ is an increasing function. We are going to derive a lower bound on the probability $\Pr\big(N_m(T) \geq n_0 \;\big|\; H_m(t_0), Z(t_0)\big)$ in (73). Since the following calculations are all conditioned on $\big\{H_m(t_0), Z(t_0)\big\}$, we temporarily omit the condition in the expressions for conciseness. We will revive the condition after these calculations. Consider the interval time process $\big\{X_m(n), n \geq 1\big\}$. Then

$$\Pr\big(N_m(T) \geq n_0\big) = \Pr\left(\sum_{n=1}^{n_0} X_m(n) \leq T\right).$$

By the choice of n_0 in (72),

$$g(n_0) \leq g(n_0 - 1) + \frac{1}{\gamma} < (1-\epsilon_3)T + \frac{1}{\gamma}. \tag{74}$$

Since $T \geq \frac{1}{1+\gamma(1-\epsilon_3)}\Big(\frac{1}{\gamma\epsilon_3} + \frac{C_3}{\epsilon_3}\Big)$, this implies that $T - g(n_0) - C_3 > \epsilon_3 T - \frac{1}{\gamma} - C_3 > 0$. Further, since $T - g(n_0) > C_3 \geq 0$ and $g(n_0) \geq n_0$, we have $T > n_0$ and $\epsilon_3 T - \frac{1}{\gamma} - C_3 > \epsilon_3 n_0 - \frac{1}{\gamma} - C_3$. Then by (72), $(1-\epsilon_3)T \leq g(n_0) \leq \frac{1}{\gamma} n_0$ and hence $\epsilon_3 n_0 - \frac{1}{\gamma} - C_3 \geq \epsilon_3\gamma(1-\epsilon_3)T - \frac{1}{\gamma} - C_3 > 0$. The property (71) of $g(\cdot)$ indicates that $\sum_{n=1}^{n_0} \mathbb{E}\big[X_m(n)\big] \leq g(n) + C_3$. Therefore combing these inequalities gives

$$T - \sum_{n=1}^{n_0} \mathbb{E}\big[X_m(n)\big] \geq T - g(n) - C_3 > \epsilon_3 n_0 - \frac{1}{\gamma} - C_3 > 0. \tag{75}$$

Then the probability can be further calculated as

$$\Pr\big(N_m(T) \geq n_0\big) = \Pr\left(\frac{1}{n_0}\sum_{n=1}^{n_0}\big(X_m(n) - \mathbb{E}\big[X_m(n)\big]\big) \leq \frac{1}{n_0}\left(T - \sum_{n=1}^{n_0}\mathbb{E}\big[X_m(n)\big]\right)\right)$$
$$\geq 1 - \Pr\left(\left|\frac{1}{n_0}\sum_{n=1}^{n_0}\big(X_m(n) - \mathbb{E}\big[X_m(n)\big]\big)\right| > \frac{1}{n_0}\Big(\epsilon_3 n_0 - \frac{1}{\gamma} - C_3\Big)\right).$$

Suppose F_L and F_R have variances σ_L^2 and σ_R^2 respectively, and let $\sigma^2 = \max\big\{\sigma_L^2, \sigma_R^2, 1\big\}$. Applying Chebyshev's inequality yields

$$\Pr\big(N_m(T) \geq n_0\big) \geq 1 - \frac{\frac{1}{n_0}\sigma^2}{\frac{1}{n_0^2}\Big(\epsilon_3 n_0 - \frac{1}{\gamma} - C_3\Big)^2}$$
$$= 1 - \frac{n_0\sigma^2}{\epsilon_3^2\Big(n_0 - \frac{1}{\epsilon_3\gamma} - \frac{C_3}{\epsilon_3}\Big)^2}$$
$$= 1 - h(n_0).$$

Then $\lim_{n\to\infty} h(n) = 0$. Thus for any $\epsilon_4 > 0$, there exists $N_0 > 0$ such that for any $n \geq N_0$, $h(n) < \epsilon_4$.

Let

$$\tau_m^{(3)} = \max\Big\{ \frac{1}{1+\gamma(1-\epsilon_3)}\Big(\frac{1}{\gamma\epsilon_3} + \frac{C_3}{\epsilon_3}\Big), \frac{N_0}{\gamma(1-\epsilon_3)} \Big\}.$$

Then if $T \geq \tau_m^{(3)}$, the choice of n_0 in (72) ensures that $n_0 \geq \gamma(1-\epsilon_3)T \geq N_0$. In this case $\Pr\big(N_m(T) \geq n_0\big) \geq 1-\epsilon_4$. Reviving the condition $\big\{H_m(t_0), Z(t_0)\big\}$ gives the lower bound $\Pr\big(N_m(T) \geq n_0 \;\big|\; H_m(t_0), Z(t_0)\big) \geq 1-\epsilon_4$.

To sum up, for any ϵ_3 with $0 < \epsilon_3 < 1$ and any ϵ_4 with $0 < \epsilon_4 < 1$, there exists $\tau_m^{(3)}$ defined as above, such that for any $T \geq \tau_m^{(3)}$,

$$\mathbb{E}\bigg[\frac{1}{\alpha}\sum_t S_m^l(t) + \frac{1}{\gamma}\sum_t S_m^r(t) \;\bigg|\; H_m(t_0), Z(t_0)\bigg] \geq (1-\epsilon_3)(1-\epsilon_4)T.$$

Hence by (69),

$$\begin{aligned}
&\mathbb{E}\bigg[Q_m(t_0)\sum_t S_m^l(t) + \overline{Q}(t_0)\sum_t S_m^r(t) \;\bigg|\; H_m(t_0), Z(t_0)\bigg] \\
&\geq (1-\epsilon_3)(1-\epsilon_4)T\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\} - T\Big(\frac{C_L}{\gamma} + \frac{C_R}{\alpha}\Big).
\end{aligned}$$

Now we combine the three cases. For any ϵ' with $0 < \epsilon' < 1$, choose $\epsilon_1, \epsilon_1', \epsilon_2, \epsilon_2', \epsilon_3, \epsilon_4$ such that $1 - \frac{\epsilon_1}{\gamma} - \epsilon_1' = 1 - \frac{\epsilon_2}{\alpha} - \epsilon_2' = (1-\epsilon_3)(1-\epsilon_4) = 1-\epsilon'$. Let $\tau_m = \max\Big\{\tau_m^{(1)}, \tau_m^{(2)}, \tau_m^{(3)}, \frac{1}{\gamma\epsilon_1'}, \frac{1}{\alpha\epsilon_2'}\Big\}$. Then for any $T \geq \tau_m$, utilizing (63) and (65) gives

$$\begin{aligned}
&\mathbb{E}\bigg[\sum_t \Big(Q_m(t)S_m^l(t) + \overline{Q}(t)S_m^r(t)\Big) \;\bigg|\; Z(t_0)\bigg] \\
&\geq (1-\epsilon')T\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\} - T\Big(\frac{C_L}{\gamma} + \frac{C_R}{\alpha}\Big) - (M+1)T^2.
\end{aligned}$$

Now we have bounded both the arrival term and the service term for each $m \in \mathcal{M}$ in (62). Choose ϵ' such that $\epsilon' < \frac{\epsilon}{1+\epsilon}$ and let $T_0 = \max\big\{\tau_m \;\big|\; m \in \mathcal{M}\big\}$. Then for any $T \geq T_0$,

$$\begin{aligned}
&\mathbb{E}\bigg[\sum_{t=t_0}^{t_0+T-1}\Big(\langle Q(t), \tilde{\lambda}\rangle - \langle Q(t), S(t)\rangle\Big) \;\bigg|\; Z(t_0)\bigg] \\
&\leq \Big(\frac{1}{1+\epsilon} - (1-\epsilon')\Big)T\sum_{m=1}^{M}\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\} + C_1,
\end{aligned}$$

where $\frac{1}{1+\epsilon} - (1-\epsilon') < 0$ and $C_1 = 2MC_AT^2 + MT\Big(\frac{C_L}{\gamma} + \frac{C_R}{\alpha}\Big) + M(M+1)T^2$ is a positive constant. For each $m \in \mathcal{M}$,

$$\begin{aligned}
\max\big\{\alpha Q_m(t_0), \gamma\overline{Q}(t_0)\big\} &\geq \alpha Q_m(t_0)\frac{M\gamma}{\alpha+M\gamma} + \gamma\overline{Q}(t_0)\frac{\alpha}{\alpha+M\gamma} \\
&= \frac{M\alpha\gamma}{\alpha+M\gamma}\Big(Q_m(t_0) + \frac{1}{M}\overline{Q}(t_0)\Big).
\end{aligned}$$

