An optimization approach to load balancing, scheduling and right sizing of cloud computing systems with data locality

Fernando Paganini, Diego Goldsztajn and Andres Ferragut
Universidad ORT Uruguay

Abstract— **We model a cloud computing infrastructure over a set of locations, with multiple server instances per location. The service rate offered by each server is differentiated by the type of task, depending on whether its data is locally available. Resource allocation questions for such systems include load balancing of tasks between locations, scheduling of tasks within each location, and sizing of the active server population at each location.**

Using a fluid queue model, we first characterize the capacity region of a system with a fixed number of servers at each location, recovering known results on throughput optimality of certain policies. Next we allow the server populations to vary, and pose the problem of minimizing a convex cost function subject to load stabilization. Such *right sizing* of service capacity is most interesting when it can be done dynamically, without knowledge of the load. Invoking Lagrange duality, we propose a primal-dual dynamic control with queues and server populations as state variables, that also embeds the optimal load balancing and scheduling. Its Lyapunov stability is established, and illustrative simulations are given.

I. Introduction

Resource sharing has been at the center of the evolution of both telecommunication networks and computing systems. In the former, the transition from circuit to packet switching was made long ago, mutualizing the communication substrate among the large number of end-to-end connections. In computation, a similar but more recent development are cloud computing clusters made up of thousands of machines, whose processing power is dynamically shared among a large number of applications.

Managing such large scale computer infrastructures are resource allocation algorithms: *load balancing* policies that distribute tasks between different servers; *scheduling* algorithms that prioritize tasks at each server; and *right-sizing* rules that control the number of active servers. These control laws interact in non-trivial ways: on their integrated performance rests the tradeoff between the requirement of low task latency and the provisioning cost of the infrastructure, with its related energy consumption.

An additional feature of cloud computing involving large datasets are distributed file systems (e.g. [1]) which divide data into chunks and store multiple replicas of each at different *locations*. In the MapReduce framework [2], large *jobs* are broken up into many *tasks*, and routed for parallel processing at servers in each location. Their processing time will, however, depend on whether the server has the data stored locally, or must retrieve it from a remote location. Following [3], we model the situation by a differentiated service rate for local versus remote tasks; this *data locality* issue adds a new level of complexity to the resource allocation question.

The recent literature on mathematical modeling for these problems [3], [4] uses stochastic queueing tools to characterize the capacity region of a cluster, and its heavy traffic behavior. Here service is provided by a fixed set of machines. On the other hand, the literature on speed-scaling or right-sizing [5]–[9] allows for the dynamic variation of server resources, but does not consider issues of data locality. Our purpose in this paper is to integrate these different resource allocation questions.

Returning to our initial parallel with communication networks, in that case substantial progress was made in cross-layer resource allocation by using continuous variable (fluid) models and convex optimization tools [10], [11]. Through this technique, decomposition strategies were found that yield decentralized control algorithms, with provable performance properties. Our main contribution here is carrying this approach through to the aforementioned cloud computing problems.

The paper is organized as follows. Section II contains the problem formulation and fluid variable models. In Section III we study the case of fixed service capacity systems, recovering results of [3] in this fluid setting. Adding the server right sizing issue, we formulate in Section IV a convex optimization problem, and in Section V we present a dynamic solution, based on a suitable primal-dual control law. Conclusions are given in Section VI.

II. Problem formulation

Server *locations* are indexed by $j = 1, \dots, N$. At each location j we have s_j servers of equal capacity, which we may think of as sharing the same physical rack. Alternatively they could share a larger local facility; the key assumption is that servers at location j have access to the same local data, and therefore offer the same service rates to arriving tasks.

Tasks arriving into the system are classified in *types*[1] $i = 1, \dots, M$; tasks of the same type will receive the same level of service from all servers in each rack. Let μ_{ij} be the maximum service rate (in e.g. tasks/sec) that a single server at location j can provide to a task of type i. We denote by λ_i the arrival rate of tasks of type i, an exogenous quantity.

Research supported by ANII-Uruguay under Grant FCE_1_2017_1_136748. E-mail:paganini@ort.edu.uy.

[1]For instance, in [3] the task type is defined by the three machines that are local for its respective data chunks.

The decision variables that must be controlled are:

- The rates λ_{ij} in which tasks of type i are split between server locations, subject to the conservation constraint

$$\sum_{j=1}^{N} \lambda_{ij} = \lambda_i, \quad i = 1, \dots, M. \tag{1}$$

  We denote by $\Lambda \in \mathbb{R}_+^{M \times N}$ the matrix formed by the λ_{ij}. The selection of this matrix subject to the conservation constraint is the *load balancing* problem.
- The *aggregate* service rate r_{ij} (in tasks/sec) that location j provides to tasks of type i. Such an assignment will consume a quantity $\frac{r_{ij}}{\mu_{ij}}$ of server instances; thus the constraint on overall use of server resources at each location across all types is:

$$\sum_{i=1}^{M} \frac{r_{ij}}{\mu_{ij}} \leq s_j, \quad j = 1, \dots, N. \tag{2}$$

