Resource Allocation for Data-Parallel Computing in Networks with Data Locality

Weina Wang and Lei Ying

School of Electrical, Computer and Energy Engineering
Arizona State University, Tempe, AZ 85281
{weina.wang, lei.ying.2}@asu.edu

***Abstract*—This paper studies resource allocation for data-parallel applications in a networked computing system with data locality. Data-parallel computing tasks have two components: data and computation. To support efficient data-processing in the system, the resource allocation algorithm should jointly consider load-balancing, data transmissions and processing scheduling. In this paper, we consider a general model of a computing system where the computing system is a network represented by a graph, with nodes being computing devices or switches and edges being communication links. The data chunks stored at the nodes can be processed locally or be transmitted to other computing nodes in the network to be processed. The throughput of such a system for processing data-parallel applications is determined jointly by the computing capacity of the nodes and the communication capacity of the network, and the resource allocation algorithm should strike a right balance between computing and communication. In this paper, we propose a throughput-optimal resource allocation algorithm, which is also able to control the tradeoff between the expected amount of data transmitted and the expected number of backlogged tasks in steady state through a parameter $q_{\rm th}$. We show that the gap between the expected data transmission rate under our algorithm and the optimal value is $O(1/q_{\rm th})$, while the expected number of total backlogged tasks is upper bounded by $O(q_{\rm th})$.**

I. Introduction

Data-parallel computing frameworks, such as MapReduce/Hadoop and Dryad, provide essential infrastructure for big data analytics. Executing a data-processing task requires not only computation resource but also the availability of its corresponding input data, which is usually stored in the same computing system. The coupling between computation and data locality makes the resource allocation problem for data-parallel computing tasks intrinsically rich, and poses challenges to the design of high performance resource allocation algorithms, which has become an active research area (see, e.g., [1]–[9]).

When scheduling data-processing tasks, the system needs to balance the computation loads to maximize the computing throughput. Since each task has associated input data, if we place a task on a device that does not store the data initially, the device needs to obtain a copy of the data through the communication network, which incurs data transmissions in the network. Therefore, to make the computing system efficient, load-balancing, data transmissions and processing scheduling should be considered jointly.

Data locality used to be addressed by a crude model [1]–[6], [9]: data is classified coarsely as data in the same Java Virtual Machine (in Spark [3]), data on the same node, data on a different server on the same rack, and data not in the same rack. A more sophisticated approach is adopted in [8], which explicitly considers the topology and bandwidth of a communication network by assuming a tree topology.

In this paper, we consider a computing system where computing devices are connected by a communication network with a general topology. Specifically, we model the network by a graph, where the nodes are the computing devices and switches and the edges are the communication links between them. The computing nodes also store the data chunks to be processed by computing tasks. When a computing task is submitted to the system, it is first routed to one of the nodes that store its input data. Then the computing task is processed either at the node or is transmitted (together with the data chunk) to a different computing node for processing. Therefore, the resource allocation decisions in this system include where to route an incoming task, whether a node processes a task by itself or transmits the task to its neighbors.

In such a system, the computing throughput is determined by both the computing capacity of the devices and the communication capacity of the network. Our goal is to develop algorithms under which the system is stable for an as large as possible range of incoming traffic intensity, i.e., algorithms that achieve optimal throughput. On top of this, we also want the amount of data transmitted in the network to be minimized in order to minimize the usage of network bandwidth.

By looking at the resource allocation problem that minimizes data transmission subject to stability constraints, we propose a joint load balancing, data transmission and computing scheduling algorithm that is throughput optimal, i.e., it stabilizes the system for any incoming traffic as long as the traffic is supportable under some algorithm. Further, our proposed algorithm has a tuning parameter $q_{\rm th}$ that controls the tradeoff between the amount of transmitted data and the total number of backlogged tasks in steady state. We show

that in the steady state, the gap between the expected amount of transmitted data in each time slot under our algorithm and the optimal value is $O(1/q_{\rm th})$, and the expected number of total backlogged tasks is upper bounded by $O(q_{\rm th})$. Therefore, increasing this parameter makes the amount of transmitted data closer to the optimal value, but results in a possibly larger number of backlogged tasks.

II. Related Work

The problem of scheduling data-processing tasks has attracted much attention since computing systems for large-scale data-processing such as MapReduce/Hadoop [1] were proposed. Besides the computation resources, data locality and the resultant usage of network resources due to data transmission also need to be taken into account. Data locality used to be addressed by a crude model: data is classified coarsely as data in the same JVM (in Spark [3]), data on the same node, data on a different server on the same rack, and data not in the same rack. Based on this model, both heuristic scheduling algorithms (e.g., [1]–[3]) and algorithms with theoretic performance guarantees (e.g., [5], [6], [9]) have been developed. The work [8] explicitly models a communication network with a tree topology, and a throughput optimal scheduling/routing algorithm is developed. The present paper considers a general network topology, which includes the system in [5], [6], [8], [9] as special cases, and further considers heterogeneous computing devices, data types and computing tasks. There is a connection between our approach and the packet scheduling problem that minimizes the path lengths between sources and destinations subject to stability constraints [10], [11], which improves the delay performance compared with the traditional back-pressure algorithm first proposed in the seminal work [12].

