

Temperature Aware Workload Management in Geo-distributed Datacenters

Hong Xu, Chen Feng, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto

ABSTRACT

For geo-distributed datacenters, lately a workload management approach that routes user requests to locations with cheaper and cleaner electricity has been shown promising in reducing the energy cost. We consider two key aspects that have not been explored before. First, the energy-gobbling cooling systems are often modeled with a location-independent efficiency factor. Yet, through empirical studies, we find that their actual energy efficiency depends directly on the ambient temperature, which exhibits a significant degree of geographical diversity. Temperature diversity can be used to reduce the overall cooling energy overhead. Second, datacenters run not only interactive workloads driven by user requests, but also delay tolerant batch workloads at the back-end. The elastic nature of batch workloads can be exploited to further reduce the energy consumption.

In this paper, we propose to make workload management for geo-distributed datacenters *temperature aware*. We formulate the problem as a joint optimization of request routing for interactive workloads and capacity allocation for batch workloads. We develop a distributed algorithm based on an *m*-block *alternating direction method of multipliers* (ADMM) algorithm that extends the classical 2-block algorithm. We prove the convergence of our algorithm under general assumptions. Through trace-driven simulations with real-world electricity prices, historical temperature data, and an empirical cooling efficiency model, we find that our approach is consistently capable of delivering a 15%–20% cooling energy reduction, and a 5%–20% overall cost reduction for geo-distributed clouds.

1. INTRODUCTION

Geo-distributed datacenters operated by organizations such as Google and Amazon are the powerhouses behind many Internet-scale services. They are deployed across the Internet to provide better latency and redundancy. These datacenters run hundreds of thousands of servers, consume megawatts of power with massive carbon footprint, and incur electricity bills of millions of dollars [17, 34]. Thus, the topic of reducing their energy consumption and cost has received significant attention [7, 11–13, 15, 17, 19, 26–29, 34, 35, 40].

Energy consumption of individual datacenters can be re-

duced with more energy efficient hardware and integrated thermal management [7, 11, 15, 28, 40]. Recently, important progress has been made on a new *workload management* approach that instead focuses on the overall energy cost of geo-distributed datacenters. It exploits the geographical diversity of electricity prices by optimizing the *request routing* algorithm to route user requests to locations with cheaper and cleaner electricity [12, 17, 18, 26, 27, 29, 34, 35].

In this paper, we consider two key aspects of geo-distributed datacenters that have not been explored in the literature.

First, cooling systems, which consume 30% to 50% of the total energy [33, 40], are often modeled with a constant and location-independent energy efficiency factor in existing efforts. This tends to be an over-simplification in reality. Through our study of a state-of-the-art production cooling system (Sec. 2), we find that temperature has direct and profound impact on cooling energy efficiency. This is especially true with *outside air cooling* technology, which has seen increasing adoption in mission-critical datacenters [1–3]. As we will show, its partial PUE (power usage effectiveness), defined as the sum of server power and cooling overhead divided by server power, varies from 1.30 to 1.05 when temperature drops from 35 °C (90 °F) to -3.9 °C (25 °F).

Through an extensive empirical analysis of daily and hourly climate data for 13 Google datacenters, we further find that temperature varies significantly across both time and location, which is intuitive to understand. These observations suggest that datacenters at different locations have distinct and time-varying cooling energy efficiency. This establishes a strong case for making workload management *temperature aware*, where such temperature diversity can be used along with price diversity in making request routing decisions to reduce the overall cooling energy overhead for geo-distributed datacenters.

Second, energy consumption comes not only from interactive workloads driven by user requests, but also from delay tolerant batch workloads, such as indexing and data mining jobs, that run at the back-end. Existing efforts focus mainly on request routing to minimize the energy cost of interactive workloads, which is only a part of the entire picture. Such a mixed nature of datacenter workloads, verified by measurement studies [36], provides more opportunities to utilize the

cost diversity of energy. The key observation is that batch workloads are elastic to resource allocations, whereas interactive workloads are highly sensitive to latency and have more profound impact on revenue [25]. Thus at times when one location is comparatively cost efficient (in terms of dollar per unit energy), we can increase the capacity for interactive workloads by reducing the resources for batch jobs. More requests can then be routed to and processed at this location, and the cost saving can be more substantial. We thus advocate a holistic workload management approach, where *capacity allocation* between interactive and batch workloads is dynamically optimized with request routing. Dynamic capacity allocation is also technically feasible because jobs run on highly scalable systems such as MapReduce.

Towards temperature aware workload management, we propose a general framework to capture the important trade-offs involved (Sec. 3). We model both energy cost and utility loss, which correspond to performance-related revenue reduction. We develop an empirical cooling efficiency model based on a production system. The problem is formulated as a joint optimization of request routing and capacity allocation. The technical challenge is then to develop a distributed algorithm to solve the large-scale optimization with tens of millions of variables for a production geo-distributed cloud. Dual decomposition with subgradient methods are often used to develop distributed optimization algorithms. However they require delicate adjustments of step sizes that make convergence difficult to achieve for large-scale problems. The method of multipliers [22] achieves fast convergence, at the cost of tight coupling among variables.

We rely on the *alternating direction method of multipliers* (ADMM), a simple yet powerful algorithm that blends the advantages of the two approaches. ADMM recently has found practical use in many large-scale distributed convex optimization problems in machine learning and data mining [10]. It works for problems whose objective and variables can be divided into *two* disjoint parts. It alternatively optimizes part of the objective with one block of variables to iteratively reach the optimum. Our formulation has three blocks of variables, yet little is known about the convergence of m -block ($m \geq 3$) ADMM algorithms, with two exceptions [20, 23] very recently. [20] establishes the convergence of m -block ADMM for strongly convex objective functions, but not linear convergence; [23] shows the linear convergence of m -block ADMM under the assumption that the relation matrix is full column rank, which is, however, not the case in our formation. This motivates us to refine the framework in [23] so that it can be applied to our setup.

In particular, in Sec. 4 we show that by replacing the full-rank assumption with some mild assumptions on the objective functions, we are not only able to obtain the same convergence and rate of convergence result, but also to simplify the proof of [23]. The m -block ADMM algorithm is general and can be applied in other problem domains. For our case, we further develop a distributed algorithm in Sec. 5, which

is amenable to a parallel implementation in datacenters.

We conduct extensive trace-driven simulations with real-world electricity prices, historical temperature data, and an empirical cooling efficiency model to realistically assess the potential of our approach (Sec. 6). We find that temperature aware workload management is consistently able to deliver a 15%–20% cooling energy reduction and a 5%–20% overall cost reduction for geo-distributed datacenters. The distributed ADMM algorithm converges quickly within 70 iterations, while a dual decomposition approach with subgradient methods fails to converge within 200 iterations. We thus believe our algorithm is practical for large-scale real-world problems.