  An implicit assumption is that servers can time-share their capacity among different tasks. We denote by $R \in \mathbb{R}_+^{M \times N}$ the matrix formed by the r_{ij}. The choice of this matrix subject to the above resource constraint is the *scheduling* problem.
- In much of the literature the service capacities $\{s_j\}_{j=1}^N$ are assumed given, and the question is to characterize the capacity region of such system in terms of the vector $\lambda = (\lambda_i)$ of arrival rates that can be supported. We will revisit this question, with optimization tools, in Section III. A more interesting possibility arises when the number s_j of active servers is itself subject to control (in practice, by turning ON or OFF active server instances). This *right sizing* problem has been studied often in terms of a single server class [5], [9], [12], but not to our knowledge in the multi-locality situation. We will tackle this question in Sections IV and V.

As a final modeling step, we will use a *fluid* model of the task queues present in this system. Let q_{ij} denote the number of tasks of type i in service at location j; its dynamics is given by:

$$\dot{q}_{ij} = [\lambda_{ij} - r_{ij}]^+_{q_{ij}}. \tag{3}$$

This is simply flow balance plus a saturation to maintain positive queues. We use above the *positive projection* notation: $[z]_q^+ = z$ if $q > 0$ or $z \geq 0$, and $[z]_q^+ = 0$ if $q = 0, z < 0$; the projection is called *active* in the latter case.

Denote also by Q the corresponding matrix of queues. A compact matrix form for (3) across i, j is:

$$\dot{Q} = [\Lambda - R]^+_Q. \tag{4}$$

We will denote by $\langle \cdot, \cdot \rangle$ the standard componentwise inner product for vectors, and also for $M \times N$ matrices:

$$\langle Q, R \rangle = \sum_{i,j} q_{ij} r_{ij}.$$

In the latter case the corresponding Frobenius norm is, $\|Q\|_F = \sqrt{\langle Q, Q \rangle}$.

III. Capacity region and throughput optimal policies for fixed server systems

This question has been studied in the queueing literature, from a stochastic perspective. We believe it is useful to formulate a fluid version of this problem, and characterize its solution by tools of convex optimization.

For the remainder of this section, $s = (s_j)$ is taken to be a **fixed** vector.

Definition 1: The capacity region of our server system is the set $\mathcal{C} \subset \mathbb{R}^M$ of vector rates $\lambda = (\lambda_i)$, such that there exists a control law for $(\Lambda(t), R(t))$ as a function of the state $Q(t)$, satisfying constraints (1-2), such that the solution to (4) is stable in the Lyapunov sense.

We proceed to characterize the capacity region, by introducing the constraint in $\Lambda = (\lambda_{ij})$:

$$\sum_{i=1}^{M} \frac{\lambda_{ij}}{\mu_{ij}} < s_j, \quad j = 1, \dots, N, \tag{5}$$

and defining the region

$$\mathcal{C}_\mu := \left\{ \lambda \in \mathbb{R}_+^M : \exists \Lambda \in \mathbb{R}^{M \times N} \text{ satisfying (1) and (5)} \right\}.$$

We will also consider its closure $\overline{\mathcal{C}}_\mu$ in $\mathbb{R}^M$, with the same definition except the inequality in (5) becomes nonstrict. We have the following characterization of the capacity region:

Theorem 1: $\mathcal{C}_\mu \subset \mathcal{C} \subset \overline{\mathcal{C}}_\mu$.

The above statement includes two separate inclusions, which we will prove in succession. For the first we will rely on a specific stabilizing policy, already known in the queueing literature for this kind of problem [3]: a combination of Join-the-Shortest Queue load balancing with Max-Weight scheduling. In our context, these may be defined as follows:

$$\text{JSQ}: \quad \Lambda^*(Q) := \arg\min_{\Lambda} \langle \Lambda, Q \rangle, \text{ subject to (1)}; \tag{6}$$

$$\text{MW}: \quad R^*(Q) := \arg\max_{R} \langle R, Q \rangle, \text{ subject to (2)}; \tag{7}$$

We elaborate some more on the structure of these policies. Starting with (6), the minimum of $\sum_i \left[\sum_j \lambda_{ij} q_{ij} \right]$ subject to (1) decouples over i, since the constraints are decoupled. For each i, clearly the minimizing $\{\lambda_{ij}^*\}_{j=1}^N$ is achieved by placing the total mass λ_i at the smallest q_{ij}; hence the JSQ nomenclature. Ties can be broken arbitrarily. The optimum, for future reference, is

$$\langle \Lambda^*(Q), Q \rangle = \sum_i \lambda_i \min_j (q_{ij}). \tag{8}$$

For the maximization in (7) we have the cost $\sum_j [\sum_i r_{ij} q_{ij}]$, and the constraints (2), decoupled over j. For a fixed j we have the problem

$$\max_r \sum_i r_{ij} q_{ij}, \quad \text{subject to} \quad \sum_i \frac{r_{ij}}{\mu_{ij}} \leq s_j,$$

which may be transformed by the change of variables $z_{ij} = \frac{r_{ij}}{\mu_{ij} s_j}$ into:

$$\max_z s_j \sum_i z_{ij} \mu_{ij} q_{ij}, \quad \text{subject to} \quad \sum_i z_{ij} \leq 1.$$

In this form the optimum is clearly $s_j \max_i(\mu_{ij} q_{ij})$, and the maximizing schedule assigns nonzero server rate r^*_{ij} only to task types i which maximize the *weight* $\mu_{ij} q_{ij}$, with an arbitrary split if there are ties.