III. System Model

We consider a computing system represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The node set $\mathcal{N}$ is the set of devices, which can be computing devices or switches, and the edge set $\mathcal{E}$ is the set of communication links. The bandwidth of link (m, n), i.e., the link from node m to node n, is denoted by $C_{(m,n)}$. Data chunks are stored on the computing nodes. Let $\mathcal{D}$ denote the set of data chunks stored in the system, and $\mathcal{C}$ denote the set of computing operations (e.g., sorting, searching, encryption/decryption). We classify the data into *classes* such that data chunks of the same class have the same size and require the same processing time for the same computing operation. Let $\mathcal{K}$ denote the set of classes. We call a computing task a (α, c)-task if it processes a class-α data chunk with operation c. Let α_d denote the class to which data chunk d belongs, l_α the size of a data chunk in class α, and $P_{\alpha,c,n}$ be node n's computing capacity of processing (α, c)-tasks, e.g., the number of class-α data chunks node n can serve with operation c in one time slot. Note that if node n is a switch, we let $P_{\alpha,c,n} = 0$ and follow the convention $0/0 = 0$ since a switch does not have computing capacity. We also note that a switch node should be able to reach at least one computing node in the graph, which holds in a practical system. Furthermore, let $\mathcal{S}_d$ denote the set of nodes that store data chunk d, and $\mathcal{D}_n$ the set of data chunks stored at node n. The notation introduced above and used throughout the paper is summarized in Table I.

$\mathcal{N}$	set of devices
$\mathcal{E}$	set of communication links
$C_{(m,n)}$	bandwidth of link (m, n)
$\mathcal{D}$	set of data stored in the network
$\mathcal{K}$	set of data classes
$\mathcal{C}$	set of computing operations
α_d	class to which data chunk d belongs
l_α	size of a data chunk in class α
$P_{\alpha,c,n}$	node n's computing capacity of processing a class-α data chunk with operation c
$\mathcal{S}_d$	set of nodes that store data chunk d
$\mathcal{D}_n$	set of data chunks stored at node n
$A_{d,c}[t]$	number of computing tasks to process data d with operation c
$\lambda_{d,c}$	$\lambda_{d,c} = E[A_{d,c}[t]]$

TABLE I: Notation Table

In the computing system considered in this paper, when a computing task c for data chunk d is requested, the request (also called a task) is first routed to one of the nodes who stores the data. Then, the computing task is processed either at the node or is transmitted (together with the data chunk) to a different node for processing.

Now let $A_{d,c}[t]$ denote the number of computing tasks to process data d with operation c at time slot t, where we assume that the random process $(A_{d,c}[t]\colon t \geq 0)$ is independent and identically distributed (i.i.d.) across time slots. We further assume that $A_{d,c}[t]$ has bounded mean and variance and has positive probability to be zero, and define $\lambda_{d,c} = \mathbb{E}[A_{d,c}[t]]$. Let $(\lambda_{d,c}) = (\lambda_{d,c}, d \in \mathcal{D}, c \in \mathcal{C})$. We say a computing traffic $(\lambda_{d,c})$ can be supported in the system if there exists some algorithm such that the number of unfinished tasks at time slot t, denoted by $\Phi[t]$, satisfies

$$\lim_{C\to\infty} \lim_{t\to\infty} \Pr(\Phi[t] > C) = 0.$$

The capacity region of the computing system consists of supportable $(\lambda_{d,c})$'s. Our first result characterizes the capacity region.

Theorem 1. *If computing traffic $(\lambda_{d,c})$ can be supported in the system, then there exist $\mathcal{S}_d$, $\lambda_{d,c,n} \geq 0$, $\mu_{\alpha,c,(m,n)} \geq 0$ and $\gamma_{\alpha,c,n} \geq 0$ such that the following inequalities hold:*

$$\sum_{d:\alpha_d=\alpha} \lambda_{d,c,n} + \sum_{m:(m,n)\in\mathcal{E}} \mu_{\alpha,c,(m,n)}$$

$$\leq \sum_{k:(n,k)\in\mathcal{E}} \mu_{\alpha,c,(n,k)} + \gamma_{\alpha,c,n}, \quad \forall \alpha, c, n, \tag{1}$$

$$\sum_{\alpha}\sum_{c} \frac{\gamma_{\alpha,c,n}}{P_{\alpha,c,n}} \leq 1, \quad \forall n, \tag{2}$$

$$\sum_{\alpha}\sum_{c} l_\alpha \mu_{\alpha,c,(m,n)} \leq C_{(m,n)}, \quad \forall m, n, \tag{3}$$

$$\sum_{n\in\mathcal{S}_d} \lambda_{d,c,n} = \lambda_{d,c}, \quad \forall d, c. \tag{4}$$