Let $\theta = \left(\frac{\epsilon}{1+\epsilon} - \epsilon'\right)\frac{M\alpha\gamma}{\alpha+M\gamma}$ and then by the choice of ϵ' we have $\theta > 0$. Therefore

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=t_0}^{t_0+T-1}\Big(\langle Q(t),\tilde{\lambda}\rangle - \langle Q(t),S(t)\rangle\Big)\,\middle|\, Z(t_0)\right] \\
&\le \Big(\frac{1}{1+\epsilon} - (1-\epsilon')\Big)T\frac{M\alpha\gamma}{\alpha+M\gamma}\left(\sum_{m=1}^{M} Q_m(t_0) + \overline{Q}(t_0)\right) + C_1 \\
&= -\theta T\|Q(t_0)\|_1 + C_1,
\end{aligned}$$

which finishes the proof. □

C Proof of Lemma 4

We are going to prove the existence by constructing a decomposition that meets the conditions. We model the relationship between the task types and the machines by a bipartite graph $\mathcal{G} = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$. For each task type $\vec{L}$, we put a vertex x in $\mathcal{X}$ and give x a budget $b(x) = \lambda_{\vec{L}}$. For each machine m, we put a vertex y in $\mathcal{Y}$. If machine m is local for type $\vec{L}$, we put an edge xy in $\mathcal{E}$. Figure 8 shows an example of such a graph. For any vertex v in the graph, let $\mathcal{N}(v)$ denote the set of its neighbor vertices, and for any vertex set $\mathcal{V}$ let $\mathcal{N}(\mathcal{V}) = \cup_{v\in\mathcal{V}}\mathcal{N}(v)$. Consider a weight function

$$\begin{aligned}
w\colon \qquad & \mathcal{E} \to [0,+\infty) \\
& xy \mapsto w(xy)
\end{aligned}$$

and let $w(x) = \sum_{y\in\mathcal{N}(x)} w(xy)$. If a weight function w satisfies that for any $x \in \mathcal{X}$, $w(x) \le b(x)$, it is said to be a *proper weight function*. In the rest of this proof we only consider proper weight functions. To prove the lemma, it suffices to find a weight function w such that

$$\forall x \in \mathcal{X}, y \in \mathcal{Y}, \quad w(xy) > \kappa, \tag{76}$$

$$\forall x \in \mathcal{X}, \quad b(x) - w(x) > \kappa, \tag{77}$$

$$\forall y \in \mathcal{Y}, \quad w(y) = \alpha - \frac{\epsilon}{M_l + 1}. \tag{78}$$

First we obtain a proper weight function w such that for any $y \in \mathcal{Y}$, $w(y) = \alpha + \frac{\kappa'}{M_l}$, where

$$\kappa' = \min\left\{\min_{\mathcal{H}\subseteq\mathcal{M}_l}\left\{\sum_{\vec{L}\in\mathcal{N}(\mathcal{H})}\lambda_{\vec{L}} - |\mathcal{H}|\alpha\right\}, \frac{\alpha}{3|\mathcal{L}|}, \frac{\min_{\vec{L}\in\mathcal{L}}\lambda_{\vec{L}}}{12}\right\}. \tag{79}$$

By the local heavy traffic assumption, $\kappa' > 0$, and for any $\mathcal{H} \subseteq \mathcal{M}_l$,

$$\sum_{\vec{L}\in\mathcal{N}(\mathcal{H})}\lambda_{\vec{L}} \ge |\mathcal{H}|\alpha + \kappa'. \tag{80}$$

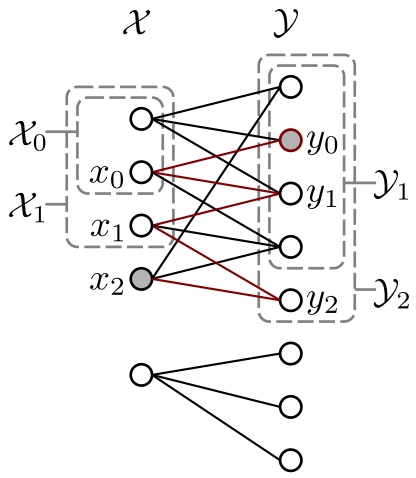


Figure 8: Example bipartite graph representing the relationship between task types and machines.

Once this weight function is obtained, the desired weight function can be constructed from it. To obtain this weight function, we take the following steps.

- Let $\mathcal{W}$ be a set of proper weight functions such that for any $w \in \mathcal{W}$, we have that for any $y \in \mathcal{Y}$, $w(y) \leq \alpha + \frac{\kappa'}{M_l}$. Then $\mathcal{W}$ is nonempty since the zero function is in it.
- We have the following claim.

  Claim 1. *For any weight function $w \in \mathcal{W}$, if there exists $y_0 \in \mathcal{Y}$ with $w(y_0) < \alpha + \frac{\kappa'}{M_l}$, then there exists a path $\mathcal{P} = y_0 x_0 y_1 x_1 \cdots y_k x_k$ such that $w(x_i y_{i+1}) > 0$ and $b(x_i) - w(x_i) = 0$ for $i = 0, \cdots, k-1$, and $b(x_k) - w(x_k) > 0$.*

  Proof of the Claim. If there exists $x_0 \in \mathcal{N}(y_0)$ such that $b(x_0) - w(x_0) > 0$, then let $\mathcal{P} = y_0 x_0$ and we are done. Otherwise for any $x \in \mathcal{N}(y_0)$ we have $b(x) - w(x) = 0$. Consider the sets $\mathcal{X}_0 = \mathcal{N}(y_0)$ and $\mathcal{Y}_1 = \mathcal{N}(\mathcal{X}_0)$. Let $\mathcal{Z} = \{y \in \mathcal{Y}_1 \mid \exists x \in \mathcal{N}(y), \text{ s.t. } x \notin \mathcal{X}_0\} \subseteq \mathcal{Y}_1$, i.e., the set of vertices that have neighbors outside $\mathcal{X}_0$. See the illustration in Figure 8. Then $\mathcal{Z}$ is nonempty and there exist $x_0 \in \mathcal{X}_0$ and $y_1 \in \mathcal{Z}$ such that $w(x_0 y_1) > 0$, since if this is not case then

  $$\sum_{x \in \mathcal{X}_0} b(x) = \sum_{x \in \mathcal{X}_0} w(x) = \sum_{y \in \mathcal{Y}_1 \setminus \mathcal{Z}} w(y) < |\mathcal{Y}_1 \setminus \mathcal{Z}| \Big(\alpha + \frac{\kappa'}{M_l}\Big) \leq |\mathcal{Y}_1 \setminus \mathcal{Z}|\alpha + \kappa', \tag{81}$$