Aggregating over j we have:

$$\langle R^*(Q), Q\rangle = \sum_j s_j \max_i(\mu_{ij} q_{ij}). \tag{9}$$

Armed with these definitions we now prove the following result on the capacity region:

Proposition 2: Suppose $\lambda \in \mathcal{C}_\mu$. Then the feedback laws $\Lambda^*(Q)$ and $R^*(Q)$ are such that the closed loop dynamics

$$\dot{Q} = [\Lambda^*(Q) - R^*(Q)]^+_Q \tag{10}$$

is Lyapunov stable.

Remark 1: The above control laws involve switching: when the shortest queues or maximum weights change the control will respond discontinuously. This raises technical difficulties in terms of existence and uniqueness of solutions for (10), which must be defined in a generalized sense; this is the subject of a specialized literature, see e.g. [13]. We will not address these issues in the present paper, and assume in the proof below that classical differentiation of a suitable Lyapunov function is allowed.

Proof: We first note that if $q_{ij} = 0$ for a certain queue, without loss of generality one can assume that at the maximum in (7) the corresponding $r^*_{ij} = 0$; hence the positive projection will never be active in (10).

By hypothesis, there exists a fixed $\Lambda \in R_+^{M\times N}$ satisfying both (1) and (5). Since $\Lambda^*(Q)$ is minimizing among all Λ satisfying (1), we have

$$\langle \Lambda^*, Q\rangle \le \langle \Lambda, Q\rangle.$$

Similarly, Λ satisfies the contraint (in R) given by (2), among which $R^*(Q)$ is maximizing. This implies that

$$\langle \Lambda, Q\rangle \le \langle R^*, Q\rangle;$$

We conclude that $\langle \Lambda^*, Q\rangle \le \langle R^*, Q\rangle$ and hence that

$$\langle \dot{Q}, Q\rangle = \langle \Lambda^*(Q) - R^*(Q), Q\rangle \le 0;$$

the Lyapunov function $V(Q) = \frac{1}{2}\langle Q, Q\rangle = \frac{1}{2}\|Q\|_F^2$ is decreasing across trajectories, implying the desired stability. ∎

We now turn to the second part of Theorem 1, that provides an outer bound for the capacity region. We will rely on the following result on *strong alternatives* from convex optimization (see [14], Section 5.8.2). Given the set

$$\mathcal{X} = \{x \in \mathbb{R}^d_+ : h_i(x) = 0, i = 1, \dots, M; \\ f_j(x) \le 0, j = 1, \dots N\},$$

where $h_i(x)$ are affine functions, and $f_j(x)$ are convex functions. Construct the Lagrangian

$$L(x, \eta, \gamma) = \sum_i \eta_i h_i(x) + \sum_j \gamma_j f_j(x),$$

the dual function $g(\eta, \gamma) = \inf_{x\in\mathbb{R}^d_+} L(x, \eta, \gamma)$, and the set

$$\mathcal{Y} = \{(\eta, \gamma) \in \mathbb{R}^{M+N} : \gamma \ge 0, g(\eta, \gamma) > 0\}.$$

Assume: there exists x in the interior of $\mathbb{R}^d_+$ satisfying the equality constraints $h_i(x) = 0\ \forall i$, and also $\max_j f_j(x) \to \infty$ as $x \to \infty$; then $\mathcal{X}$ and $\mathcal{Y}$ are strong alternatives, i.e. exactly one of them is nonempty.

This result will be used to establish that for arrival rates $\lambda = (\lambda_i)$ outside $\overline{\mathcal{C}}_\mu$ there is *no* stabilizing control law.

Proposition 3: Suppose $\lambda \notin \overline{\mathcal{C}}_\mu$. For any $\Lambda(t), R(t)$ satisfying (1)-(2) at all t, the resulting solution $Q(t)$ of (3) is unbounded.