More specifically, define Λ to be the capacity region of the computing system. Then for any $(\lambda_{d,c}) \in \Lambda$, there exist a set of $\mathcal{S}_d$, $\lambda_{d,c,n}$, $\mu_{\alpha,c,(m,n)}$, $\gamma_{\alpha,c,n}$ such that (1)–(4) hold. To facilitate understanding, we can regard $\lambda_{d,c,n}$ as the average rate of tasks that process data chunk d with operating c coming to node n, $\mu_{\alpha,c,(m,n)}$ as the average transmission rate of (α, c)-tasks from node m to node n, and $\gamma_{\alpha,c,n}$ as the average processing rate of (α, c)-tasks on node n. Then (4) can be explained as a decomposition of the incoming traffic to the nodes that store the corresponding data. Inequality (1) states that the total incoming rate of (α, c)-tasks to node n should be less than or equal to the total outgoing rate. The inequalities (2) and (3) are processing capacity and data transmission capacity constraints, respectively.

We assume that $\mathcal{S}_d$ is pre-determined, i.e., the distributed storage algorithm is independent of data-processing. This is common in nowadays big-computing systems such as MapReduce/Hadoop, where by default each data chunk is replicated three times and distributed to three different machines. The locations of the replicas do not change. In such a setting, we consider the following optimization problem:

$$\min \sum_{(m,n)\in\mathcal{E}} \sum_{\alpha}\sum_{c} l_\alpha \mu_{\alpha,c,(m,n)} \tag{5}$$

$$\text{s.t.} \sum_{d:\alpha_d=\alpha} \lambda_{d,c,n} + \sum_{m:(m,n)\in\mathcal{E}} \mu_{\alpha,c,(m,n)} \leq \sum_{k:(n,k)\in\mathcal{E}} \mu_{\alpha,c,(n,k)} + \gamma_{\alpha,c,n}, \quad \forall \alpha, c, n, \tag{6}$$

$$\sum_{\alpha}\sum_{c} \frac{\gamma_{\alpha,c,n}}{P_{\alpha,c,n}} \leq 1, \quad \forall n, \tag{7}$$

$$\sum_{\alpha}\sum_{c} l_\alpha \mu_{\alpha,c,(m,n)} \leq C_{(m,n)}, \quad \forall m, n, \tag{8}$$

$$\sum_{n\in\mathcal{S}_d} \lambda_{d,c,n} = \lambda_{d,c}, \quad \forall d, c, \tag{9}$$

where the optimization variables are $\lambda = (\lambda_{d,c,n}, d \in \mathcal{D}, c \in \mathcal{C}, n \in \mathcal{N})$, $\mu = (\mu_{\alpha,c,(m,n)}, \alpha \in \mathcal{K}, (m,n) \in \mathcal{E})$ and $\gamma = (\gamma_{\alpha,c,n}, \alpha \in \mathcal{K}, c \in \mathcal{C}, n \in \mathcal{N})$. Let $(\lambda^*, \mu^*, \gamma^*)$ denote an optimal solution of this problem. We remark that the objective function (5) is to minimize the data transmitted in the network while guaranteeing all computing tasks are fulfilled. The reason for adding this objective is that there are other applications on the same network that need communication bandwidth. So data transmission should be avoided unless the local node is heavily congested.

To develop a protocol to solve the problem above, we consider the following partial Lagrangian dual subject to constraint (6), where $q_{\alpha,c,n}$ is the Lagrangian multiplier:

$$\begin{aligned}
&\min_{\lambda,\mu,\gamma} \sum_{(m,n)\in\mathcal{E}} \sum_{\alpha}\sum_{c} l_\alpha \mu_{\alpha,c,(m,n)} \\
&\quad + \sum_{\alpha,c,n} q_{\alpha,c,n} \Bigg(\sum_{d:\alpha_d=\alpha} \lambda_{d,c,n} + \sum_{m:(m,n)\in\mathcal{E}} \mu_{\alpha,c,(m,n)} \\
&\qquad\qquad - \sum_{k:(n,k)\in\mathcal{E}} \mu_{\alpha,c,(n,k)} - \gamma_{\alpha,c,n} \Bigg) \\
&= \min_{\lambda} \sum_{\alpha,c,n} q_{\alpha,c,n} \sum_{d:\alpha_d=\alpha} \lambda_{d,c,n} \\
&\quad - \max_{\mu} \sum_{\alpha,c} \sum_{(n,k)\in\mathcal{E}} \mu_{\alpha,c,(n,k)} \left(q_{\alpha,c,n} - q_{\alpha,c,k} - l_\alpha\right) \\
&\quad - \max_{\gamma} \sum_{\alpha,c,n} q_{\alpha,c,n} \gamma_{\alpha,c,n}.
\end{aligned}$$

The equation above leads to the following algorithm including load-balancing, data transmissions and computing scheduling.