  where the strict inequality follows from that $y_0 \in \mathcal{Y}_1 \setminus \mathcal{Z}$ has $w(y_0) < \alpha + \frac{\kappa'}{M_l}$, and for any $y \in \mathcal{Y}_1 \setminus \mathcal{Z}$, $w(y) \leq \alpha + \frac{\kappa'}{M_l}$. However, this contradicts the local heavy traffic assumption in (80). Thus there exist $x_0 \in \mathcal{X}_0$ and $y_1 \in \mathcal{Z}$ such that $w(x_0 y_1) > 0$. If there exists $x_1 \in \mathcal{N}(y_1)$ such that $b(x_1) - w(x_1) > 0$, then let $\mathcal{P} = y_0 x_0 y_1 x_1$ and we are done. Otherwise let $\mathcal{X}_1 = \mathcal{N}(\{y_0, y_1\})$ and $\mathcal{Y}_2 = \mathcal{N}(\mathcal{X}_1)$. Then using the same arguments we can find $x_1 \in \mathcal{X}_1$ and $y_2 \in \mathcal{Y}_2$ such that y_2 has neighbors outside $\mathcal{X}_1$ and $w(x_1 y_2) > 0$, where x_1 can be a neighbor of y_0 as well as a neighbor of y_1. Still if there exists $x_2 \in \mathcal{N}(y_2)$ such that $b(x_2) - w(x_2) > 0$, then let $\mathcal{P} = y_0 x_0 y_1 x_1 y_2 x_2$ if $(x_1, y_1) \in \mathcal{E}$, or let $\mathcal{P} = y_0 x_1 y_2 x_2$ if $x_1 y_0 \in \mathcal{E}$ and we are done. Otherwise we continue to consider $\mathcal{X}_2 = \mathcal{N}(\{y_0, y_1, y_2\})$ and $\mathcal{Y}_3 = \mathcal{N}(\mathcal{X}_2)$. This procedure will end

in finite steps for the following reasons. In the connected component $\big(\mathcal{X}_C, \mathcal{Y}_C, \mathcal{E}_C\big)$ that contains y_0, there exists at least one $x \in \mathcal{X}_C$ such that $b(x) - w(x) > 0$, since otherwise

$$\sum_{x\in\mathcal{X}_C} b(x) = \sum_{x\in\mathcal{X}_C} w(x) = \sum_{y\in\mathcal{Y}_C} w(y) < |\mathcal{Y}_C|\Big(\alpha + \frac{\kappa'}{M_l}\Big), \tag{82}$$

which contradicts the local heavy traffic assumption again. The procedure results in a sequence $\mathcal{X}_0 \subsetneq \mathcal{X}_1 \subsetneq \mathcal{X}_2 \subsetneq \cdots$ in $\mathcal{X}_C$, so it will end when the sequence hits some $x \in \mathcal{X}_C$ with $b(x) - w(x) > 0$, which takes no more than $|\mathcal{X}_C|$ steps. Therefore the claim holds. □

- Consider the weight function $w \in \mathcal{W}$ such that $\min_{y\in\mathcal{Y}} w(y)$ is maximized. Then w satisfies that for any $y \in \mathcal{Y}$, $w(y) = \alpha + \frac{\kappa'}{M_l}$. We prove this by contradiction. If w does not satisfy the condition above, let $y_0 \in \arg\min_{y\in\mathcal{Y}} w(y)$. Then $w(y_0) < \alpha + \frac{\kappa'}{M_l}$. By Claim 1, there exists a path $\mathcal{P} = y_0x_0y_1x_1\cdots y_kx_k$ such that $w(x_iy_{i+1}) > 0$ and $b(x_i) - w(x_i) = 0$ for $i = 0, \cdots, k-1$, and $b(x_k) - w(x_k) > 0$. Let $\kappa'' = \min\big\{w(x_0y_1), w(x_1y_2), \cdots, w(x_{k-1}y_k), b(x_k) - w(x_k)\big\}$. Since $k \le M$ is finite, $\kappa'' > 0$. Modify w to get another weight function $\tilde{w}$ as follows

$$\tilde{w}(xy) = \begin{cases} w(xy) + \kappa'' & \text{if } x = x_i, y = y_i, \text{ where } i = 0, \cdots k, \\ w(xy) - \kappa'' & \text{if } x = x_i, y = y_{i+1}, \text{ where } i = 0, \cdots k-1, \\ w(xy) & \text{otherwise.} \end{cases} \tag{83}$$

Then $\tilde{w}(xy) \ge 0$ for any $xy \in \mathcal{E}$ by the definition of κ''. For any $x \in \mathcal{X}$, $b(x) - \tilde{w}(x) = b(x) - w(x) \ge 0$ if $x \ne x_k$, and $b(x_k) - \tilde{w}(x_k) = b(x_k) - w(x_k) - \kappa'' \ge 0$. Thus $\tilde{w}$ is a proper weight function. For any $y \in \mathcal{Y}$, $\tilde{w}(y) = w(y)$ if $y \ne y_0$, and $\tilde{w}(y_0) = w(y_0) + \kappa'' > w(y_0)$. Using similar method we further modify $\tilde{w}$ according to other vertices in $\arg\min_{y\in\mathcal{Y}} w(y)$ and finally get a proper weight function w^*. Then $\min_{y\in\mathcal{Y}} w^*(y) > \min_{y\in\mathcal{Y}} w(y)$, which contradicts the assumption that w maximizes $\min_{y\in\mathcal{Y}} w(y)$.

Next we modify the weight function w to make it satisfy the conditions (76)–(78). Consider any ϵ with $0 \le \epsilon \le \epsilon_{\max}$, where

$$\frac{\epsilon_{\max}}{M_l + 1} = \min\left\{\frac{\alpha}{3|\mathcal{L}|}, \frac{\min_{\vec{L}\in\mathcal{L}} \lambda_{\vec{L}}}{12}\right\}. \tag{84}$$

- We first modify w to w' such that w' satisfies condition (78). Due to the modification procedure w' is a proper weight function and for each $x \in \mathcal{X}$, $b(x) - w'(x) \ge \frac{\kappa'}{|\mathcal{L}|M_l}$. Let

$$d = \min\left\{\frac{\alpha}{|\mathcal{L}|}, \frac{\min_{\vec{L}\in\mathcal{L}} \lambda_{\vec{L}}}{4}\right\}. \tag{85}$$

Then $\frac{\kappa'}{M_l} \le \frac{d}{3}$ and $\frac{\epsilon}{M_l+1} \le \frac{d}{3}$. Consider $y \in \mathcal{Y}$. Then for any edge xy with $w(xy) \ge d$, let $w'(xy) = w(xy) - \frac{\kappa'}{|\mathcal{N}(y)|M_l} - \frac{\epsilon}{|\mathcal{N}(y)|(M_l+1)}$. Such edge exists since $\sum_{x\in\mathcal{N}(y)} w(xy) = w(y) = \alpha + \frac{\kappa'}{M_l} \ge \alpha$, and by the pigeon hole principle, there exists an edge xy such that $w(xy) \ge \frac{\alpha}{|\mathcal{N}(y)|} \ge d$. Then $w'(xy)$ is still nonnegative due to the definition of κ' and the range of ϵ. On other edges let the values of w' and w be equal. We do this modification for each $y \in \mathcal{Y}$. Then it is easy to verify that w' is still a proper weight function and for each $y \in \mathcal{Y}$, $w'(y) = \alpha - \frac{\epsilon}{M_l+1}$. We claim that for each $x \in \mathcal{X}$, $b(x) - w'(x) \ge \frac{\kappa'}{|\mathcal{L}|M_l}$. Consider any $x \in \mathcal{X}$. Then again by the pigeon hole principle, either $b(x) - w(x) \ge \frac{\lambda_{\vec{L}}}{4}$ or there exists an

edge xy such that $w(xy) \geq \frac{\lambda_{\bar{L}}}{4}$. If $b(x) - w(x) \geq \frac{\lambda_{\bar{L}}}{4}$, then $b(x) - w'(x) \geq \frac{\lambda_{\bar{L}}}{4} \geq \frac{\kappa'}{|\mathcal{L}|M_l}$ since $w'(x) \leq w(x)$. If there exists xy such that $w(xy) \geq \frac{\lambda_{\bar{L}}}{4} \geq d$, then $w'(xy) \leq w(xy) - \frac{\kappa'}{|\mathcal{N}(y)|M_l} \leq w(xy) - \frac{\kappa'}{|\mathcal{L}|M_l}$. Thus $b(x) - w'(x) \geq b(x) - w(x) + \frac{\kappa'}{|\mathcal{L}|M_l} \geq \frac{\kappa'}{|\mathcal{L}|M_l}$. Hence the claim holds.