Proof: By hypothesis, the set $\mathcal{X} \subset \mathbb{R}^{M\times N}_+$ defined by the affine constraints (1), and the convex constraints

$$\sum_{i=1}^M \frac{\lambda_{ij}}{\mu_{ij}} \le s_j, \quad j = 1, \dots, N,$$

both in the variable Λ, is *empty*. Also, these constraints verify the assumptions of the cited strong alternative result, so the corresponding set $\mathcal{Y}$ must be non-empty. To find it we write the Lagrangian

$$\begin{aligned} L(\Lambda, \eta, \gamma) &= \sum_i \eta_i \Big(\lambda_i - \sum_j \lambda_{ij}\Big) + \sum_j \gamma_j \Big(\sum_i \frac{\lambda_{ij}}{\mu_{ij}} - s_j\Big) \\ &= \sum_i \eta_i \lambda_i - \sum_j \gamma_j s_j + \sum_{i,j} \lambda_{ij} \left(\frac{\gamma_j}{\mu_{ij}} - \eta_i\right). \end{aligned} \tag{11}$$

Taking the infimum over $\lambda_{ij} \ge 0$ we obtain the dual function

$$g(\eta, \gamma) = \begin{cases} \eta^T \lambda - \gamma^T s & \text{if } \frac{\gamma_j}{\mu_{ij}} \ge \eta_i \ \ \forall i, j; \\ -\infty & \text{otherwise.} \end{cases}$$

The strong alternative result implies there exist $\gamma^* \ge 0$, η^* satisfying:

$$\frac{\gamma^*_j}{\mu_{ij}} \ge \eta^*_i \quad \forall i, j; \tag{12}$$

$$\sum_i \eta^*_i \lambda_i > \sum_j \gamma^*_j s_j. \tag{13}$$

Without loss of generality we can take $\eta^* \ge 0$.

Consider now an arbitrary $\Lambda(t), R(t)$ satisfying (1)-(2), and the queue trajectory $Q(t)$ given by (4). Let

$$f(t) := \sum_{i,j} \eta^*_i q_{ij}(t);$$

its derivative along trajectories is

$$\begin{aligned}\dot{f}(t) &= \sum_{i,j} \eta_i^* [\lambda_{ij}(t) - r_{ij}(t)]^+_{q_{ij}(t)} \\ &\geq \sum_{i,j} \eta_i^* [\lambda_{ij}(t) - r_{ij}(t)] \\ &= \sum_i \eta_i^* \lambda_i - \sum_{ij} \eta_i^* r_{ij}(t) \\ &\geq \sum_i \eta_i^* \lambda_i - \sum_{ij} \frac{\gamma_j^*}{\mu_{ij}} r_{ij}(t) \\ &= \sum_i \eta_i^* \lambda_i - \sum_j \gamma_j^* \sum_i \frac{r_{ij}(t)}{\mu_{ij}} \\ &\geq \sum_i \eta_i^* \lambda_i - \sum_j \gamma_j^* s_j > 0.\end{aligned}$$

The first inequality follows from $\eta_i^* \geq 0$: terms with active projection are negative in the second summation. We then invoke (1), and then (12) for the next inequality. The final steps use (2) and (13).

The above implies that $f(t)$ has superlinear growth, hence it is unbounded and thus so is $Q(t)$. ∎

IV. Optimizing service capacity that supports a given load

We have assumed so far that server resources are fixed, and thus so is the capacity region. Modern day cloud systems offer, however, the possibility of adapting the number of active server instances to the external load. In practice, this is done by summoning servers from the sleep mode, or returning them to that state, as dictated by the load requirement.

In the classical case of a single task type and server class, the required minimum number of active servers is $s = \lambda/\mu$, the *offered load* in queueing terminology. However, in our case with multiple service speeds and arrival rates, motivated by the data locality question, constraints (5) leave many degrees of freedom: how many servers to activate at each location becomes a non-obvious choice.

In this section we frame this resource allocation question in optimization terms: given a vector of loads per type $\lambda = (\lambda_i)$, find the server allocations $s = (s_j)$, and the traffic split matrix $\Lambda = (\lambda_{ij})$, that satisfy the capacity region restrictions, and in addition optimize a certain cost function in the allocated services:

Problem 1: Given $\lambda = (\lambda_i)$, minimize $c(s) = \sum_j c_j(s_j)$, in the variables (s, Λ) subject to:

$$\begin{aligned}\sum_{j=1}^N \lambda_{ij} &= \lambda_i, \quad i = 1, \ldots, M; \\ \sum_{i=1}^M \frac{\lambda_{ij}}{\mu_{ij}} &\leq s_j, \quad j = 1, \ldots, N; \\ s_j &\leq \bar{s}_j, \quad j = 1, \ldots, N.\end{aligned}$$

The last constraint above models a possible limitation in the maximum number of server instances per location (rack).

A. Minimizing total capacity

We will assume initially that $\bar{s}_j = \infty$ (resources are plentiful) and that the cost function is $c(s) = \sum_j s_j$; i.e., we would like to minimize the total number of servers allocated to serve the incoming load.