2. BACKGROUND AND MOTIVATION

Before we make a case for temperature aware workload management, it is necessary to introduce some background of datacenter cooling, and empirically assess the geographical diversity of temperature.

2.1 Datacenter Cooling

Datacenter cooling is provided by the computer room air conditioners (CRACs) placed on the raised floor of the facility. Hot air exhausted from server racks travels through a cooling coil in the CRACs. Heat is often extracted by chilled water in the cooling coil, and the returned hot water is cooled through mechanical refrigeration cycles in an outside chiller plant continuously. The compressor of a chiller consumes a massive amount of energy, and accounts for the majority of the overall cooling cost [40]. The result is an energy-gobbling cooling system that typically consumes a significant portion ($\sim 30\%$) of the total datacenter power [40].

2.2 Outside Air Cooling

To improve energy efficiency, various so-called free cooling technologies that operate without mechanical chillers have recently been adopted. In this paper, we focus on a more economically viable technology called *outside air cooling*. It uses an air-side economizer to direct cold outside air into the datacenter to cool down servers. The hot exhaust air is simply rejected out instead of being cooled and recirculated. The advantage of outside air cooling can be significant: Intel ran a 10-month experiment using 900 blade servers, and reported that 67% of the cooling energy can be saved with only slightly increased hardware failure rates [24]. Companies like Google [1], Facebook [2], and HP [3] have been operating their datacenters with up to 100% outside air cooling, which brings million dollars of savings annually.

The energy efficiency of outside air cooling heavily depends on ambient temperature among other factors. When temperature is lower, less air is needed for heat exchange, and the air handler fan speed can be reduced to save energy. Thus, a CRAC with an air-side economizer usually operates in three modes. When ambient temperature is high, outside air cooling cannot be used, and the CRAC falls back to me-

chanical cooling with chillers. When temperature falls below a certain threshold, outside air cooling is utilized to provide partial or entire cooling capacity. When temperature is too low, outside air is mixed with exhaust air to maintain a suitable supply air temperature. In this mode, CRAC energy efficiency cannot be further improved since fans need to operate at a minimum speed to maintain airflow. Table 1 shows the empirical COP¹ and partial PUE (pPUE)² data of a state-of-the-art CRAC with an air-side economizer. Clearly, as the outdoor temperature drops, the CRAC switches the operating mode to use more outside air cooling. As a result the COP improves six-fold from 3.3 to 19.5, and the pPUE decreases dramatically from 1.30 to 1.05. Due to the sheer amount of energy a datacenter draws, the numbers imply huge monetary savings for the energy bill.

Outdoor ambient	Cooling mode	COP	pPUE
35°C(90°F)	Mechanical	3.3	1.30
21.1°C(70°F)	Mechanical	4.7	1.21
15.6°C(60°F)	Mixed	5.9	1.17
10°C(50°F)	Outside air	10.4	1.1
-3.9°C(25°F)	Outside air	19.5	1.05

Table 1: Efficiency of Emerson’s DSE™ cooling system with an EconoPhase air-side economizer [14]. Return air is set at 29.4°C(85°F).

With the increasing use of outside air cooling, this finding motivates our proposal to make workload management temperature aware. Intuitively, datacenters at colder and thus more energy efficient locations should be better utilized to reduce the overall energy consumption and cost simultaneously. Our idea also applies to datacenters using mechanical cooling, because contrary to previous work’s assumption [28], as shown in Table 1, the chiller energy efficiency also depends on outside temperature, albeit milder.

2.3 An Empirical Climate Study

Our idea hinges upon a key assumption: Temperatures are diverse and not well correlated at different locations. In this section, we make our case concrete by supporting it with an empirical analysis of historical climate data.

We use Google’s datacenter locations for our study, as they represent a global production infrastructure and the location information is publicly available [4]. Google has 6 datacenters in the U.S., 1 in South America, 3 in Europe, and 3 in Asia. We acquire historical temperature data from various data repositories of the National Climate Data Center [6] for all 13 locations, covering the entire one-year period of 2011.

It is useful to first understand the climate profiles at individual locations. Figure 1 plots the daily average temperatures for three select locations in North America, Europe,

¹COP, coefficient of performance, is defined for a cooling device as the ratio between cooling capacity and power.

²pPUE is defined as the sum of cooling capacity and cooling power divided by cooling capacity. Nearly all the power delivered to servers translates to heat, which matches the CRAC cooling capacity.

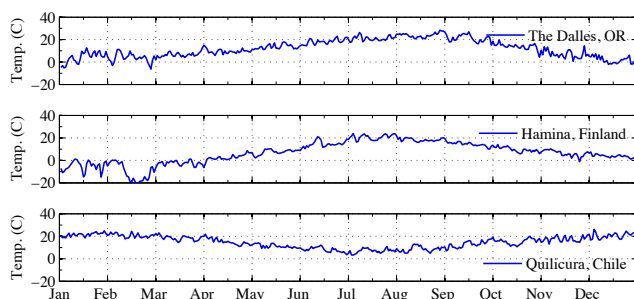


Figure 1: Daily average temperature at three Google datacenter locations. Data from the Global Daily Weather Data of the National Climate Data Center (NCDC) [6]. Time is in UTC.

and South America, respectively. Geographical diversity exists despite the clear seasonal pattern shared among all locations. For example, Finland appears to be especially favorable for cooling during winter months. Diversity is more salient for locations in different hemispheres (e.g. Chile). We also observe a significant amount of day-to-day volatility, suggesting that the availability and capability of outside air cooling constantly varies across regions, and there is no single location that is always cooling efficient.

We then examine short-term temperature volatility. As shown in Figure 2, the hourly variations are more dramatic and highly correlated with time-of-day, which is intuitive to understand. Further, the highs and lows do not occur at the same time for different regions due to time differences.

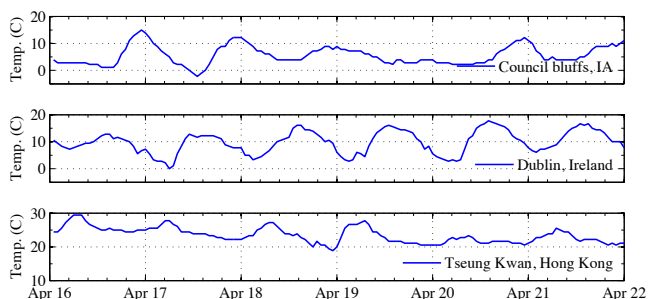


Figure 2: Hourly temperature variations at three Google datacenter locations. Data from the Hourly Global Surface Data of NCDC [6]. Time is in UTC.

Our approach would fail if hourly temperatures are well correlated at different locations. However, we find that this is not the case for datacenters that are usually far apart from each other. The pairwise temperature correlation coefficients for all 13 locations are mostly in between 0.6 and -0.6. Due to space limit, details are omitted and can be found in Sec. 2.3 of our technical report [39].