- Next we modify w' to w'' such that w'' satisfies conditions (76)–(78). First based on w' we construct a weight function w_1 by considering an $x \in \mathcal{X}$. For each edge xy with $w'(xy) < \frac{\kappa'}{4|\mathcal{L}|M_l}$, let $w_1(xy) = \frac{\kappa'}{4|\mathcal{L}|M_l}$. For each such edge xy, there exists $x' \in \mathcal{X}$ such that $w'(x'y) \geq \frac{1}{|\mathcal{L}|}\left(\alpha - \frac{\epsilon}{M_l+1}\right) \geq \frac{2\alpha}{3|\mathcal{L}|} \geq \frac{2\kappa'}{|\mathcal{L}|M_l}$. To keep $w'(y)$ unchanged, let $w_1(x'y) = w'(x'y) - \left(\frac{\kappa'}{4|\mathcal{L}|M_l} - w'(xy)\right)$. On other edges let the values of w_1 and w' be equal. Then it is easy to verify that w_1 is still a proper weight function and for each $y \in \mathcal{Y}$, $w_1(y) = \alpha - \frac{\epsilon}{M_l+1}$. We can also see that for the considered vertex x, $b(x) - w_1(x) \geq b(x) - w'(x) - \frac{3\kappa'}{4|\mathcal{L}|M_l} \geq \frac{\kappa'}{4|\mathcal{L}|M_l}$, and for each edge xy, $w_1(xy) \geq \frac{\kappa'}{4|\mathcal{L}|M_l}$. Then based on w_1, we construct a weight function w_2 by considering another $x \in \mathcal{X}$ using the same method. Keep doing this until all the vertices in $\mathcal{X}$ are considered. Let the final weight function be w'' and $\kappa = \frac{\kappa'}{4|\mathcal{L}|M_l}$. Then w'' satisfies conditions (76)–(78), which completes the proof. □

D Proof of Lemma 6–Lemma 8

Proof of Lemma 6. According to the queue dynamics,

$$\begin{aligned}
&\|Q_{\|}(t+1)\|^2 - \|Q_{\|}(t)\|^2 \\
&= \langle c, A(t) - S(t) + U(t)\rangle^2 + 2\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle + 2\langle c, Q(t)\rangle\langle c, U(t)\rangle \\
&\geq 2\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle,
\end{aligned}$$

where the inequality follows from $\langle c, Q(t)\rangle\langle c, U(t)\rangle \geq 0$. □

Proof of Lemma 7. Using the fact that $\big(Q(t) - Q(t_0)\big)_{\perp} = Q_{\perp}(t) - Q_{\perp}(t_0)$ and the boundedness of arrivals and service yield

$$\big|\|Q_{\perp}(t)\| - \|Q_{\perp}(t_0)\|\big| \leq \|Q_{\perp}(t) - Q_{\perp}(t_0)\| \leq \|Q(t) - Q(t_0)\| \leq T\sqrt{M+1}\max\{M, NA_{\max}\}.$$

□

Proof of Lemma 8. First by Cauchy-Schwartz inequality,

$$G(t) = \langle Q_{\perp}(t), A_{\perp}(t) - S_{\perp}(t)\rangle \leq \|Q_{\perp}(t)\| \cdot \|A_{\perp}(t) - S_{\perp}(t)\| \leq \|Q_{\perp}(t)\| \cdot \|A(t) - S(t)\|.$$

By Lemma 7, $\big|\|Q_{\perp}(t)\| - \|Q_{\perp}(t_0)\|\big| \leq T\sqrt{M+1}\max\{M, NA_{\max}\}$. By the boundedness of the arrivals and service, $\|A(t) - S(t)\| \leq \sqrt{M+1}\max\{M, NA_{\max}\}$. Thus

$$\begin{aligned}
G(t) &\leq \big(\|Q_{\perp}(t_0)\| + T\sqrt{M+1}\max\{M, NA_{\max}\}\big) \cdot \sqrt{M+1}\max\{M, NA_{\max}\} \\
&= h\|Q_{\perp}(t_0)\| + F_1,
\end{aligned}$$

where $h = \sqrt{M+1}\max\{M, NA_{\max}\}$ and $F_1 = (M+1)T\left(\max\{M, NA_{\max}\}\right)^2$ are constants. □

E Proof of Lemma 9–Lemma 11

Proof of Lemma 9. First we derive a lower bound for $\langle Q(t), \tilde{\mu}(t)\rangle$. By definitions,

$$\langle Q(t), \tilde{\mu}(t)\rangle = \sum_{m=1}^{M} \big(Q_m(t)\tilde{\mu}_m^l(t) + \overline{Q}(t)\tilde{\mu}_m^r(t)\big).$$

For any $m \in \mathcal{M}_l$, consider the random variables $\tau_m^t = \max\big\{\tau \bigm| \tau \le t, f_m(\tau) = 0\big\}$, which is the last time slot before t at which machine m made a scheduling decision. Then $t_0 \le t_m^* \le \tau_m^t$ and $\sigma_m(t) = \sigma_m(\tau_m^t)$. Recall that the random variables $\tilde{\mu}_m^l(t)$ and $\tilde{\mu}_m^r(t)$ are defined as

$$\tilde{\mu}_m^l(t) = \begin{cases} \alpha\Big(1 - \frac{\epsilon}{\alpha(M_l+1)}\Big) & \text{if } \sigma_m(t) = 1, \\ 0 & \text{if } \sigma_m(t) = 2, \end{cases} \qquad \tilde{\mu}_m^r(t) = \begin{cases} 0 & \text{if } \sigma_m(t) = 1, \\ \gamma\big(1 - \frac{\epsilon}{\alpha(M_l+1)}\big) & \text{if } \sigma_m(t) = 2. \end{cases}$$

Thus they are functions of $\sigma_m(t)$ and thus $\tilde{\mu}_m^l(t) = \tilde{\mu}_m^l(\tau_m^t)$ and $\tilde{\mu}_m^r(t) = \tilde{\mu}_m^r(\tau_m^t)$. Then

$$\begin{aligned} Q_m(t)\tilde{\mu}_m^l(t) + \overline{Q}(t)\tilde{\mu}_m^r(t) &\overset{(a)}{\ge} Q_m(\tau_m^t)\tilde{\mu}_m^l(\tau_m^t) + \overline{Q}(\tau_m^t)\tilde{\mu}_m^r(\tau_m^t) - T(M+1) \\ &\overset{(b)}{\ge} Q_m(\tau_m^t)\alpha\left(1 - \frac{\epsilon}{\alpha(M_l+1)}\right) - T(M+1) \\ &\overset{(c)}{\ge} Q_m(t)\left(\alpha - \frac{\epsilon}{M_l+1}\right) - T(M+1) - TNA_{\max}, \end{aligned}$$

where (a) follows from the boundedness of service; (b) follows from the MaxWeight scheduling; (c) follows from the boundedness of arrivals. For any $m \in \mathcal{M}_r$, $Q_m(t) = 0$. Recall that $\tilde{\mu}_m^l(t) = 0$ and $\tilde{\mu}_m^r(t) = \gamma - \frac{\epsilon}{M_r(M_l+1)}$. Thus $Q_m(t)\tilde{\mu}_m^l(t) + \overline{Q}(t)\tilde{\mu}_m^r(t) = \overline{Q}(t)\Big(\gamma - \frac{\epsilon}{M_r(M_l+1)}\Big)$. Taking summation of these two parts yields

$$\langle Q(t), \tilde{\mu}(t)\rangle \ge \sum_{m\in\mathcal{M}_l} Q_m(t)\left(\alpha - \frac{\epsilon}{M_l+1}\right) + \overline{Q}(t)\left(M_r\gamma - \frac{\epsilon}{M_l+1}\right) - F_2',$$

where $F_2' = TM_l(M+1+NA_{\max}) > 0$. Consider the decomposition of λ in Lemma 4. Then

$$\sum_{\vec{L}\colon m\in\vec{L}} \lambda_{\vec{L},m} = \alpha - \frac{\epsilon}{M_l+1}, \ \ m \in \mathcal{M}_l, \quad \sum_{\vec{L}\in\mathcal{L}} \overline{\lambda}_{\vec{L}} = M_r\gamma - \frac{\epsilon}{M_l+1}.$$