In this case we have a linear program, whose Lagrangian becomes (adding the cost to the Lagrangian in (11)):

$$\begin{aligned}L(s, \Lambda, \eta, \gamma) = &\sum_i \eta_i \lambda_i + \sum_j (1 - \gamma_j) s_j \\ &+ \sum_{i,j} \lambda_{ij} \left(\frac{\gamma_j}{\mu_{ij}} - \eta_i \right).\end{aligned}$$

Minimizing over $\lambda_{ij} \geq 0, s_j \geq 0$ for fixed $\gamma \geq 0, \eta$, the corresponding dual function is now

$$g(\eta, \gamma) = \begin{cases} \eta^T \lambda & \text{if } \gamma_j \leq 1, \quad \frac{\gamma_j}{\mu_{ij}} \geq \eta_i \ \forall i, j; \\ -\infty & \text{otherwise.} \end{cases}$$

The dual problem is therefore

$$\begin{aligned}&\max \eta^T \lambda \\ &\text{s.t.} \quad \eta_i \leq \frac{\gamma_j}{\mu_{ij}} \quad \forall i, j; \\ &\qquad 0 \leq \gamma_j \leq 1 \ \forall j.\end{aligned}$$

The second constraint can be eliminated (setting $\gamma_j^* = 1 \ \forall j$) and gives the problem

$$\max \sum_i \eta_i \lambda_i, \quad \text{s.t.} \quad \eta_i \leq \frac{1}{\mu_{ij}} \quad \forall i, j. \tag{14}$$

Note now that the constraints decouple across i, so the explicit solution for the optimal cost is

$$c^* = \sum_i \lambda_i \min_j \frac{1}{\mu_{ij}} = \sum_i \frac{\lambda_i}{\mu_i^*},$$

where we have denoted $\mu_i^* = \max_j \mu_{ij}$, the maximal service rate available to tasks of type i. We also can say the following about the optimal load balancing allocation:

Proposition 4: For Problem 1 under $c(s) = \sum_j s_j$ and $\bar{s}_j = \infty$, the optimal Λ^* is supported in

$$\{(i, j) : j = \arg\max \mu_{ij}\};$$

i.e., traffic is sent only to locations that provide the maximum service rate per task type.

Proof: The optimal multiplier for (14) is $\eta_i^* = 1/\mu_i^*$; for any j such that $\mu_{ij} < \mu_i^*$, we will have

$$\eta_i^* < \frac{1}{\mu_{ij}} = \frac{\gamma_j^*}{\mu_{ij}};$$

looking at the Lagrangian we see that the minimizing λ_{ij}^* is zero in this case. ∎

The interpretation of the above result is very natural: if resources are unlimited at each location, and the linear cost values all locations equally, there is no reason to send traffic to any location except the one that provides maximal service. The following simple example further illustrates this.

Example 1: Consider the case where $M = N$, and for each type i, location $j = i$ is the only one that holds the data locally and thus achieves the maximal μ_{ij}. In that case the optimal allocation matrix is diagonal, $\Lambda^* = \mathrm{diag}(\lambda_i)$, and $s_j^* = \lambda_j/\mu_j^*$: we have parallel locations serving each type, with no resource sharing among them.

For a concrete illustration: take $M = N = 3$, traffic intensities $\lambda_1 = 10$, $\lambda_2 = \lambda_3 = 2$, and the following matrix of maximum service rates per location:

$$\mu = (\mu_{ij}) = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}.$$

The linear cost $\sum_j s_j$ will give a diagonal optimum $\Lambda^* = \mathrm{diag}(10, 2, 2)$ and provision the servers

$$s_1^* = 5, \quad s_2^* = s_3^* = 1.$$

The highly asymmetrical server allocation of the preceding example may not be desirable. For instance if capacity limitations $\bar{s}_j < \infty$ are at play, one location may not be able to cope with the entire load, and resource sharing becomes inevitable. The resulting optimal allocation could be found by including the corresponding capacity constraints in the linear program.

B. Promoting resource sharing by a strictly convex cost

Alternatively, one could use a soft penalty approach to favor more symmetrical allocations, for instance using the quadratic cost function $c_j(s_j) = s_j^2/2$. Below we apply it to the preceding example.

Example 2: Under the same conditions in Example 1, but using the cost $c_j(s_j) = s_j^2/2$, the optimum of Problem 1, found numerically, is the load balancing

$$\Lambda^* = (\lambda_{ij}^*) = \begin{bmatrix} 8 & 1 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}.$$

together withe the server allocation

$$s_1^* = 4, \quad s_2^* = s_3^* = 2.$$

Overall, there is an additional server required but the distribution of resources across locations is more homogeneous.

V. Dynamic right sizing of the number of servers

The results in the preceding section assume one knows in advance the loads λ_i on each data type; in practice, these quantities are uncertain and possibly varying in time. One could think of some kind of online measurement, but a far more interesting proposal is for the system to adapt itself in feedback, controlling the number of server instances in real time. This *dynamic right sizing* has been studied in the context of data centers with homogeneous servers, see e.g. [5]. In principle, summoning or removing servers on the fly based on the state of the task queue offers the possibility of adapting to an uncertain load; the only issue is that lags in startup/shutoff must be considered [9].