The analysis above reveals that for globally deployed datacenters, local temperature at individual locations exhibits both time and geographical diversity. Therefore, a carefully designed workload management scheme is critically needed,

in order to dynamically adjust datacenter operations to the ambient conditions, and to save the overall energy costs.

3. MODEL

In this section, we introduce our model first and then formulate the temperature aware workload management problem of joint request routing and capacity allocation.

3.1 System Model

We consider a discrete time model where the length of a time slot matches the time scale at which request routing and capacity allocation decisions are made, *e.g.*, hourly. The joint optimization is periodically solved at each time slot. We therefore focus only on a single time slot.

We consider a provider that runs a set of datacenters \mathcal{J} in distinct geographical regions. Each datacenter $j \in \mathcal{J}$ has a fixed capacity C_j in terms of the number of servers. To model datacenter operating costs, we consider both the *energy cost* and *utility loss* of request routing and capacity allocation, which are detailed below.

3.2 Energy Cost and Cooling Efficiency

We focus on servers and cooling system in our energy cost model. Other energy consumers, such as network switches, power distribution systems, etc., have constant power draw independent of workloads [15] and are not relevant.

For servers, we adopt the empirical model from [15] that calculates the individual server power consumption as an affine function of CPU utilization, $P_{\text{idle}} + (P_{\text{peak}} - P_{\text{idle}})u$. P_{idle} is the server power when idle, P_{peak} is the server power when fully utilized, and u is the CPU load. This model is especially accurate for calculating the aggregated power of a large number of servers [15]. Thus, assuming workloads are perfectly dispatched and servers have a uniform utilization as a result, the server power of datacenter j can be modeled as $C_j P_{\text{idle}} + (P_{\text{peak}} - P_{\text{idle}})W_j$, where W denotes the total workload in terms of the number of servers required.

For the cooling system, we take an empirical approach based on production CRACs to model its energy consumption. We choose not to rely on simplifying models for the individual components of a CRAC and their interactions [40], because of the difficulty involved in and the inaccuracy resulted from the process, especially for hybrid CRACs with both outside air and mechanical cooling. Therefore, we study CRACs as a black box, with outside temperature as the input, and its overall energy efficiency as the output.

Specifically, we use partial PUE (pPUE) to measure the CRAC energy efficiency. As in Sec. 2.2, pPUE is defined as

$$\text{pPUE} = \frac{\text{Server power} + \text{Cooling power}}{\text{Server power}}.$$

A smaller value indicates a more energy efficient system. We apply regression techniques to the empirical pPUE data of the Emerson CRAC [14] introduced in Table 1. We find that the best fitting model describes pPUE as a quadratic function of the outside temperature as plotted below.

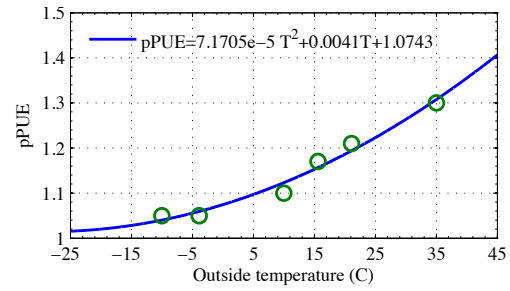


Figure 3: Model fitting of pPUE as a function of the outside temperature T for Emerson’s DSE™ CRAC [14]. Small circles denote empirical data points.

The model can be calibrated given more data from measurements. For the purpose of this paper, our approach yields a tractable model that captures the overall CRAC efficiency for the entire spectrum of its operating modes. Our model is also useful for future studies on datacenter cooling energy.

Given the outside temperature T_j , the total datacenter energy as a function of the workload W_j can be expressed as

$$E_j(W_j) = (C_j P_{\text{idle}} + (P_{\text{peak}} - P_{\text{idle}})W_j) \cdot \text{pPUE}(T_j). \quad (1)$$

Here we implicitly assume that T_j is known a priori and do not include it as the function variable. This is valid since short-term weather forecast is fairly accurate and accessible.

A datacenter’s electricity price is denoted as P_j . The price may additionally incorporate the environmental cost of generating electricity [17], which we do not consider here. In reality, electricity can be purchased from local day-ahead or hour-ahead forward markets at a pre-determined price [34]. Thus, we assume that P_j is known a priori and remains fixed for the duration of a time slot. The total energy cost, including server and cooling power, is simply $P_j E_j(W_j)$.

3.3 Utility Loss

Request routing. The concept of utility loss captures the lost revenue due to the user-perceived latency for request routing decisions. Latency is arguably the most important performance metric for most interactive services. A small increase in the user-perceived latency can cause substantial revenue loss for the provider [25]. We focus on the end-to-end propagation latency, which largely accounts for the user-perceived latency compared to other factors such as request processing times at datacenters [31]. The provider obtains the propagation latency L_{ij} between user i and datacenter j through active measurements [30] or other means.

We use α_{ij} to denote the volume of requests routed to datacenter j from user $i \in \mathcal{I}$, and D_i to denote the demand of each user that can be predicted using machine learning [28, 32]. Here, a user is an aggregated group of customers from a common geographical region, which may be identified by a unique IP prefix. The lost revenue from user i then depends on the *average* propagation latency $\sum_j \alpha_{ij} L_{ij} / D_i$ through a generic delay utility loss function U_i . U_i can take various forms depending on the interactive service. Our algorithm

and proof work for general utility loss functions as long as U_i is increasing, differentiable, and convex.

As a case study, here we use a quadratic function to model user's increased tendency to leave the service with increased latency.

$$U_i(\alpha_i) = qD_i \left(\sum_{j \in \mathcal{J}} \alpha_{ij} L_{ij} / D_i \right)^2, \quad (2)$$

where q is the delay price that translates latency to monetary terms, and $\alpha_i = (\alpha_{i1}, \dots, \alpha_{i|\mathcal{J}|})^T$. Utility loss is clearly zero when latency is zero between user and datacenter.

Capacity allocation. We denote the utility loss of allocating β_j servers for batch workloads as a differentiable, decreasing, and convex function $V_j(\beta_j)$, since allocating more resources increases the performance of batch jobs. Unlike interactive services, batch jobs are delay tolerant and resource elastic. Utility functions such as the log function are often used to capture such elasticity. However, utility functions model the benefit of resource allocation. To model the utility loss of resource allocation, since the loss is zero when the capacity is fully allocated to batch jobs, an intuitive definition can be of the following form:

$$V_j(\beta_j) = r(\log C_j - \log \beta_j), \quad (3)$$

where r is the utility price that converts the loss to monetary terms. (3) captures the intuition that increasing resources results in a decreasing marginal reduction of utility loss.