Therefore,

$$\langle Q(t), \tilde{\mu}(t)\rangle \ge \sum_{m\in\mathcal{M}_l} Q_m(t) \sum_{\vec{L}\colon m\in\vec{L}} \lambda_{\vec{L},m} + \overline{Q}(t)\sum_{\vec{L}} \overline{\lambda}_{\vec{L}} - F_2'. \tag{86}$$

Next we derive an equivalent form of $\mathbb{E}\big[\langle Q(t), A(t)\rangle \bigm| t^*, Z(t_0)\big]$. Since the JSQ routing algorithm is used, the arrival $A_{\vec{L}}(t)$ joins the queue $Q_{\vec{L}}^*(t) = \min\big\{\min_{m\in\vec{L}} Q_m(t), \overline{Q}(t)\big\}$. Thus

$$\begin{aligned} \mathbb{E}\big[\langle Q(t), A(t)\rangle \bigm| t^*, Z(t_0)\big] &= \mathbb{E}\Big[\mathbb{E}\big[\langle Q(t), A(t)\rangle \bigm| Z(t)\big] \Big| t^*, Z(t_0)\Big] \\ &= \mathbb{E}\left[\mathbb{E}\Big[\textstyle\sum_{\vec{L}} Q_{\vec{L}}^*(t)A_{\vec{L}}(t) \Big| Z(t)\Big] \Big| t^*, Z(t_0)\right] \end{aligned}$$

$$= \mathbb{E}\Big[\textstyle\sum_{\vec{L}} Q^*_{\vec{L}}(t)\lambda_{\vec{L}} \;\Big|\; t^*, Z(t_0)\Big]$$
$$= \mathbb{E}\left[\sum_{\vec{L}}\left(\sum_{m\in\vec{L}} Q^*_{\vec{L}}(t)\lambda_{\vec{L},m} + Q^*_{\vec{L}}(t)\overline{\lambda}_{\vec{L}}\right) \;\middle|\; t^*, Z(t_0)\right].$$

Finally, we combine these two parts. Let $\lambda_{\min} = \frac{\kappa}{3N+1}$, where κ is defined in Lemma 4. Then $\lambda_{\min} > 0$ and $\lambda_{\min}$ does not depend on ϵ. For any $\vec{L} \in \mathcal{L}$ and any $m \in \mathcal{M}_l$, we have $\lambda_{\vec{L},m} \geq \lambda_{\min}$ and $\frac{\overline{\lambda}_{\vec{L}}}{3N+1} \geq \lambda_{\min}$. Meanwhile, by the JSQ routing algorithm, for any $m \in \vec{L}$, $Q^*_{\vec{L}}(t) - Q_m(t) \leq 0$, and $Q^*_{\vec{L}}(t) - \overline{Q}(t) \leq 0$. Therefore,

$$\begin{aligned}
&\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\mu}(t)\rangle \;\big|\; t^*, Z(t_0)\big] \\
&= \mathbb{E}\left[\sum_{\vec{L}}\left(\sum_{m\in\vec{L}} \big(Q^*_{\vec{L}}(t) - Q_m(t)\big)\lambda_{\vec{L},m} + (3N+1)\big(Q^*_{\vec{L}}(t) - \overline{Q}(t)\big)\frac{\overline{\lambda}_{\vec{L}}}{3N+1}\right) \;\middle|\; t^*, Z(t_0)\right] + F_2' \\
&\leq \mathbb{E}\left[-\lambda_{\min}\left(\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(Q_m(t) - Q^*_{\vec{L}}(t)\big) + (3N+1)\sum_{\vec{L}}\big(\overline{Q}(t) - Q^*_{\vec{L}}(t)\big)\right) \;\middle|\; t^*, Z(t_0)\right] + F_2'. \qquad (87)
\end{aligned}$$

Let $\vec{L}^* \in \arg\min_{\vec{L}} Q^*_{\vec{L}}(t)$ and $Q_{\min}(t) = Q^*_{\vec{L}^*}(t)$. Then discarding the terms with indices not equal to $\vec{L}^*$ in the summation $\sum_{\vec{L}}\big(\overline{Q}(t) - Q^*_{\vec{L}}(t)\big)$ yields

$$\begin{aligned}
&-\lambda_{\min}\left(\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(Q_m(t) - Q^*_{\vec{L}}(t)\big) + (3N+1)\sum_{\vec{L}}\big(\overline{Q}(t) - Q^*_{\vec{L}}(t)\big)\right) \\
&\leq -\lambda_{\min}\left(\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(Q_m(t) - Q^*_{\vec{L}}(t)\big) + (3N+1)\big(\overline{Q}(t) - Q^*_{\vec{L}^*}(t)\big)\right) \\
&= -\lambda_{\min}\left(\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(Q_m(t) - Q_{\min}(t) + Q_{\min}(t) - Q^*_{\vec{L}}(t)\big) + (3N+1)\Big(\overline{Q}(t) - Q_{\min}(t)\Big)\right) \\
&\overset{(a)}{=} -\lambda_{\min}\left(\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(Q_m(t) - Q_{\min}(t)\big) + \big(\overline{Q}(t) - Q_{\min}(t)\big)\right) - \lambda_{\min}\sum_{\vec{L}}\sum_{m\in\vec{L}}\big(\overline{Q}(t) - Q^*_{\vec{L}}(t)\big) \\
&\overset{(b)}{\leq} -\lambda_{\min}\left(\sum_{m\in\mathcal{M}_l}\big(Q_m(t) - Q_{\min}(t)\big) + \big(\overline{Q}(t) - Q_{\min}(t)\big)\right),
\end{aligned}$$

where (a) follows from combining each $\big(Q_{\min}(t) - Q^*_{\vec{L}}(t)\big)$ with $\big(\overline{Q}(t) - Q_{\min}(t)\big)$; (b) follows from discarding some terms in the summation. Using the notations of vectors yields

$$\begin{aligned}
\sum_{m\in\mathcal{M}_l}\big(Q_m(t) - Q_{\min}(t)\big) + \big(\overline{Q}(t) - Q_{\min}(t)\big) &= \|Q(t) - Q_{\min}(t)\sqrt{M_l+1}c\|_1 \\
&\geq \|Q(t) - Q_{\min}(t)\sqrt{M_l+1}c\|,
\end{aligned}$$

where $\|\cdot\|_1$ is the L^1 norm and the inequality follows from the fact that the L^1 norm of a vector is no smaller

than its L^2 norm. Since $\langle c, Q(t)\rangle$ minimizes the convex function $\|Q(t) - xc\|$ over $x \in \mathbb{R}$,

$$\|Q(t) - Q_{\min}(t)\sqrt{M_l+1}c\| \geq \|Q(t) - \langle c, Q(t)\rangle c\| = \|Q_\perp(t)\|.$$

Applying these bounds to (87) yields

$$\begin{aligned}\mathbb{E}\big[\langle Q(t), A(t)\rangle - \langle Q(t), \tilde{\mu}(t)\rangle \;\big|\; t^*, Z(t_0)\big] &\leq \mathbb{E}\big[-\lambda_{\min}\|Q_\perp(t)\| \;\big|\; t^*, Z(t_0)\big] + F_2' \\ &\leq -\lambda_{\min}\|Q_\perp(t_0)\| + F_2,\end{aligned}$$

where we have used Lemma 7, and $F_2 = F_2' + \lambda_{\min} T\sqrt{M+1}\max\{M, NA_{\max}\} > 0$. □

Proof of Lemma 10. The proof of this lemma is similar to the proof of Theorem 1 in [27], so it is omitted here due to space limit. □

Proof of Lemma 11. By the boundedness of arrivals and service,

$$\langle c, Q(t_0)\rangle - \frac{TM}{\sqrt{M_l+1}} \leq \langle c, Q(t)\rangle \leq \langle c, Q(t_0)\rangle + \frac{TNA_{\max}}{\sqrt{M_l+1}}.$$

Thus

$$\begin{aligned}&\mathbb{E}\left[\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle \mid t^*, Z(t_0)\right] \\ &\geq \langle c, Q(t_0)\rangle\mathbb{E}\left[\frac{1}{\sqrt{M_l+1}}\left(\sum_{\vec{L}} A_{\vec{L}}(t) - \sum_{m=1}^{M}\big(S_m^l(t) + S_m^r(t)\big)\right) \;\middle|\; t^*, Z(t_0)\right] - \frac{2TMNA_{\max}}{M_l+1}.\end{aligned}$$