We would like to extend this kind of feedback control to the present scenario of multiple service locations and different service rates per type. The natural state variable on which to base our control decisions is the set of queues q_{ij} in (3).

As with many other resource allocation problems in communication networks (see [11] and references therein), the optimization approach followed in this paper provides natural ways to produce such control designs. Typically, these are based on some kind of gradient dynamics on either primal or dual variables, or both, combined possibly with static partial optimization over some variables. There are often many interesting alternative solutions to the same problem.

We will focus here on one such alternative, which exploits the policies studied in Section III. We first consider the following reformulation of Problem 1:

Problem 2: Given $\lambda = (\lambda_i)$, minimize $c(s) = \sum_j c_j(s_j)$, in the non-negative variables (s, Λ, R) subject to

$$\lambda_{ij} \le r_{ij} \quad \forall i, j \tag{15}$$

$$\sum_{j=1}^{N} \lambda_{ij} = \lambda_i, \quad i = 1, \dots, M. \tag{16}$$

$$\sum_{i=1}^{M} \frac{r_{ij}}{\mu_{ij}} \le s_j, \quad j = 1, \dots, N. \tag{17}$$

For simplicity we have taken $\bar{s}_j = \infty$; we rely on the penalty $c_j(s_j)$, which we assume differentiable, strictly increasing and strictly convex, to enforce resource sharing. Except for this simplification, it should be transparent that the two problems are equivalent; we have just introduced explicitly the variable $R = (r_{ij})$ of allocated rates per task type, subject to the constraint (17) (re-stated from (2)); this is suitable for real-time control and will allow us to exploit the queue dynamics (3) in our solution. Note that constraints (15)-(17) are always feasible, and the problem has a well-defined minimum solution, unique in s due to strict convexity.

Consider for this problem the Lagrangian obtained from dualizing only constraints (15), and (suggestively) naming the multiplier Q:

$$L(s, \Lambda, R, Q) = \sum_j c_j(s_j) + \sum_{i,j} q_{ij}(\lambda_{ij} - r_{ij}).$$

Now minimizing over the remaining constraints (16)-(17) yields the reduced Lagrangian

$$\bar{L}(s, Q) := \min_{\Lambda, R} L(s, \Lambda, R, Q) \tag{18}$$

$$= \sum_j c_j(s_j) + \min_{\Lambda, R} \langle Q, \Lambda - R \rangle. \tag{19}$$

The last expression reveals that the required minimization has already been studied in Section III: the results correspond respectively to the Join-the-Shortest-Queue load balancing $\Lambda^*(Q)$ in (6), and the Max Weight scheduling in (7), which we now denote as $R^*(Q, s)$ since it depends on the server

populations which are now variable. This observation allows us state the following:

Proposition 5:

$$\bar{L}(s,Q) = \sum_j [c_j(s_j) - s_j \cdot \max_i(\mu_{ij} q_{ij})] + \sum_i \lambda_i \min_j q_{ij}, \tag{20}$$

and a supergradient of $\bar{L}$ with respect to the queue variables is given by

$$\frac{\partial \bar{L}}{\partial Q} = \Lambda^*(Q) - R^*(Q,s),$$

where $\Lambda^*(Q)$ and $R^*(Q,s)$ are defined respectively by (6) and (7).

Proof: In the definition (19) of $\bar{L}(s,Q)$, the variables Λ, R satisfy the (independent) constraints (16) and (17). We can thus decouple the minimization over each variable, leading respectively to the JSQ and MW solutions, and invoke (8) and (9) for the optimal costs. This shows (20).

Also, denoting

$$\varphi(Q) = \min_{\Lambda,R} \langle Q, \Lambda - R\rangle = \langle Q, \Lambda^*(Q) - R^*(Q)\rangle,$$

we have

$$\begin{aligned}\varphi(Q') &\le \langle Q', \Lambda^*(Q) - R^*(Q,s)\rangle \\ &= \varphi(Q) + \langle Q' - Q, \Lambda^*(Q) - R^*(Q,s)\rangle;\end{aligned}$$

adding $\sum_j c_j(s_j)$ to both sides we conclude that

$$\bar{L}(s,Q') \le \bar{L}(s,Q) + \langle Q' - Q, \Lambda^*(Q) - R^*(Q,s)\rangle \tag{21}$$

which is the desired supergradient property. ∎

Note that the two allocation decisions (JSQ, MW) can be considered instantaneous in comparison with the much slower dynamics of the queues and the number of servers.

Therefore, we may focus on designing a slower time-scale dynamics that converges to a saddle-point (minimum in s, maximum in Q) of the reduced Lagrangian $\bar{L}(s,Q)$, which is strictly convex in s and concave in Q. If such a point (s^*, Q^*) is found, supplementing it with the corresponding $\Lambda^*(Q^*)$, $R^*(Q^*, s^*)$ we will have a saddle point of $L(s,\Lambda,R,Q)$, hence a solution to Problem 2.