Joint load-balancing, data transmissions and computing scheduling: At each node n, let $Q_{\alpha,c,n}[t]$ denote the number of buffered computing tasks that process a class-α data chunk with operation c at the beginning of time slot t. Let $Q[t] = (Q_{\alpha,c,n}[t], \alpha \in \mathcal{K}, c \in \mathcal{C}, n \in \mathcal{N})$. When a computing task c for data d arrives, it is routed to a node n that achieves

$$\min_{n\in\mathcal{S}_d} Q_{\alpha,c,n}[t], \tag{10}$$

i.e., joining the shortest queue. At each link (n, k), consider the (α, c) that achieves

$$\max_{\alpha,c} \frac{1}{l_\alpha} \left(Q_{\alpha,c,n}[t] - Q_{\alpha,c,k}[t]\right) - q_{\text{th}}, \tag{11}$$

where q_{th} is a positive parameter. Then node n transmits computing task in queue $Q_{\alpha,c,n}$ together with the associated data to node k when $(Q_{\alpha,c,n}[t] - Q_{\alpha,c,k}[t])/l_\alpha - q_{\text{th}} > 0$. In the performance analysis to be presented in the next section, we will see that the parameter q_{th} controls the tradeoff between the amount of transmitted data and the total queue length. Furthermore, node n selects tasks in $Q_{\alpha,c,n}$ to process such that (α, c) achieves

$$\max_{\alpha,c} P_{\alpha,c,n} Q_{\alpha,c,n}[t]. \tag{12}$$

□

Note that under this algorithm, the queueing process $(Q[t] : t \geq 0)$ is a Markov chain. Let its state space be the set of states reachable from the initial state.

IV. Performance Analysis

In this section, we analyze the performance of the proposed algorithm. Specifically, we focus on its throughput performance and the tradeoff between the amount of transmitted data and the total queue length, which is closely related to the delay performance.

We first derive the dynamics of the queueing system. At the beginning of each time slot t, each node n checks the lengths of the queues on it, i.e., obtains $(Q_{\alpha,c,n}[t], \alpha \in \mathcal{K}, c \in \mathcal{C})$. Based on the queue-length information, incoming computing tasks are routed to nodes according to (10). Let $A_{d,c,n}[t]$ denote the number of incoming computing tasks to process data d with operation c that are routed to node n. Then each node makes decisions on what tasks (together with the associated data) to transmits and what tasks to process according to (11) and (12), respectively. The transmitted tasks will be received at the end of the time slot. Let $S_{\alpha,c,n}[t]$ denote the number of tasks processed from queue $Q_{\alpha,c,n}$ at time slot t, and $S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]$ denote the number of tasks transmitted from queue $Q_{\alpha,c,n}$ to queue $Q_{\alpha,c,k}$ at time slot t. Then the queues evolve according to the following equation:

$$\begin{aligned} Q_{\alpha,c,n}[t+1] = Q_{\alpha,c,n}[t] &+ \sum_{d:\alpha_d=\alpha} A_{d,c,n}[t] - S_{\alpha,c,n}[t] \\ &- \sum_{k:(n,k)\in\mathcal{E}} S^{\mathrm{I}}_{\alpha,c,(n,k)}[t] \\ &+ \sum_{m:(m,n)\in\mathcal{E}} S^{\mathrm{I}}_{\alpha,c,(m,n)}[t]. \end{aligned} \tag{13}$$

Theorem 2. *For any traffic $(\lambda_{d,c}) \in \mathrm{int}(\Lambda)$, where $\mathrm{int}(\Lambda)$ is the interior of Λ, the Markov chain $(Q[t]\colon t \geq 0)$ is positive recurrent. Further,*

$$\lim_{t\to\infty} \mathbb{E}\left[\sum_{\alpha,c,n} Q_{\alpha,c,n}[t]\right] \leq B_1 q_{\mathrm{th}} + B_2 \tag{14}$$

and

$$\begin{aligned} &\lim_{t\to\infty} \mathbb{E}\left[\sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}} l_\alpha S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]\right] \\ &\leq \sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}} l_\alpha \mu^*_{\alpha,c,(n,k)} + \frac{B_3}{q_{\mathrm{th}}}, \end{aligned} \tag{15}$$

for some positive B_1, B_2 and B_3, where μ^* together with λ^* and γ^* is an optimal solution of (5) *subject to constraints* (6)–(9).

It can be verified that the Markov chain $(Q[t]\colon t \geq 0)$ is irreducible and aperiodic. Then the positive recurrence implies that the traffic $(\lambda_{d,c})$ can be supported in the system by the proposed algorithm since the distribution of the Markov chain converges to the unique stationary distribution. Since the algorithm supports any traffic strictly within the capacity region, we say that the algorithm is throughput optimal.

The bounds in (14) and (15) show that the parameter q_{th} controls the tradeoff between the amount of transmitted data and the total queue length. Specifically, if we use smaller q_{th}, then the total queue length becomes smaller but the gap between the amount of transmitted data in the system and the optimal value becomes larger. This is consistent with our intuition: smaller q_{th} means more data transmission but tasks have a better chance finding a less loaded node.