Obviously $\langle c, Q(t_0)\rangle = \|Q_\parallel(t_0)\|$. By the assumption $\mathbb{E}\big[\sum_{\vec{L}} A_{\vec{L}}(t) \;\big|\; t^*, Z(t_0)\big] = M_l\alpha + M_r\gamma - \epsilon$. By the definitions $\mathbb{E}\big[\sum_{m=1}^{M}\big(S_m^l(t) + S_m^r(t)\big) \;\big|\; t^*, Z(t_0)\big] \leq M_l\alpha + M_r\gamma$. Therefore,

$$\mathbb{E}\big[\langle c, Q(t)\rangle\langle c, A(t) - S(t)\rangle \;\big|\; t^*, Z(t_0)\big] \geq -\frac{\epsilon}{\sqrt{M_l+1}}\|Q_\parallel(t_0)\| - F_4,$$

where $F_4 = \frac{2TMNA_{\max}}{M_l+1}$ is a positive constant. □

F Proof of Lemma 13

Proof. First notice that for any $m \in \mathcal{M}_r$ and any $t \geq 0$, $S_m^r(t) \sim \text{Bern}(\gamma)$ and $S_m^r(t)$ is independent from $Q(t)$. Then the two terms in the inequality can be written as

$$\begin{aligned}\mathbb{E}\big[\langle Q(t), S'(t) - S(t)\rangle\big] &= \sum_{m\in\mathcal{M}_l}\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big)\big], \\ \sqrt{M_l+1}\mathbb{E}\big[\langle c, S'(t) - S(t)\rangle\big] &= \sum_{m\in\mathcal{M}_l}\mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t)\big].\end{aligned}$$

Then it suffices to show that for any $m \in \mathcal{M}_l$,

$$\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big)\big] \leq R_1\mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t)\big].$$

To prove this inequality, for each $m \in \mathcal{M}_l$, we use the random variable $Z(t) = \big(Q(t), f(t)\big)$ to decompose the probability space as follows.

$$\begin{aligned}
&\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big)\big] \\
=& \sum_{Z:\ f_m=0} \mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big) \\
&+ \sum_{Z:\ f_m=1} \mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big) \\
&+ \sum_{Z:\ f_m=2} \mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big).
\end{aligned}$$

Denote the three summations above as LH_0, LH_1 and LH_2. Similarly,

$$\begin{aligned}
\mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t)\big] =& \sum_{Z:\ f_m=0} \mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big) \\
&+ \sum_{Z:\ f_m=1} \mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big) \\
&+ \sum_{Z:\ f_m=2} \mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t) \ \big|\ Z(t) = Z\big] \cdot \Pr\big(Z(t) = Z\big),
\end{aligned}$$

and denote these three summations as RH_0, RH_1 and RH_2. Next we are going to discuss the three cases $f_m = 0$, $f_m = 1$ and $f_m = 2$ case by case.

When $f_m = 0$, recall that this means machine m makes a scheduling decision at the current time slot t. Then by the MaxWeight scheduling algorithm

$$\begin{aligned}
&\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] \\
=& Q_m\alpha - \big(Q_m\,\mathbb{E}\big[S_m^l(t) \ \big|\ Z(t) = Z\big] + \overline{Q}\,\mathbb{E}\big[S_m^r(t) \ \big|\ Z(t) = Z\big]\big) \le 0.
\end{aligned}$$

Meanwhile, $\mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t) \ \big|\ Z(t) = Z\big] = \alpha - \mathbb{E}\big[S_m^l(t) + S_m^r(t) \ \big|\ Z(t) = Z\big] \ge 0$. Thus $LH_0 \le RH_0$.

When $f_m = 1$, recall that this means machine m is scheduled to the local queue at t, so $\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] = 0$ and $\mathbb{E}\big[X_m^l(t) - S_m^l(t) - S_m^r(t) \ \big|\ Z(t) = Z\big] = 0$. Therefore $LH_1 = RH_2$.

When $f_m = 2$, recall that this means machine m is scheduled to the common remote queue at t. Consider the random variable τ_m^t defined in the proof of Lemma 9, which is the last time slot before t at which machine m made a scheduling decision. Then

$$\begin{aligned}
&\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ Z(t) = Z\big] \\
=& \sum_{n=1}^{t} \Big\{ \mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \ \big|\ \tau_m^t = t - n, Z(t) = Z\big] \qquad (88) \\
&\qquad \cdot \Pr\big(\tau_m^t = t - n \ \big|\ Z(t) = Z\big) \Big\}. \qquad (89)
\end{aligned}$$

The conditional expectation in (88) can be bounded as

$$\begin{aligned}
&\mathbb{E}\big[Q_m(t)\big(X_m^l(t) - S_m^l(t)\big) + \overline{Q}(t)\big(-S_m^r(t)\big) \;\big|\; \tau_m^t = t-n, Z(t) = Z\big] \\
&= \mathbb{E}\big[Q_m(t)\alpha - \overline{Q}(t)\gamma \;\big|\; \tau_m^t = t-n, Z(t) = Z\big] \\
&\le \mathbb{E}\big[\big(Q_m(\tau_m^t)\alpha - \overline{Q}(\tau_m^t)\gamma\big) + \big(nNA_{\max}\alpha + nM\gamma\big) \;\big|\; \tau_m^t = t-n, Z(t) = Z\big] \qquad (90) \\
&\le n\big(NA_{\max}\alpha + M\gamma\big), \qquad (91)
\end{aligned}$$

where (90) follows from the boundedness of arrivals and service; (91) follows from the fact that machine m is scheduled to serve the common remote queue at time slot τ_m^t by the MaxWeight scheduling algorithm. The probability in (89) can be calculated as

$$\Pr\big(\tau_m^t = t-n \;\big|\; Z(t) = Z\big) = \frac{\Pr\left(\tau_m^t = t-n, Z(t) = Z\right)}{\Pr\left(Z(t) = Z\right)}.$$

The event $(\tau_m^t = t-n, Z(t) = Z)$ is equivalent to the event that at time slot $t-n$, machine m is available and is scheduled to the common remote queue, and for the time slots from $t-n$ to t, the working status of machine m all equal to 2. Thus

$$\begin{aligned}
&\Pr\big(\tau_m^t = t-n, Z(t) = Z\big) \\
&= \Pr\left(f_m(t-n) = 0, \sigma_m(t-n) = 2, f_m(t-n+1) = 2, \cdots, f_m(t-1) = 2, Z(t) = Z\right) \\
&\le \Pr\left(f_m(t-n+1) = 2, \cdots, f_m(t-1) = 2, Z(t) = Z \mid f_m(t-n) = 0, \sigma_m(t-n) = 2\right),
\end{aligned}$$

where $(f_m(t-n) = 0, \sigma_m(t-n) = 2)$ has positive probability since the Markov chain is irreducible. Let $\mathcal{A}_i = (f_m(t-i-1) = 2, \cdots, f(t-n+1) = 2, f_m(t-n) = 0, \sigma_m(t-n) = 2)$ for $i = 0, 1, \cdots, n-2$. Then using the chain rule of probability yields

$$\begin{aligned}
&\Pr\left(f_m(t-n+1) = 2, \cdots, f_m(t-1) = 2, Z(t) = Z \mid f_m(t-n) = 0, \sigma_m(t-n) = 2\right) \\
&= \Pr\left(f_m(t-n+1) = 2 \mid f_m(t-n) = 0, \sigma_m(t-n) = 2\right) \\
&\quad \cdot \prod_{i=1}^{n-2} \Pr\left(f_m(t-i) = 2 \mid \mathcal{A}_i\right) \cdot \Pr\left(Z(t) = Z \mid \mathcal{A}_0\right).
\end{aligned}$$

Given that machine m is scheduled to the common remote queue at time slot $t-n$, the working status $f_m(t-n+1)$ is determined by the random variable $S_m^r(t-n+1) \sim \text{Bern}(\gamma)$, so

$$\Pr\left(f_m(t-n+1) = 2 \mid f_m(t-n) = 0, \sigma_m(t-n) = 2\right) = 1-\gamma.$$