3.4 Problem Formulation

We now formulate the temperature aware workload management problem. For a given request routing decision α , the total cost associated with interactive workloads can be written as

$$\sum_{j \in \mathcal{J}} E_j \left(\sum_{i \in \mathcal{I}} \alpha_{ij} \right) P_j + \sum_{i \in \mathcal{I}} U_i(\alpha_i). \quad (4)$$

For a given capacity allocation decision β , the total cost associated with batch workloads is:

$$\sum_{j \in \mathcal{J}} E_j(\beta_j) P_j + \sum_{j \in \mathcal{J}} V_j(\beta_j). \quad (5)$$

Putting everything together, the optimization can be formulated as:

$$\text{minimize} \quad (4) + (5) \quad (6)$$

$$\text{subject to:} \quad \forall i : \sum_{j \in \mathcal{J}} \alpha_{ij} = D_i, \quad (7)$$

$$\forall j : \sum_{i \in \mathcal{I}} \alpha_{ij} \leq C_j - \beta_j, \quad (8)$$

$$\alpha, \beta \succeq 0, \quad (9)$$

$$\text{variables:} \quad \alpha \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{J}|}, \beta \in \mathbb{R}^{|\mathcal{J}|}.$$

(6) is the objective function that jointly considers the cost of request routing and capacity allocation. (7) is the workload

conservation constraint to ensure the user demand is satisfied. (8) is the datacenter capacity constraint, and (9) is the nonnegativity constraint.

3.5 Transforming to the ADMM Form

Problem (6) is a large-scale convex optimization problem. The number of users, i.e., unique IP prefixes, is typically $O(10^5)$ – $O(10^6)$ for production systems. Hence, our problem can have tens of millions of variables, and millions of constraints. In such a setting, a distributed algorithm is preferable to fully utilize the computing resources of datacenters. Traditionally, dual decomposition with subgradient methods [9] are often used to develop distributed optimization algorithms. However, they suffer from the curse of step sizes. For the final output to be close to the optimum, we need to strategically pick the step size at each iteration, leading to well-known problems of slow convergence and performance oscillation with large-scale problems.

Alternating direction method of multipliers is a simple yet powerful algorithm that is able to overcome the drawbacks of dual decomposition methods, and is well suited to large-scale distributed convex optimization. Though developed in the 1970s [8], ADMM has recently received renewed interest, and found practical use in many large-scale distributed convex optimization problems in statistics, machine learning, etc. [10]. Before illustrating our new convergence proof and distributed algorithm that extend the classical framework, we first introduce the basics of ADMM, followed by a transformation of (6) to the ADMM form.

ADMM solves problems in the form

$$\begin{aligned} \min \quad & f_1(x_1) + f_2(x_2) \\ \text{s.t.} \quad & A_1 x_1 + A_2 x_2 = b, \\ & x_1 \in C_1, x_2 \in C_2, \end{aligned} \quad (10)$$

with variables $x_\ell \in \mathbb{R}^{n_\ell}$, where $A_\ell \in \mathbb{R}^{p \times n_\ell}$, $b \in \mathbb{R}^p$, f_ℓ 's are convex functions, and C_ℓ 's are non-empty polyhedral sets. Thus, the objective function is *separable* over two sets of variables, which are coupled through an equality constraint.

We can form the augmented Lagrangian [22] by introducing an extra L_2 norm term $\|A_1 x_1 + A_2 x_2 - b\|_2^2$ to the objective:

$$\begin{aligned} L_\rho(x_1, x_2; y) = & f_1(x_1) + f_2(x_2) + y^T (A_1 x_1 + A_2 x_2 - b) \\ & + (\rho/2) \|A_1 x_1 + A_2 x_2 - b\|_2^2. \end{aligned}$$

Here, $\rho > 0$ is the penalty parameter (L_0 is the standard Lagrangian for the problem). The benefits of introducing the penalty term are improved numerical stability and faster convergence in practice [10].

Our formulation (6) has a separable objective function due to the joint nature of the workload management problem. However, the request routing decision α and capacity allocation decision β are coupled by an inequality constraint rather than an equality constraint as in ADMM problems. Thus we

introduce a slack variable $\gamma \in \mathbb{R}^{|\mathcal{J}|}$, and transform (6) to the following

$$\text{minimize} \quad (4) + (5) + I_{\mathbb{R}_+^{|\mathcal{J}|}}(\gamma) \quad (11)$$

subject to: (7), (9),

$$\forall j : \sum_i \alpha_{ij} + \beta_j + \gamma_j = C_j, \quad (12)$$

variables: $\alpha \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{J}|}, \beta \in \mathbb{R}^{|\mathcal{J}|}, \gamma \in \mathbb{R}^{|\mathcal{J}|}$.

Here, $I_{\mathbb{R}_+^{|\mathcal{J}|}}(\gamma)$ is an indicator function defined as

$$I_{\mathbb{R}_+^{|\mathcal{J}|}}(\gamma) = \begin{cases} 0, & \gamma \succeq 0, \\ +\infty, & \text{otherwise.} \end{cases} \quad (13)$$

The new formulation (11) is equivalent to (6), since for any feasible α and β , $\gamma \succeq 0$ holds, and the indicator function in the objective values to zero. Clearly, it is in the ADMM form, with a key difference that it has three sets of variables in the objective function and equality constraint (12). The convergence of the generalized m -block ADMM, where $m \geq 3$, has long remained an open question. Though it seems natural to directly extend the classical 2-block algorithm to the m -block case, such an algorithm may not converge unless some additional back-substitution step is taken [21]. Recently, some progresses have been made by [20, 23] that prove the convergence of m -block ADMM for strongly convex objective functions and the linear convergence of m -block ADMM under a full-column-rank relation matrix. However, the relation matrix in our setup is not full column rank. Thus, we need a new proof for the linear convergence under a general relation matrix, together with a distributed algorithm inspired by the proof.

4. THEORY

This section first introduces a generalized m -block ADMM algorithm inspired by [20, 23]. Then a new convergence proof is presented, which replaces the full column rank assumption with some mild assumptions on the objective function, and further simplifies the proof in [23]. The notations and discussions in this section are made intentionally independent of the other parts of the paper in order to present the proof in a mathematically general way.

4.1 Algorithm

We consider a convex optimization problem in the form

$$\begin{aligned} \min \quad & \sum_{i=1}^m f_i(x_i) \\ \text{s.t.} \quad & \sum_{i=1}^m A_i x_i = b \end{aligned} \quad (14)$$

with variables $x_i \in \mathbb{R}^{n_i}$ ($i = 1, \dots, m$), where $f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}$ ($i = 1, \dots, m$) are closed proper convex functions; $A_i \in \mathbb{R}^{l \times n_i}$ ($i = 1, \dots, m$) are given matrices; and $b \in \mathbb{R}^l$ is a given vector.