Proof. We will prove Theorem 2 for the case that all the nodes are computing devices to highlight the main idea. But note that the theorem holds for the general case.

We first prove the positive recurrence and the bound (14). Consider any traffic $(\lambda_{d,c}) \in \mathrm{int}(\Lambda)$. Then there exists an $\epsilon > 0$ such that $(1+\epsilon)(\lambda_{d,c}) \in \Lambda$. We use the Foster-Lyapunov theorem [13] to prove positive recurrence. Consider the Lyapunov function $V\colon \mathbb{Z}_+^{|\mathcal{K}|\cdot|\mathcal{C}|\cdot|\mathcal{N}|} \to \mathbb{R}_+$ defined by

$$V(Q) = \sum_{\alpha,c,n} Q^2_{\alpha,c,n}. \tag{16}$$

Then the drift can be written as

$$\begin{aligned} &\mathbb{E}[V(Q[t+1]) - V(Q[t]) \mid Q[t]] \\ &= 2\mathbb{E}\Bigg[\sum_{\alpha,c,n}\sum_{d:\alpha_d=\alpha} Q_{\alpha,c,n}[t] A_{d,c,n}[t] \\ &\quad - \sum_{\alpha,c,n} Q_{\alpha,c,n}[t] S_{\alpha,c,n}[t] \\ &\quad - \sum_{\alpha,c,n} Q_{\alpha,c,n}[t] \sum_{k:(n,k)\in\mathcal{E}} S^{\mathrm{I}}_{\alpha,c,(n,k)}[t] \\ &\quad + \sum_{\alpha,c,n} Q_{\alpha,c,n}[t] \sum_{m:(m,n)\in\mathcal{E}} S^{\mathrm{I}}_{\alpha,c,(m,n)}[t] \;\Bigg|\; Q[t]\Bigg] \\ &\quad + C_1, \end{aligned}$$

where C_1 is a constant independent of $Q[t]$. Next we analyze the summands on the right hand side.

For the summand corresponding to incoming tasks, there holds

$$\begin{aligned} &\mathbb{E}\left[\sum_{\alpha,c,n}\sum_{d:\alpha_d=\alpha} Q_{\alpha,c,n}[t] A_{d,c,n}[t] \;\middle|\; Q[t]\right] \\ &= \mathbb{E}\left[\sum_{c,d}\sum_{n:n\in\mathcal{S}_d} Q_{\alpha_d,c,n}[t] A_{d,c,n}[t] \;\middle|\; Q[t]\right] \\ &= \mathbb{E}\left[\sum_{c,d} A_{d,c}[t] \min_{n:n\in\mathcal{S}_d} Q_{\alpha_d,c,n}[t] \;\middle|\; Q[t]\right] \\ &= \sum_{c,d} \lambda_{d,c} \min_{n:n\in\mathcal{S}_d} Q_{\alpha_d,c,n}[t], \end{aligned} \tag{17}$$

where (17) follows from the join-the-shortest-queue policy. Since $(\lambda_{d,c}) \in \mathrm{int}(\Lambda)$, by (1)–(4), there exist nonnegative

$\lambda_{d,c,n}$'s such that

$$\sum_{n\in\mathcal{S}_d}\lambda_{d,c,n}=\lambda_{d,c}. \tag{18}$$

Therefore,

$$\begin{aligned}&\mathbb{E}\left[\sum_{\alpha,c,n}\sum_{d:\alpha_d=\alpha}Q_{\alpha,c,n}[t]A_{d,c,n}[t]\;\middle|\;Q[t]\right]\\ \le&\sum_{c,d}\sum_{n:n\in\mathcal{S}_d}\lambda_{d,c,n}Q_{\alpha_d,c,n}[t]\\ =&\sum_{\alpha,c,n}\sum_{d:\alpha_d=\alpha}\lambda_{d,c,n}Q_{\alpha,c,n}[t].\end{aligned} \tag{19}$$

For the summand corresponding to task processing, there holds

$$\begin{aligned}&\mathbb{E}\left[-\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]S_{\alpha,c,n}[t]\;\middle|\;Q[t]\right]\\ =&\mathbb{E}\left[-\sum_{n}Q_{\alpha_n^*,c_n^*,n}[t]S_{\alpha_n^*,c_n^*,n}[t]\;\middle|\;Q[t]\right],\end{aligned} \tag{20}$$

where

$$(\alpha_n^*,c_n^*)\in\arg\max_{(\alpha,c)}P_{\alpha,c,n}Q_{\alpha,c,n}[t],$$

and the equation follows from the processing scheduling policy in (12). By the boundedness of the processing and transmission capacity, we have

$$\begin{aligned}&-Q_{\alpha_n^*,c_n^*,n}[t]S_{\alpha_n^*,c_n^*,n}[t]\\ \le&-Q_{\alpha_n^*,c_n^*,n}[t]P_{\alpha_n^*,c_n^*,n}+C_2,\end{aligned}$$

where C_2 is a constant. By the definition of the capacity region Λ, there exist nonnegative $\gamma_{\alpha,c,n}$'s such that for any node n,

$$\sum_{\alpha,c}\frac{\gamma_{\alpha,c,n}}{P_{\alpha,c,n}}\le\frac{1}{1+\epsilon}. \tag{21}$$