A standard method [15]–[17] to seek a saddle point of a convex-concave function $\bar{L}(s,Q)$ is a *primal-dual* gradient dynamics of the form

$$\dot{s}_j = \beta_j \left[-\frac{\partial \bar{L}}{\partial s_j}\right]^+_{s_j}; \tag{22}$$

$$\dot{q}_{ij} = \left[\frac{\partial \bar{L}}{\partial q_{ij}}\right]^+_{q_{ij}}. \tag{23}$$

In our case the function is differentiable in s, but piecewise linear in Q; the right-hand side of (23) invokes the supergradient already discussed. More specifically we have the control laws:

$$\dot{s}_j = \beta_j \left[\max_i(\mu_{ij} q_{ij}) - c_j'(s_j)\right]^+_{s_j}; \tag{24}$$

$$\dot{q}_{ij} = \lambda^*_{ij}(Q) - r^*_{ij}(Q,s), \tag{25}$$

where again:

- $\Lambda^*(Q)$ is Join-the-Shortest-Queue load balancing, which sends for each i the traffic λ_i to the location(s) with the lowest q_{ij}; ties are broken arbitrarily.
- $R^*(Q,s)$ is the Max-Weight schedule, which at location j assigns the s_j servers to the local queues with largest $\mu_{ij} q_{ij}$; ties are broken arbitrarily.

We note that:

- The dynamic right-sizing rule (24) updates the server population at each location j. The constant $\beta_j > 0$ can be assigned a practical interpretation, as a first order model for the time lags inherent in server summoning and deletion; see [9] for more discussion in a single location scenario.
- (25) coincides with natural queue dynamics (3) for each type of task i and server location j. Therefore the queues already implement the dual portion of our control. We have removed the positive projection from this equation, since whenever $q_{ij} = 0$ the MW schedule will assign $r^*_{ij} = 0$ to this queue.
- As in Section III, the component (25) exhibits switching, and would require a generalized solution notion. Again as mentioned in Remark 1, we will not consider these issues in the present paper.

In the classical reference [15], a general quadratic Lyapunov function is given for such primal-dual dynamics. We provide (and, for completeness, prove) the version for our situation.

Proposition 6: Let (s^*, Q^*) be a saddle point of $\bar{L}(s,Q)$. Then the Lyapunov function

$$V(s,Q) = \frac{1}{2}\sum_j \frac{(s_j - s_j^*)^2}{\beta_j} + \frac{1}{2}\|Q - Q^*\|_F^2. \tag{26}$$

is decreasing along trajectories of (24)-(25). In particular the saddle point is a stable equilibrium.

Proof: Differentiating along trajectories we have

$$\begin{aligned}\dot{V} &= \sum_j (s_j - s_j^*)\left[-\frac{\partial \bar{L}}{\partial s_j}\right]^+_{s_j} + \langle Q - Q^*, \frac{\partial \bar{L}}{\partial Q}\rangle \\ &\le \sum_j (s_j - s_j^*)\left[-\frac{\partial \bar{L}}{\partial s_j}\right] - \langle Q^* - Q, \Lambda^* - R^*\rangle,\end{aligned}$$

where in the second step positive projections in s have been removed. This is based on the fact that $(x-y)[z]^+_x \le (x-y)z$ (assuming $x, y \ge 0$): in the non-trivial case of an active projection, the left-hand side is zero and right-hand side is non-negative.

Now the first sum may be written as $\langle \frac{\partial \bar{L}}{\partial s}(s,Q), s^* - s\rangle$ and bounded above by $\bar{L}(s^*,Q) - \bar{L}(s,Q)$, invoking the first order conditions for convexity of $\bar{L}$ in s.

Similarly the supergradient condition (21) (for $Q' = Q^*$) applies to the second term, leading to:

$$\begin{aligned}\dot{V} &\le [\bar{L}(s^*,Q) - \bar{L}(s,Q)] + [\bar{L}(s,Q) - \bar{L}(s,Q^*)] \\ &= [\bar{L}(s^*,Q) - \bar{L}(s^*,Q^*)] + [\bar{L}(s^*,Q^*) - \bar{L}(s,Q^*)].\end{aligned}$$

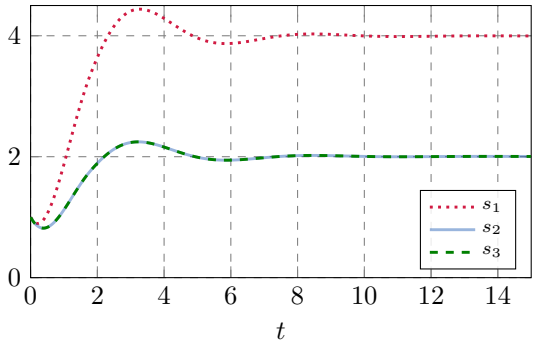

Fig. 1. Resource allocation $s = (s_j)$ across the locations $j = 1, 2, 3$ resulting from the primal-dual dynamics (22)-(23).

Both terms in brackets in the last expression are non-positive, because of the saddle point condition of $\bar{L}$ at (s^*, Q^*) (maximum in Q, minimum in s).