We form the augmented Lagrangian

$$\begin{aligned} L_\rho(x_1, \dots, x_m; y) = & \sum_{i=1}^m f_i(x_i) + y^T \left(\sum_{i=1}^m A_i x_i - b \right) \\ & + (\rho/2) \left\| \sum_{i=1}^m A_i x_i - b \right\|_2^2. \end{aligned} \quad (15)$$

As in [23], a generalized ADMM algorithm has the following:

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}} L_\rho(x_1^{k+1}, \dots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \dots, x_m^k; y^k), \quad i = 1, \dots, m,$$

$$y^{k+1} = y^k + \varrho \left(\sum_{i=1}^m A_i x_i^{k+1} - b \right),$$

where $\varrho > 0$ is the step size for the dual update. Note that when $m = 2$ and the step size ϱ equals to the penalty parameter ρ , the above algorithm is reduced to the standard ADMM algorithm presented in [8].

4.2 Assumptions

We present two assumptions on the objective functions, based on which we are able to show the convergence of the generalized m -block ADMM algorithm.

ASSUMPTION 1. *The objective functions f_i ($i = 1, \dots, m$) are strongly convex.*

Note that strong convexity is quite reasonable in engineering practice. This is because a convex function $f(x)$ can be always well-approximated by a strongly convex function $\bar{f}(x)$. For instance, if we choose $\bar{f}(x) = f(x) + \epsilon \|x\|_2^2$ for some sufficiently small $\epsilon > 0$, then $\bar{f}(x)$ is strongly convex.

ASSUMPTION 2. *The gradients ∇f_i ($i = 1, \dots, m$) are Lipschitz continuous.*

Assumption 2 says that, for each i , there exists some constant $\kappa_i > 0$ such that for all $x_1, x_2 \in \mathbb{R}^{n_i}$,

$$\|\nabla f_i(x_1) - \nabla f_i(x_2)\|_2 \leq \kappa_i \|x_1 - x_2\|_2,$$

which is again reasonable in practice, since κ_i can be made sufficiently large.

4.3 Convergence

In this section, we outline the proof for the convergence of the generalized ADMM algorithm. The detailed proof can be found in Sec. 4.3 of our technical report [39].

For convenience, we write

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, \quad f(x) = \sum_{i=1}^m f_i(x_i), \quad \text{and} \quad A = [A_1 \ \dots \ A_m].$$

Then the problem (14) can be rewritten as

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

with the optimal value $p^* = \inf\{f(x) \mid Ax = b\}$. Similarly, the augmented Lagrangian can be rewritten as

$$L_\rho(x; y) = f(x) + y^T(Ax - b) + (\rho/2)\|Ax - b\|_2^2,$$

with the associated dual function defined by

$$d(y) = \inf_x L_\rho(x; y)$$

and the optimal value $d^* = \sup\{d(y)\}$.

Now define the primal and dual optimality gaps as

$$\begin{aligned} \Delta_p^k &= L_\rho(x^{k+1}; y^k) - d(y^k), \\ \Delta_d^k &= d^* - d(y^k), \end{aligned}$$

respectively. Clearly, we have $\Delta_p^k \geq 0$ and $\Delta_d^k \geq 0$. Define

$$V^k = \Delta_p^k + \Delta_d^k.$$

We will see that V^k is a *Lyapunov function* for the algorithm, i.e., a nonnegative quantity that decreases in each iteration.

Our proof relies on three technical lemmas.

LEMMA 1. *There exists a constant $\vartheta > 0$ such that*

$$V^k \leq V^{k-1} - \varrho\|A\bar{x}^{k+1} - b\|_2^2 - \vartheta\|x^{k+1} - x^k\|_2^2, \quad (16)$$

in each iteration, where $\bar{x}^{k+1} = \operatorname{argmin}_x L_\rho(x; y^k)$.

PROOF. See Appendix C in the technical report [39]. \square

LEMMA 2. *For any given $\delta > 0$, there exists a constant $\tau > 0$ (depending on δ) such that for any (x, y) satisfying $\|x\| + \|y\| \leq 2\delta$, the following inequality holds*

$$\|x - \bar{x}(y)\| \leq \tau\|\nabla_x L_\rho(x; y)\|, \quad (17)$$

where $\bar{x}(y) = \operatorname{argmin}_x L_\rho(x; y)$.

PROOF. See Appendix B in the technical report [39]. \square

LEMMA 3. *There exists a constant $\eta > 0$ such that*

$$\|\nabla_x L_\rho(x^k; y^k)\|_2 \leq \eta\|x^k - x^{k+1}\|_2. \quad (18)$$

PROOF. See Appendix A in the technical report [39]. \square

By Lemma 1, we have

$$\sum_{k=0}^{\infty} (\varrho\|A\bar{x}^{k+1} - b\|_2^2 + \vartheta\|x^{k+1} - x^k\|_2^2) \leq V^0.$$

Hence, $\|A\bar{x}^{k+1} - b\|_2^2 \rightarrow 0$ and $\|x^{k+1} - x^k\|_2^2 \rightarrow 0$, as $k \rightarrow \infty$. Suppose that the level set of $\Delta_p + \Delta_d$ is bounded. Then by the Bolzano-Weierstrass theorem, the sequence $\{x^k, y^k\}$ has a convergent subsequence, i.e.,

$$\lim_{k \in \mathcal{R}, k \rightarrow \infty} (x^k, y^k) = (\tilde{x}, \tilde{y}),$$

for some subsequence \mathcal{R} , where (\tilde{x}, \tilde{y}) denotes the limit point. By using Lemma 2 and Lemma 3, we can show that the limit point (\tilde{x}, \tilde{y}) is an optimal primal-dual solution. Hence,

$$\lim_{k \in \mathcal{R}, k \rightarrow \infty} V^k = \lim_{k \in \mathcal{R}, k \rightarrow \infty} \Delta_p^k + \Delta_d^k = 0.$$

Since V^k decreases in each iteration, the convergence of a subsequence of V^k implies the convergence of V^k , and we have

$$\lim_{k \rightarrow \infty} \Delta_p^k + \Delta_d^k = 0.$$

This further implies that both Δ_p^k and Δ_d^k converge to 0.

To sum up, we have the following convergence theorem for our generalized ADMM algorithm.

THEOREM 1. *Suppose that Assumptions 1 and 2 hold and that the level set of $\Delta_p + \Delta_d$ is bounded. Then both the primal gap Δ_p^k and the dual gap Δ_d^k converge to 0.*

Due to space limit, the rate of convergence is omitted and can be found in Sec. 4.3 of [39].