Similarly, for any $i = 1, 2, \cdots, n-2$, given $f_m(t-i-1) = 2$, the random variable $f_m(t-i)$ only depends on $S_m^r(t-i) \sim \text{Bern}(\gamma)$, so $\Pr\left(f_m(t-i) = 2 \mid \mathcal{A}_i\right) = 1-\gamma$. By these calculations,

$$\Pr\big(\tau_m^t = t-n \;\big|\; Z(t) = Z\big) \le (1-\gamma)^{n-1} \frac{\Pr\left(Z(t) = Z \mid \mathcal{A}_0\right)}{\Pr\left(Z(t) = Z\right)}. \qquad (92)$$

Utilizing the bound (91), (92) and the equality $\sum_{n=1}^{\infty} n(1-\gamma)^{n-1} = \frac{1}{\gamma^2}$ yields

$$\begin{aligned} LH_2 &= \sum_{Z\colon f_m=2} \mathbb{E}\left[Q_m(t)\left(X_m^l(t) - S_m^l(t)\right) + \overline{Q}(t)\left(-S_m^r(t)\right) \,\middle|\, Z(t) = Z\right] \cdot \Pr\left(Z(t) = Z\right) \\ &\le \left(NA_{\max}\alpha + M\gamma\right) \sum_{n=1}^{t} n(1-\gamma)^{n-1} \cdot \sum_{Z\colon f_m=2} \Pr\left(Z(t) = Z \mid \mathcal{A}_0\right) \\ &\le \frac{NA_{\max}\alpha + M\gamma}{\gamma^2}. \end{aligned}$$

Meanwhile, $\mathbb{E}\left[X_m^l(t) - S_m^l(t) - S_m^r(t) \,\middle|\, Z(t) = Z\right] = \alpha - \gamma$. Thus

$$\begin{aligned} RH_2 &= \sum_{Z\colon f_m=2} \mathbb{E}\left[X_m^l(t) - S_m^l(t) - S_m^r(t) \,\middle|\, Z(t) = Z\right] \cdot \Pr\left(Z(t) = Z\right) \\ &= (\alpha - \gamma) \Pr\left(f_m(t) = 2\right), \end{aligned}$$

where again $\Pr\left(f_m(t) = 2\right)$ is positive since the Markov chain is irreducible. Therefore,

$$LH_2 \le \frac{NA_{\max}\alpha + M\gamma}{\gamma^2(\alpha - \gamma)\Pr(f_m(t) = 2)} RH_2.$$

From the discussion of these three cases, we can conclude that for any $m \in \mathcal{M}_l$,

$$\mathbb{E}\left[Q_m(t)\left(X_m^l(t) - S_m^l(t)\right) + \overline{Q}(t)\left(-S_m^r(t)\right)\right] \le R_1 \mathbb{E}\left[X_m^l(t) - S_m^l(t)\right],$$

where

$$R_1 = \max\left\{1, \frac{NA_{\max}\alpha + M\gamma}{\gamma^2(\alpha - \gamma)\Pr(f_m(t) = 2)}\right\} > 0$$

is a constant, and further

$$\mathbb{E}\left[\langle Q(t), S'(t) - S(t)\rangle\right] \le R_1\sqrt{M_l + 1}\mathbb{E}\left[\langle c, S'(t) - S(t)\rangle\right].$$

□

G Proof of Lemma 14

Proof. We first prove that

$$\lim_{\epsilon \to 0^+} \mathbb{E}\left[\|S'(t) - S(t)\|^2\right] = 0$$

by contradiction. Suppose the equation above does not hold. Then there exists $a > 0$ such that for

$$\epsilon_0 = \min\left\{\frac{a(\alpha - \gamma)}{2\alpha}, \frac{a(\alpha - \gamma)}{2M_l\gamma}\right\}, \tag{93}$$

there exists ϵ with $0 < \epsilon \le \epsilon_0$ such that

$$\mathbb{E}\left[\|S'(t) - S(t)\|^2\right] > a.$$

Since

$$\mathbb{E}\big[\|S'(t)-S(t)\|^2\big]=\mathbb{E}\bigg[\sum_{m\in\mathcal{M}_l}\big(S'_m(t)-S_m(t)\big)^2\bigg]+\mathbb{E}\Big[\big(S'_{M+1}(t)-S_{M+1}(t)\big)^2\Big],$$

then either

$$\mathbb{E}\bigg[\sum_{m\in\mathcal{M}_l}\big(S'_m(t)-S_m(t)\big)^2\bigg]>\frac{a}{2}, \tag{94}$$

or

$$\mathbb{E}\Big[\big(S'_{M+1}(t)-S_{M+1}(t)\big)^2\Big]>\frac{a}{2}. \tag{95}$$

First consider the case (94). For any $m\in\mathcal{M}_l$, by the coupling of $S'(t)$ and $S(t)$, $0\le S'_m(t)-S_m(t)\le 1$. Therefore,

$$\mathbb{E}\bigg[\sum_{m\in\mathcal{M}_l}\big(S'_m(t)-S_m(t)\big)\bigg]\ge\mathbb{E}\bigg[\sum_{m\in\mathcal{M}_l}\big(S'_m(t)-S_m(t)\big)^2\bigg]>\frac{a}{2},$$

and thus

$$\mathbb{E}\bigg[M_l\alpha-\sum_{m\in\mathcal{M}_l}S_m(t)\bigg]>\frac{a}{2}.$$

Let $N_r(t)$ be the number of machines in $\mathcal{M}_l$ that are scheduled to serve remote tasks at time slot t. Then since

$$\mathbb{E}\bigg[M_l\alpha+M_r\gamma-\sum_{m=1}^{M+1}S_m(t)\;\bigg|\;Z(t)\bigg]=(\alpha-\gamma)N_r(t)$$
$$\mathbb{E}\bigg[M_l\alpha-\sum_{m\in\mathcal{M}_l}S_m(t)\;\bigg|\;Z(t)\bigg]=\alpha N_r(t),$$

we have

$$\mathbb{E}\bigg[M_l\alpha+M_r\gamma-\sum_{m=1}^{M+1}S_m(t)\bigg]=\frac{\alpha-\gamma}{\alpha}\mathbb{E}\bigg[M_l\alpha-\sum_{m\in\mathcal{M}_l}S_m(t)\bigg]>\frac{a(\alpha-\gamma)}{2\alpha}.$$

Therefore,

$$\mathbb{E}\bigg[\sum_{m=1}^{M+1}S_m(t)\bigg]<M_l\alpha+M_r\gamma-\frac{a(\alpha-\gamma)}{2\alpha}. \tag{96}$$

However, since the queueing process is in steady state, by the queue dynamics,

$$\begin{aligned}\mathbb{E}\bigg[\sum_{m=1}^{M+1}S_m(t)\bigg]&=\mathbb{E}\bigg[\sum_{m=1}^{M+1}A_m(t)\bigg]+\mathbb{E}\bigg[\sum_{m=1}^{M+1}U_m(t)\bigg]\\&\ge M_l\alpha+M_r\gamma-\epsilon\\&\overset{(a)}{\ge} M_l\alpha+M_r\gamma-\frac{a(\alpha-\gamma)}{2\alpha},\end{aligned}$$

where (a) follows from that $\epsilon\le\epsilon_0\le\frac{a(\alpha-\gamma)}{2\alpha}$. This contradicts with (96).