Then

$$\begin{aligned}&-Q_{\alpha_n^*,c_n^*,n}[t]P_{\alpha_n^*,c_n^*,n}\\ =&-\frac{\epsilon}{1+\epsilon}Q_{\alpha_n^*,c_n^*,n}[t]P_{\alpha_n^*,c_n^*,n}\\ &-\frac{1}{1+\epsilon}Q_{\alpha_n^*,c_n^*,n}[t]P_{\alpha_n^*,c_n^*,n}\\ \le&-\frac{\epsilon}{1+\epsilon}\frac{1}{|\mathcal{K}||\mathcal{C}|}\sum_{\alpha,c}P_{\alpha,c,n}Q_{\alpha,c,n}[t]\\ &-\sum_{\alpha,c}\frac{\gamma_{\alpha,c,n}}{P_{\alpha,c,n}}Q_{\alpha_n^*,c_n^*,n}[t]P_{\alpha_n^*,c_n^*,n}\\ \le&-\frac{\epsilon}{1+\epsilon}\frac{1}{|\mathcal{K}||\mathcal{C}|}\sum_{\alpha,c}P_{\alpha,c,n}Q_{\alpha,c,n}[t]\\ &-\sum_{\alpha,c}\gamma_{\alpha,c,n}Q_{\alpha,c,n}[t],\end{aligned}$$

and therefore,

$$\begin{aligned}&\mathbb{E}\left[-\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]S_{\alpha,c,n}[t]\;\middle|\;Q[t]\right]\\ \le&-\sum_{\alpha,c,n}\gamma_{\alpha,c,n}Q_{\alpha,c,n}[t]-\frac{\epsilon}{1+\epsilon}\frac{1}{|\mathcal{K}||\mathcal{C}|}\sum_{\alpha,c}P_{\alpha,c,n}Q_{\alpha,c,n}[t]\\ &+nC_2.\end{aligned} \tag{22}$$

For the summands corresponding to task transmission, there holds

$$\begin{aligned}&\mathbb{E}\Bigg[-\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]\sum_{k:(n,k)\in\mathcal{E}}S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]\\ &\quad+\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]\sum_{m:(m,n)\in\mathcal{E}}S^{\mathrm{I}}_{\alpha,c,(m,n)}[t]\;\Bigg|\;Q[t]\Bigg]\\ =&\mathbb{E}\left[-\sum_{(n,k)\in\mathcal{E}}\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t])S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]\;\middle|\;Q[t]\right].\end{aligned}$$

Let

$$(\alpha^*_{(n,k)},c^*_{(n,k)})\in\arg\max_{\alpha,c}\frac{1}{l_\alpha}\left(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t]\right)-q_{\mathrm{th}}.$$

Then for any $(n,k)\in\mathcal{E}$,

$$\begin{aligned}&-\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t])S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]\\ \le&-\left(\frac{1}{l_{\alpha^*_{(n,k)}}}\left(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\right)-q_{\mathrm{th}}\right)\\ &\cdot l_{\alpha^*_{(n,k)}}S^{\mathrm{I}}_{\alpha^*_{(n,k)},c^*_{(n,k)},(n,k)}[t]-q_{\mathrm{th}}\sum_{\alpha,c}l_\alpha S^{\mathrm{I}}_{\alpha,c,(n,k)}[t],\end{aligned} \tag{23}$$

where the last term $q_{\mathrm{th}}\sum_{\alpha,c}l_\alpha S^{\mathrm{I}}_{\alpha,c,(n,k)}[t]$ can be further removed since it is nonnegative. When $\left(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\right)/l_{\alpha^*_{(n,k)}}-q_{\mathrm{th}}\le 0$, the quantities on the both sides of (23) are 0. When $\left(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\right)/l_{\alpha^*_{(n,k)}}-q_{\mathrm{th}}>0$, there holds

$$\begin{aligned}&-\left(\frac{1}{l_{\alpha^*_{(n,k)}}}\left(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\right)-q_{\mathrm{th}}\right)\\ &\cdot l_{\alpha^*_{(n,k)}}S^{\mathrm{I}}_{\alpha^*_{(n,k)},c^*_{(n,k)},(n,k)}[t]\\ \le&-\left(\frac{1}{l_{\alpha^*_{(n,k)}}}\left(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\right)-q_{\mathrm{th}}\right)\\ &\cdot C_{(n,k)}+C_3,\end{aligned}$$

where C_3 is a constant and again we have used the boundedness of processing and transmission capacity. By the definition of the capacity region Λ, there exist nonnegative $\mu_{\alpha,c,(n,k)}$'s such that

$$\sum_{\alpha,c}l_\alpha\mu_{\alpha,c,(n,k)}\le C_{(n,k)}. \tag{24}$$

Then

$$
\begin{aligned}
&-\left(\frac{1}{l_{\alpha^*_{(n,k)}}}\Big(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\Big)-q_{\text{th}}\right)\\
&\quad\cdot C_{(n,k)}\\
&\le-\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t]-l_\alpha q_{\text{th}})\mu_{\alpha,c,(n,k)}.
\end{aligned}
$$