Therefore V is decreasing, and since it is defined as a norm-squared distance to the saddle point, we have stability in the Lyapunov sense: trajectories starting in a neighborhood of (s^*, Q^*) will stay in it. ■

The question arises as to whether we can claim *asymptotic* stability, in particular convergence to the saddle point. We note in this regard that the equilibrium Q^* may not be unique: in that case we may at most claim convergence to a set. We do expect, convergence in the variable s^* since this point is indeed unique due to the assumed strict convexity.

The usual method to establish asymptotic stability in these problems has been (see [16], [17]) the Lasalle invariance principle, essentially present also in [15] for the case of a smooth function $\bar{L}(s, Q)$.

The Lasalle theory is, however, more delicate for systems with switching. In particular regarding positive projections, [17] found flaws in the arguments given in [16], and provided an alternative proof based on the theory of projected dynamical systems [18]. In the current case, switching is not confined to projections, it also appears in the queue dynamics under JSQ/MW. This is related to the non-differentiability in Q of the reduced Lagrangian $\bar{L}(s, Q)$. Since we have not dealt with such issues we will refrain from stating a convergence theorem, although we conjecture that such a result holds for our primal-dual dynamics.

We end the section with a numerical simulation of our dynamic approach.

Example 3: Reconsider the system of Example 2, with quadratic cost $c_j(s_j) = s_j^2/2$, but in this case assuming that the arrival rates λ_i are unknown by the system. We use our primal-dual dynamics (22)-(23) to find the best resource allocation $s^* = (s_j^*)$. The corresponding dynamics solution is represented in Fig 1. We see a transient response that converges to the same optimal allocations of Example 2.

VI. Conclusions and Future Work

We have presented initial results on optimization-based control design for resource allocation problems in cloud computing, integrating server sizing, load balancing and scheduling with consideration of data locality. Some technical issues have been postponed for future work, particularly those arising from discontinuous dynamics.

Clearly our primal-dual law is not the only possible solution strategy: for instance, a smoother update of the primal variables Λ, R can be considered, replacing the static switching with a dynamic update of e.g. traffic splits, analogous to proposals in the area multipath routing [19].

Another line of inquiry could be the integration of data localization decisions (which determine service rates per type) into the resource allocation.

References

[1] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop distributed file system," in *2010 IEEE 26th symposium on mass storage systems and technologies (MSST)*, 2010, pp. 1–10.

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

[3] W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Maptask scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 190–203, 2016.

[4] Q. Xie, A. Yekkehkhany, and Y. Lu, "Scheduling with multi-level data locality: Throughput and heavy-traffic optimality," in *IEEE INFOCOM 2016*. IEEE, 2016, pp. 1–9.

[5] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 5, pp. 1378–1391, 2013.

[6] L. Chen and N. Li, "On the interaction between load balancing and speed scaling," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2567–2578, 2015.

[7] D. Goldsztajn, A. Ferragut, and F. Paganini, "A feedback control approach to dynamic speed scaling in computing systems," in *51st Annual Conference on Information Sciences and Systems (CISS)*, 2017.

[8] D. Goldsztajn, A. Ferragut, F. Paganini, and M. Jonckheere, "Controlling the number of active instances in a cloud environment," *ACM SIGMETRICS Perf. Eval. Review*, vol. 45, no. 2, pp. 15–20, 2018.

[9] D. Goldsztajn, A. Ferragut, and F. Paganini, "Feedback control of server instances for right sizing in the cloud," in *56th Annual Allerton Conference on Communication, Control, and Computing*, 2018.

[10] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[11] R. Srikant and L. Ying, *Communication networks: an optimization, control, and stochastic networks perspective*. Cambridge University Press, 2013.

[12] D. Mukherjee, S. Dhara, S. C. Borst, and J. S. van Leeuwaarden, "Optimal service elasticity in large-scale distributed systems," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 1, p. 25, 2017.

[13] J. Cortés, "Discontinuous dynamical systems: a tutorial on solutions, nonsmooth analysis, and stability," in *ArXiv*, 2009. [Online]. Available: https://arxiv.org/pdf/0901.3583.pdf

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2004.

[15] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies in linear and non-linear programming*. Cambridge Univ. Press, 1958.

[16] D. Feijer and F. Paganini, "Stability of primal–dual gradient dynamics and applications to network optimization," *Automatica*, vol. 46, no. 12, pp. 1974–1981, 2010.

[17] A. Cherukuri, E. Mallada, and J. Cortés, "Asymptotic convergence of constrained primal–dual dynamics," *Systems & Control Letters*, vol. 87, pp. 10–15, 2016.

[18] A. Nagurney and D. Zhang, *Projected dynamical systems and variational inequalities with applications*. Springer Science & Business Media, 2012, vol. 2.

[19] F. Paganini and E. Mallada, "A unified approach to congestion control and node-based multipath routing," *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 5, pp. 1413–1426, 2009.