Next consider the case (95). By the coupling of $S'(t)$ and $S(t)$, $0 \le S_{M+1}(t) - S'_{M+1}(t) \le M_l$. Therefore

$$\mathbb{E}\big[S_{M+1}(t) - S'_{M+1}(t)\big] \ge \frac{1}{M_l}\mathbb{E}\Big[\big(S'_{M+1}(t) - S_{M+1}(t)\big)^2\Big] > \frac{a}{2M_l},$$

and thus

$$\mathbb{E}\big[S_{M+1}(t) - M_l\gamma\big] > \frac{a}{2M_l}.$$

Let $N_r(t)$ be the number of machines in $\mathcal{M}_l$ that are scheduled to serve remote tasks at time slot t. Then since

$$\mathbb{E}\big[S_{M+1}(t) - M_l\gamma \;\big|\; Z(t)\big] = \gamma N_r(t),$$

we have

$$\mathbb{E}\left[M_l\alpha + M_r\gamma - \sum_{m=1}^{M+1} S_m(t)\right] = \frac{\alpha-\gamma}{\gamma}\mathbb{E}\big[S_{M+1}(t) - M_l\gamma\big] > \frac{a(\alpha-\gamma)}{2M_l\gamma}.$$

Therefore,

$$\mathbb{E}\left[\sum_{m=1}^{M+1} S_m(t)\right] < M_l\alpha + M_r\gamma - \frac{a(\alpha-\gamma)}{2M_l\gamma}.$$

By similar arguments as in the case (94) we get a contradiction.

Now we have proved that

$$\lim_{\epsilon\to 0^+} \mathbb{E}\big[\|S'(t) - S(t)\|^2\big] = 0.$$

Since $\|U'(t)\| \le \sqrt{M_l + M^2}$,

$$\begin{aligned}\mathbb{E}\big[\|U'(t)\|^2\big] &\le \sqrt{M_l + M^2}\,\mathbb{E}\big[\|U'(t)\|\big] \\ &\le \sqrt{M_l + M^2}\Big(\mathbb{E}\big[\|U'(t) - U(t)\|\big] + \mathbb{E}\big[\|U(t)\|\big]\Big) \end{aligned} \tag{97}$$

By the definition $U'(t) = S'(t) - S(t) + U(t)$,

$$\begin{aligned}\mathbb{E}\big[\|U'(t) - U(t)\|\big] &= \mathbb{E}\big[\|S'(t) - S(t)\|\big] \\ &\le \sqrt{\mathbb{E}\big[\|S'(t) - S(t)\|^2\big]}. \end{aligned} \tag{98}$$

Consider the second term $\mathbb{E}\big[\|U(t)\|\big]$. Since $\langle c, S'(t) - S(t)\rangle \ge 0$,

$$\mathbb{E}\big[\langle c, U(t)\rangle\big] \le \mathbb{E}\big[\langle c, U'(t)\rangle\big] = \frac{\epsilon}{\sqrt{M_l+1}},$$

and thus

$$\mathbb{E}\big[\langle c, U(t)\rangle^2\big] \le \frac{M}{\sqrt{M_l+1}}\mathbb{E}\big[\langle c, U(t)\rangle\big] \le \frac{\epsilon M}{M_l+1}.$$

Then

$$\begin{aligned}\mathbb{E}\big[\|U(t)\|\big] &\le \sqrt{\mathbb{E}\big[\|U(t)\|^2\big]} \\ &\le \sqrt{\mathbb{E}\big[(M_l+1)\langle c, U(t)\rangle^2\big]} \\ &\le \sqrt{\epsilon M}. \end{aligned} \tag{99}$$

By the bounds (97), (98), (99),

$$\lim_{\epsilon \to 0^+} \mathbb{E}\big[\|U'(t)\|^2\big] = 0,$$

which completes the proof. □

References

[1] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *ACM Commun.*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[2] "Hadoop," http://hadoop.apache.org.

[3] G. Ananthanarayanan, S. Agarwal, S. Kandula, A. Greenberg, I. Stoica, D. Harlan, and E. Harris, "Scarlett: coping with skewed content popularity in mapreduce clusters," in *Proc. European Conf. Computer Systems (EuroSys)*, Salzburg, Austria, 2011, pp. 287–300.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. ACM Symp. Operating Systems Principles (SOSP)*, Bolton Landing, NY, 2003, pp. 29–43.

[5] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop distributed file system," in *IEEE Symp. Mass Storage Systems and Technologies (MSST)*, Incline Villiage, NV, May 2010, pp. 1–10.

[6] T. White, *Hadoop: The definitive guide*. Yahoo Press, 2010.

[7] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling," in *Proc. European Conf. Computer Systems (EuroSys)*, Paris, France, 2010, pp. 265–278.

[8] J. Polo, C. Castillo, D. Carrera, Y. Becerra, I. Whalley, M. Steinder, J. Torres, and E. Ayguadé, "Resource-aware adaptive scheduling for mapreduce clusters," in *Proc. ACM/IFIP/USENIX Int. Conf. Middleware*, Lisbon, Portugal, 2011, pp. 187–207.

[9] J. Jin, J. Luo, A. Song, F. Dong, and R. Xiong, "Bar: An efficient data locality driven task scheduling algorithm for cloud computing," in *Proc. IEEE/ACM Int. Conf. Cluster, Cloud and Grid Computing (CCGRID)*, Newport Beach, CA, 2011, pp. 295–304.

[10] Q. Xie and Y. Lu, "Degree-guided map-reduce task assignment with data locality constraint," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Cambridge, MA, 2012, pp. 985–989.

[11] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, "Quincy: fair scheduling for distributed computing clusters," in *Proc. ACM Symp. Operating Systems Principles (SOSP)*, Big Sky, MT, 2009, pp. 261–276.

[12] S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan, "An analysis of traces from a production mapreduce cluster," in *Proc. IEEE/ACM Int. Conf. Cluster, Cloud and Grid Computing (CCGRID)*, Melbourne, Australia, 2010, pp. 94–103.

[13] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz, "Energy efficiency for large-scale mapreduce workloads with significant interactive analysis," in *Proc. European Conf. Computer Systems (EuroSys)*, Bern, Switzerland, 2012, pp. 43–56.

[14] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in *Proc. USENIX Conf. Operating Systems Design and Implementation (OSDI)*, San Diego, CA, 2008, pp. 29–42.

[15] F. Chen, M. Kodialam, and T. V. Lakshman, "Joint scheduling of processing and shuffle phases in MapReduce systems," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Orlando, FL, Mar. 2012, pp. 1143–1151.

[16] J. Tan, X. Meng, and L. Zhang, "Coupling task progress for mapreduce resource-aware scheduling," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Turin, Italy, Apr. 2013, pp. 1618–1626.

[17] M. Lin, L. Zhang, A. Wierman, and J. Tan, "Joint optimization of overlapping phases in MapReduce," in *Proc. Int. Symp. Computer Performance, Modeling, Measurements and Evaluation (IFIP Performance)*, vol. 70, no. 10, Vienna, Austria, 2013, pp. 720–735.

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

[19] S. T. Maguluri and R. Srikant, "Scheduling jobs with unknown duration in clouds," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Turin, Italy, 2013.

[20] A. Eryilmaz and R. Srikant, "Asymptotically tight steady-state queue length bounds implied by drift conditions," *Queueing Syst.*, vol. 72, no. 3-4, pp. 311–359, Dec. 2012.

[21] S. T. Maguluri, R. Srikant, and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," in *Int. Teletraffic Congr. (ITC)*, Krakow, Poland, 2012.

[22] W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Map task scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality," Arizona State Univ., Tempe, AZ, Tech. Rep., Aug. 2013. [Online]. Available: http://inlab.lab.asu.edu/Publications/MapReduce13Tech.pdf

[23] J. F. C. Kingman, "Some inequalities for the queue GI/G/1," *Biometrika*, vol. 49, no. 3-4, pp. 315–324, Dec. 1962.

[24] B. Hajek, "Hitting-time and occupation-time bounds implied by drift analysis with applications," *Ann. Appl. Prob.*, pp. 502–525, 1982.

[25] G. Ananthanarayanan, A. Ghodsi, A. Wang, D. Borthakur, S. Kandula, S. Shenker, and I. Stoica, "Pacman: coordinated memory caching for parallel jobs," in *USENIX Symp. Networked Systems Design and Implementations (NSDI)*, 2012, pp. 20–20.

[26] R. G. Gallager, *Discrete Stochastic Processes*, ser. The Springer International Series in Engineering and Computer Science. New York, NY: Springer US, 1996.

[27] W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "A throughput optimal algorithm for map task scheduling in mapreduce with data locality," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 33–42, Mar. 2013.