5. A DISTRIBUTED ALGORITHM

We now develop a distributed solution algorithm based on the generalized ADMM algorithm in Sec. 4.1. Directly applying the algorithm to our problem (11) will lead to a centralized algorithm. The reason is that when the augmented Lagrangian is minimized over α , the penalty term $\sum_j \left(\sum_i \alpha_{ij} + \beta_j + \gamma_j - C_j \right)^2$ couples α_{ij} 's across i , and the utility loss $\sum_i U_i(\alpha_i)$ couples α_{ij} 's across j . The joint optimization of utility loss and the quadratic penalty is particularly difficult to solve, especially when the number of users is large, since $U_i(\alpha_i)$ can take any general form. If they can be separated, then we will have a distributed algorithm where each $U_i(\alpha_i)$ is optimized in parallel, and the quadratic penalty term is optimized efficiently with existing methods.

Towards this end, we introduce a new set of auxiliary variables $a_{ij} = \alpha_{ij}$, and re-formulate the problem (11):

$$\text{minimize} \quad \sum_j E_j \left(\sum_i a_{ij} \right) P_j + \sum_i U_i(\alpha_i) + (5) + I_{\mathbb{R}^{|\mathcal{J}|}}(\gamma)$$

subject to: (7), (9),

$$\forall j : \sum_i a_{ij} + \beta_j + \gamma_j = C_j,$$

$$\forall i, j : a_{ij} = \alpha_{ij},$$

$$\text{variables:} \quad a, \alpha \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{J}|}, \beta, \gamma \in \mathbb{R}^{|\mathcal{J}|}. \quad (19)$$

This is a 4-block ADMM problem, where a_{ij} replaces α_{ij} in the objective function and constraint (12) when the coupling happens across users i . This is the key step that enables the decomposition of the α -minimization problem. The augmented Lagrangian can then be readily obtained from (15). By omitting the irrelevant terms, we can see that at each iteration $k + 1$, the α -minimization problem is

$$\begin{aligned} \text{min} \quad & \sum_i U_i(\alpha_i) - \sum_j \sum_i \left(\varphi_{ij} \alpha_{ij} - \frac{\rho}{2} (\alpha_{ij}^2 - 2\alpha_{ij} a_{ij}^k) \right) \\ \text{s.t.} \quad & \forall i : \sum_j \alpha_{ij} = D_i, \alpha_i \succeq 0, \end{aligned} \quad (20)$$

where φ_{ij} is the dual variable for the equality constraint $a_{ij} = \alpha_{ij}$. This is clearly decomposable over i into $|\mathcal{I}|$ per-user sub-problems since the objective function and constraint are separable over i . The per-user sub-problem is of a much smaller scale with only $|\mathcal{J}|$ variables and $|\mathcal{J}| + 1$ constraints, and is easy to solve even though it is a non-linear problem for a general U_i .

Some may now wonder if the auxiliary variable a is hard to solve for. As it turns out, the a -minimization problem is decomposable over j into $|\mathcal{J}|$ per-datacenter sub-problems. Moreover, each per-datacenter sub-problem is a quadratic program. Though it is large-scale, it can be transformed into a second-order cone program and solved efficiently. More details can be found in Sec. 5 in the technical report [39].

β - and γ -minimization steps are clearly decomposable over j . The entire procedure is summarized below.

Distributed 4-block ADMM. Initialize $a, \alpha, \beta, \gamma, \lambda, \varphi$ to 0. For $k = 0, 1, \dots$, repeat

1. **α -minimization:** Each user solves the following sub-problem for α_i^{k+1} :

$$\begin{aligned} \min \quad & U_i(\alpha_i) - \sum_j \left(\varphi_{ij} \alpha_{ij} - \frac{\rho}{2} (\alpha_{ij}^2 - 2\alpha_{ij} a_{ij}^k) \right) \\ \text{s.t.} \quad & \sum_j \alpha_{ij} = D_i, \alpha_i \succeq 0. \end{aligned} \quad (21)$$

2. **a -minimization:** Each datacenter solves the following sub-problem for $a_j^{k+1} = (a_{1j}^{k+1}, \dots, a_{|\mathcal{I}|j}^{k+1})^T$:

$$\begin{aligned} \min \quad & E_j \left(\sum_i a_{ij} \right) P_j + \sum_i a_{ij} (\lambda_j^k + \varphi_{ij}^k) + \frac{\rho}{2} \left(\sum_i a_{ij} \right)^2 \\ & + \rho \left(\sum_i a_{ij} (\beta_j^k + \gamma_j^k - C_j + 0.5a_{ij} - \alpha_{ij}^{k+1}) \right) \\ \text{s.t.} \quad & a_j \succeq 0. \end{aligned} \quad (22)$$

3. **β -minimization:** Each datacenter solves the following sub-problem for β_j^{k+1} :

$$\begin{aligned} \min \quad & E_j(\beta_j) P_j + V_j(\beta_j) + \lambda_j^k \beta_j \\ & + \frac{\rho}{2} \left(\sum_i a_{ij}^{k+1} + \beta_j + \gamma_j^k - C_j \right)^2 \\ \text{s.t.} \quad & \beta_j \geq 0. \end{aligned}$$

4. **γ -minimization:** Each datacenter solves:

$$\gamma_j^{k+1} = \max \left\{ 0, C_j - \frac{\lambda_j}{\rho} - \sum_i a_{ij}^{k+1} - \beta_j^{k+1} \right\}, \forall j.$$

5. **Dual update:** Each datacenter updates λ_j for the capacity constraint (8):

$$\lambda_j^{k+1} = \lambda_j^k + \varrho \left(\sum_i a_{ij}^{k+1} + \beta_j^{k+1} + \gamma_j^{k+1} - C_j \right).$$

Each user updates φ_{ij} for the equality constraint $a_{ij} = \alpha_{ij}$:

$$\varphi_{ij}^{k+1} = \varphi_{ij}^k + \varrho (a_{ij}^{k+1} - \alpha_{ij}^{k+1}), \forall j.$$

The distributed nature of our algorithm allows for an efficient parallel implementation in datacenters with a large number of servers. The per-user sub-problem (21) can be solved in parallel on each server. Since (21) is a small-scale convex optimization as discussed above, the complexity is low. A multi-threaded implementation can further speed up the algorithm with multi-core hardware. The penalty parameter ρ and utility loss function U_i can be configured at each server before the algorithm runs. Step 2 and 3 involve solving $|\mathcal{J}|$ per-datacenter sub-problems respectively, which can also be implemented in parallel with only $|\mathcal{J}|$ servers.

6. EVALUATION

We perform trace-driven simulations to realistically assess the potential of temperature aware workload management.

6.1 Setup

We rely on the Wikipedia request traces [38] to represent the interactive workloads of a cloud service. The dataset we use contains, among other things, 10% of all user requests issued to Wikipedia from the 24-hour period between January 1, 2008 UTC to January 2, 2008 UTC. The workloads are normalized to a number of servers, assuming that each request requires 10% of a server's CPU. The traces reflect the diurnal pattern of real-world interactive workloads. The prediction of workloads can be done accurately as demonstrated by previous work [28, 32], and we do not consider the effect of prediction error here. The optimization is solved hourly.