Thus the inequality

$$
\begin{aligned}
&-\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t])S^{\text{I}}_{\alpha,c,(n,k)}[t]\\
&\le-\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t]-l_\alpha q_{\text{th}})\mu_{\alpha,c,(n,k)}+C_3
\end{aligned}
$$

holds for both the case that $\big(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\big)/l_{\alpha^*_{(n,k)}}-q_{\text{th}}\le 0$ and the case that $\big(Q_{\alpha^*_{(n,k)},c^*_{(n,k)},n}[t]-Q_{\alpha^*_{(n,k)},c^*_{(n,k)},k}[t]\big)/l_{\alpha^*_{(n,k)}}-q_{\text{th}}>0$. Therefore,

$$
\begin{aligned}
&\mathbb{E}\Bigg[-\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]\sum_{k:(n,k)\in\mathcal{E}}S^{\text{I}}_{\alpha,c,(n,k)}[t]\\
&\qquad+\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]\sum_{m:(m,n)\in\mathcal{E}}S^{\text{I}}_{\alpha,c,(m,n)}[t]\;\Bigg|\;Q[t]\Bigg]\\
&\le-\sum_{(n,k)\in\mathcal{E}}\sum_{\alpha,c}(Q_{\alpha,c,n}[t]-Q_{\alpha,c,k}[t]-l_\alpha q_{\text{th}})\mu_{\alpha,c,(n,k)}\\
&\quad+n^2C_3\\
&=-\sum_{\alpha,c,n}\left(\sum_{k:(n,k)\in\mathcal{E}}\mu_{\alpha,c,(n,k)}-\sum_{m:(m,n)\in\mathcal{E}}\mu_{\alpha,c,(m,n)}\right)\\
&\qquad\cdot Q_{\alpha,c,n}[t]\\
&\quad+q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu_{\alpha,c,(m,n)}+n^2C_3.
\end{aligned}
\tag{25}
$$

Now we combine the bounds on the summands in (19), (22) and (25) to bound the drift. By Theorem 1,

$$
\begin{aligned}
&\sum_{d:\alpha_d=\alpha}\lambda_{d,c,n}+\sum_{m:(m,n)\in\mathcal{E}}\mu_{\alpha,c,(m,n)}\\
&\qquad\le\sum_{k:(n,k)\in\mathcal{E}}\mu_{\alpha,c,(n,k)}+\gamma_{\alpha,c,n},\quad\forall\alpha,c,n.
\end{aligned}
$$

Thus

$$
\begin{aligned}
&\mathbb{E}[V(Q[t+1])-V(Q[t])\mid Q[t]]\\
&\le-\frac{2\epsilon}{1+\epsilon}\frac{1}{|\mathcal{K}||\mathcal{C}|}\sum_{\alpha,c,n}P_{\alpha,c,n}Q_{\alpha,c,n}[t]\\
&\quad+q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu_{\alpha,c,(m,n)}+C_4,
\end{aligned}
\tag{26}
$$

where $C_4=C_1+nC_2+n^2C_3$. Note that $P_{\alpha,c,n}>0$ for any α, c and n. Therefore, there exists $\delta>0$ such that the drift is smaller than $-\delta$ for states outside a finite set and the drift is bounded for states within the finite set. By Foster-Lyapunov theorem $(Q[t]\colon t\ge 0)$ is positive recurrent.

Since the Markov chain $(Q[t]\colon t\ge 0)$ is irreducible, aperiodic and positive recurrent, its distribution converges to the stationary distribution, and in the steady state, there holds

$$
\mathbb{E}[V(Q[t+1]-V(Q[t]))]=0.
$$

We now assume that the Markov chain is in the steady state and take expectation on both sides of (26), which gives

$$
\begin{aligned}
0\le&-\frac{2\epsilon}{1+\epsilon}\frac{1}{|\mathcal{K}||\mathcal{C}|}\sum_{\alpha,c,n}P_{\alpha,c,n}\mathbb{E}[Q_{\alpha,c,n}[t]]\\
&+q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu_{\alpha,c,(m,n)}+C_4.
\end{aligned}
$$

Thus

$$
\begin{aligned}
&\mathbb{E}\left[\sum_{\alpha,c,n}Q_{\alpha,c,n}[t]\right]\\
&\le q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu_{\alpha,c,(m,n)}\frac{(1+\epsilon)|\mathcal{K}||\mathcal{C}|}{2\epsilon P_{\min}}\\
&\quad+\frac{C_4(1+\epsilon)|\mathcal{K}||\mathcal{C}|}{2\epsilon P_{\min}},
\end{aligned}
$$

where $P_{\min}=\min\{P_{\alpha,c,n}\colon\alpha\in\mathcal{K},c\in\mathcal{C},n\in\mathcal{N}\}$. This implies the bound in (14).