We consider Google's infrastructure [4] to represent a geo-distributed cloud as discussed in Sec. 2.3. Each datacenter's capacity C_j is uniformly distributed between $[1, 2] \times 10^5$ servers. The empirical CRAC efficiency model developed in Sec. 3.2 is used to derive the total energy consumption of all 13 locations under different temperatures. We use the 2011 annual average day-ahead on peak prices [16] at the local markets as the power prices P_j for the 6 U.S. locations³. For non-U.S. locations, the power price is calculated based on the retail industrial power price available on the local utility company websites with a 50% wholesale discount, which is usually the case in reality [37]. The power prices at each location are shown in Table 2 in the technical report [39]. The servers have peak power $P_{\text{peak}} = 200$ W, and consume 50% power at idle. These numbers represent state-of-the-art datacenter hardware [15, 34].

To calculate the utility loss of interactive workloads, we obtain the latency matrix L from iPlane [30], a system that collects wide-area network statistics from Planetlab vantage points. Since the Wikipedia traces do not contain client side information, we emulate the geographical diversity of user requests by splitting the total interactive workloads among users following a normal distribution. We set the number of

³The U.S. electricity market is consisted of multiple regional markets. Each regional market has several hubs with their own pricing. We thus use the price of the specific hub that each U.S. datacenter locates in.

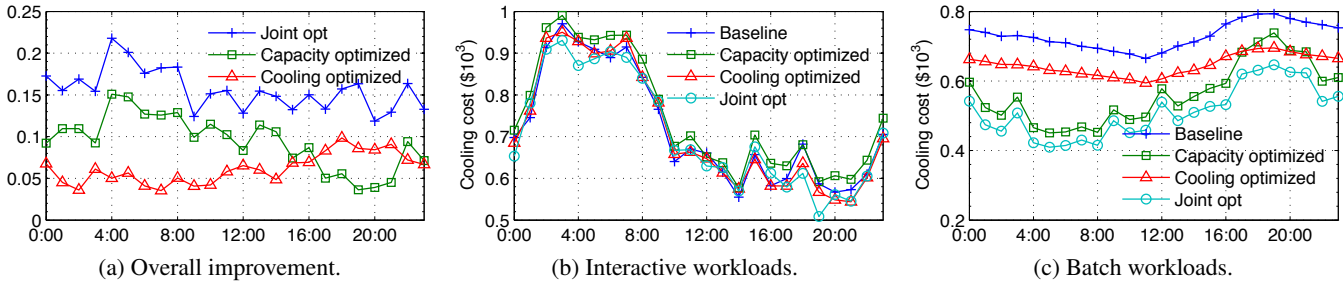


Figure 4: Cooling energy cost savings. Time is in UTC.

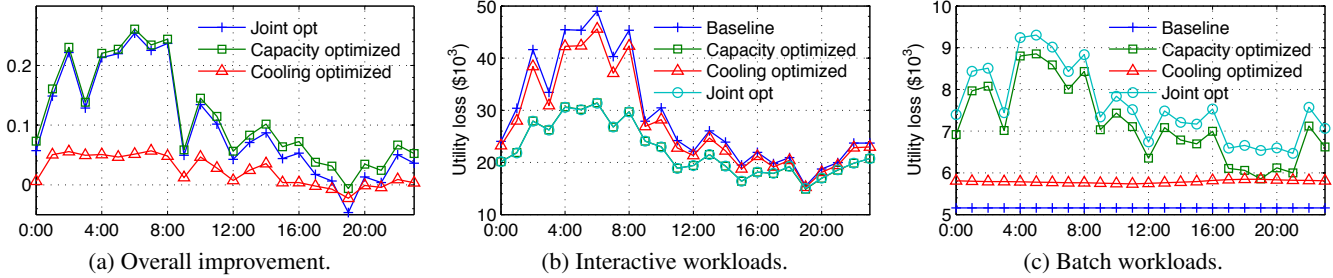


Figure 5: Utility loss reductions. Time is in UTC.

users $|\mathcal{I}| = 10^5$, and choose 10^5 IP prefixes from a RouteViews [5] dump. Note that in our context, each user, i.e. IP prefix, represents many customers accessing the service. We then extract the corresponding round trip times from iPlane logs, which contain traceroutes made to IP addresses from Planetlab nodes. We only use latency measurements from Planetlab nodes that are close to our datacenter locations to resemble the user-datacenter latency. We use utility loss functions defined in (2) and (3). The delay price $q = 4 \times 10^{-6}$, and the utility loss price for batch jobs $r = 500$.

We investigate the performance of temperature aware workload management. We benchmark our ADMM algorithm, referred to as *Joint opt*, against three baseline strategies, which use different amounts of information in managing workloads.

The first benchmark, called *Baseline*, is a temperature agnostic strategy that separately considers capacity allocation and request routing of the workload management problem. It first allocates capacity to batch jobs by minimizing the backend total cost with (5) as the objective. The remaining capacity is used to solve the request routing optimization with (4) as the objective. Only the electricity price diversity is used, and cooling energy is calculated with a constant pPUE of 1.2 that corresponds to an ambient temperature of 20°C for the two cost minimization problems. Though naive, such an approach is widely used in current Internet-scale cloud services. It also allows an implicit comparison with prior work [17, 27, 29, 34, 35].

The second benchmark, called *Capacity Optimized*, improves upon *Baseline* by jointly solving capacity allocation and request routing, but still ignores the cooling energy efficiency diversity. This demonstrates the impact of capacity allocation in datacenter workload management.

The third benchmark, called *Cooling Optimized*, improves upon *Baseline* by exploiting the temperature and cooling efficiency diversity in minimizing cost, but does not adopt joint management of the interactive and batch workloads. This demonstrates the impact of being temperature aware.

We run the four benchmarks above with our 24-hour traces at each day of January 2011, using the empirical hourly temperature data we collected in Sec. 2.3. The distributed ADMM algorithm is used to solve them until convergence is achieved. The figures show the average results over 31 runs.

6.2 Cooling energy savings

The central thesis of this paper is to save datacenter cost through temperature aware workload management that exploits the cooling efficiency diversity with capacity allocation. We examine the effectiveness of our approach by comparing the cooling energy consumption first. Figure 4 shows the results.

In particular, Figure 4a shows that overall, *Joint opt* saves 15%–20% cooling energy compared to *Baseline*. A breakdown of the saving shown in the same figure reveals that dynamic capacity allocation provides 10%–15% saving, and cooling efficiency diversity provides 5%–10% saving, respectively. Note that the cost saving is achieved with cutting-edge CRACs whose efficiency is already substantially improved with outside air cooling capability. The results confirm that our temperature aware workload management is able to further optimize the cooling efficiency and cost of geo-distributed datacenters.