To prove the bound on the amount of transmitted data in (15), we consider the optimal solution $(\lambda^*,\mu^*,\gamma^*)$ instead of the rates in (18), (21) and (24). Using similar arguments, except that here we do not remove the last term in (23), we obtain the following bound on the drift

$$
\begin{aligned}
&\mathbb{E}[V(Q[t+1])-V(Q[t])\mid Q[t]]\\
&\le-q_{\text{th}}\sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}}l_\alpha S^{\text{I}}_{\alpha,c,(n,k)}[t]\\
&\quad+q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu^*_{\alpha,c,(m,n)}+C_4.
\end{aligned}
\tag{27}
$$

Again we assume that the Markov chain is in the steady state and take expectation on both sides of (27). Then

$$
\begin{aligned}
0\le&-q_{\text{th}}\mathbb{E}\left[\sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}}l_\alpha S^{\text{I}}_{\alpha,c,(n,k)}[t]\right]\\
&+q_{\text{th}}\sum_{(m,n)\in\mathcal{E}}\sum_{\alpha,c}l_\alpha\mu^*_{\alpha,c,(m,n)}+C_4,
\end{aligned}
$$

and thus

$$
\mathbb{E}\left[\sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}}l_\alpha S^{\text{I}}_{\alpha,c,(n,k)}[t]\right]
\tag{28}
$$

$$
\le\sum_{\alpha,c}\sum_{(n,k)\in\mathcal{E}}l_\alpha\mu^*_{\alpha,c,(n,k)}+\frac{C_4}{q_{\text{th}}}.
\tag{29}
$$

This implies the bound in (15) and completes the proof. □

V. Conclusions

In this paper, we studied the resource allocation problem of parallel-data computing. Due to the coupling between the data and computation in each task, the problem involves data storage, communication and computing. Our end goal is to support efficient data-processing in the system, so unnecessary transmission should be avoided to reduce the usage of bandwidth resources, while high throughput should be maintained. This led to the networking architecture and the resource allocation algorithm we proposed. We first classified the data into classes and let each node maintain a queue for each $(\text{data class}, \text{computation operation})$ pair. Then the algorithm allocates computation loads and schedules data transmission according to these queue lengths. We proved that our proposed algorithm is throughput optimal, i.e., it stabilizes the system for the maximum range of incoming traffic intensity. Beyond throughput, our algorithm can also control the tradeoff between the amount of data transmission and the expected number of backlogged tasks in steady state through a parameter q_{th}. We showed that the gap between the expected data transmission rate under our algorithm and the optimal value is $O(1/q_{\text{th}})$, and the expected number of total backlogged tasks is upper bounded by $O(q_{\text{th}})$.

VI. Acknowledgement

This work was supported in part by the NSF under Grant ECCS-1255425 and ECCS-1609202.

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] 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.

[3] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. USENIX Conf. Hot Topics in Cloud Computing (HotCloud)*, Boston, MA, Jun. 2010, pp. 10–10.

[4] 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.

[5] W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Map task scheduling in MapReduce with data locality: throughput and heavy-traffic optimality," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Turin, Italy, Apr. 2013, pp. 1609–1617.

[6] Q. Xie and Y. Lu, "Priority algorithm for near-data scheduling: Throughput and heavy-traffic optimality," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Hong Kong, China, Apr. 2015, pp. 963–972.

[7] W. Wang, M. Barnard, and L. Ying, "Decentralized scheduling with data locality for data-parallel computation on peer-to-peer networks," in *Proc. Ann. Allerton Conf. Communication, Control and Computing*, Monticello, IL, Sep. 2015, pp. 337–344.

[8] W. Wang and L. Ying, "Data locality in MapReduce: A network perspective," *Perform. Eval.*, pp. 1–11, Feb. 2016.

[9] Q. Xie, A. Yekkehkhany, and Y. Lu, "Scheduling with multi-level data locality: Throughput and heavy-traffic optimality," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, San Francisco, CA, Apr. 2016.

[10] L. Bui, R. Srikant, and A. Stolyar, "Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, Rio de Janeiro, Brazil, Apr. 2009, pp. 2936–2940.

[11] L. Ying, S. Shakkottai, A. Reddy, and S. Liu, "On combining shortest-path and back-pressure routing over multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 841–854, Jun. 2011.

[12] 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. 37, pp. 1936–1948, Dec. 1992.

[13] R. Srikant and L. Ying, *Communication Networks: An Optimization, Control and Stochastic Networks Perspective*. New York: Cambridge Univ. Press, 2014.