Figure 4b and 4c show a detailed breakdown of cooling energy cost. Cooling cost attributed to interactive workloads, as in Figure 4b, exhibits a diurnal pattern and peaks between 2:00 and 8:00 UTC (21:00 to 3:00 EST, 18:00 to 0:00 PST), implying that most of the Wikipedia traffic origi-

nates from the U.S. The four strategies perform fairly closely, while *Baseline* and *Capacity optimized* consistently incur more cooling energy cost due to their cooling agnostic nature that underestimates the overall energy cost.

Cooling cost attributed to batch workloads is shown in Figure 4c. *Baseline* incurs the highest cost since it underestimates the energy cost, and runs more batch workloads than necessary. *Cooling optimized* improves *Baseline* by taking into account cooling efficiency diversity and reducing batch workloads as a result. Both strategies fail to exploit the trade-off with interactive workloads. Thus their cooling cost closely follows the daily temperature trend in that it gradually decreases from 0:00 to 12:00 UTC (19:00 to 7:00 EST) and then slowly increases from 12:00 to 20:00 UTC (7:00 to 15:00 EST). *Capacity optimized* adjusts capacity allocation with request routing, and further reduces batch workloads in order to allocate more resources for interactive workloads. *Joint opt* combines temperature aware cooling optimization with holistic workload management, and has the lowest cooling cost with least batch workloads. Though this increases the back-end utility loss, the overall effect is a net reduction of total cost since interactive workloads enjoy lower latency as will be observed soon.

6.3 Utility loss reductions

The other component of datacenter cost is utility loss. From Figure 5a, the relative reduction follows the interactive workloads and also has a visible diurnal pattern. *Joint opt* and *Capacity optimized* provide the most significant utility loss reductions from 5% to 25%, while *Cooling optimized* provides a modest 5% reduction compared to *Baseline*. To study the reasons for the varying degrees of reductions, Figure 5b and 5c show the respective utility loss of interactive and batch workloads. We observe that interactive workloads incur most of the utility loss, reflecting its importance compared to batch workloads. *Baseline* and *Cooling optimized* have much larger utility loss from interactive workloads as shown in Figure 5b, because of the separate management of two workloads. The average latency performances under these two strategies are also worse as can be seen in Figure 7 of our technical report [39].

On the other hand, *Capacity optimized* and *Joint opt* outperform the two by allocating more capacity to interactive workloads at cost-efficient locations while reducing batch workloads (recall Figure 4c). This is especially effective during peak hours as shown in Figure 5b. *Capacity optimized* and *Joint opt* do have larger utility loss from batch workloads as seen in Figure 5c. However since interactive workloads attribute to the majority of the provider’s utility and revenue, the overall effect of joint workload management is positive.

6.4 Sensitivity to seasonal changes

One natural question is, since the results above are obtained in winter times (January), would the benefits be less significant during summer times when cooling is more ex-

pensive? In other words, are the benefits sensitive to the seasonal changes? We thus run our *Joint opt* with *Baseline* at each day of May, which represents typical Spring/Fall weather, and August, which represents typical Summer weather, respectively. Figure 6 shows the average overall cost savings achieved in different seasons. We observe that the cost savings, ranging from 5% to 20%, are consistent and insensitive to seasonal changes. The reason is that our approach depends on: 1) the geographical diversity of temperature and cooling efficiency; 2) the mixed nature of datacenter workloads, both of which exist at all times of the year no matter which cooling method is used. Temperature aware workload management is thus able to offer consistent cost benefits.

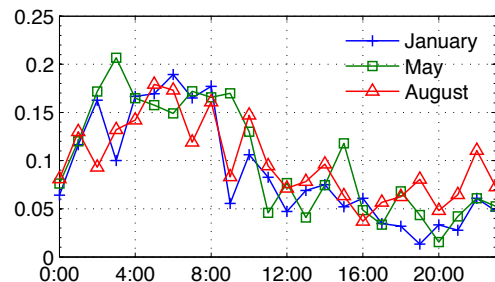


Figure 6: Overall cost saving is insensitive to seasonal changes of the climate.

We also compare the convergence speed of our the distributed ADMM algorithm with the conventional subgradient method. We have found that our algorithm converges within around 60 iterations, while the subgradient method does not converge even after 200 iterations. Our distributed ADMM algorithm is thus better suited to large-scale convex optimization problems. More details can be found in Sec. 6.3 in the technical report [39].

7. CONCLUSION

We propose temperature aware workload management, which explores two key aspects of geo-distributed datacenters that have not been well understood in the past. First, as we show empirically, energy efficiency of cooling systems, especially outside air cooling, varies widely with outside temperature. The geographical diversity of temperature is utilized to reduce cooling energy consumption. Second, the elastic nature of batch workloads is further capitalized by dynamically adjusting capacity allocation along with the widely studied request routing for interactive workloads. We formulate the joint optimization under a general framework with an empirical cooling efficiency model. To solve large-scale problems for production systems, we rely on the ADMM algorithm. We provide a new convergence proof for a generalized m -block ADMM algorithm. We further develop a novel distributed ADMM algorithm for our problem. Extensive simulations highlight that temperature aware workload management saves 15%–20% cooling energy and 5%–20% overall energy cost and the distributed ADMM algorithm is practical to solve large-scale workload management problems with only tens of iterations.

8. REFERENCES

- [1] <http://tinyurl.com/89ros64>.
- [2] <http://tinyurl.com/8ulxfzp>.
- [3] <http://tinyurl.com/bpqv6tl>.
- [4] <https://www.google.com/about/datacenters/inside/locations/>.
- [5] <http://www.routeviews.org>.
- [6] National climate data center (NCDC). <http://www.ncdc.noaa.gov>.
- [7] BASH, C., AND FORMAN, G. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *Proc. USENIX ATC* (2007).
- [8] BERTSEKAS, D. P., AND TSITSIKLIS, J. N. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.
- [9] BOYD, S., AND MUTAPCIC, A. Subgradient methods. Lecture notes of EE364b, Stanford University, Winter Quarter 2006-2007. http://www.stanford.edu/class/ee364b/notes/subgrad_method_notes.pdf.
- [10] BOYD, S., PARIKH, N., CHU, E., PELEATO, B., AND ECKSTEIN, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning* 3, 1 (2010), 1–122.
- [11] CHEN, Y., GMACH, D., HYSER, C., WANG, Z., BASH, C., HOOVER, C., AND SINGHAL, S. Integrated management of application performance, power and cooling in datacenters. In *Proc. NOMS* (2010).
- [12] DENG, N., STEWART, C., GMACH, D., ARLITT, M., AND KELLEY, J. Adaptive green hosting. In *Proc. ACM ICAC* (2012).
- [13] EL-SAYED, N., STEFANOVICI, I., AMVROSIADIS, G., AND HWANG, A. A. Temperature management in data centers: Why some (might) like it hot. In *Proc. ACM Sigmetrics* (2012).
- [14] EMERSON NETWORK POWER. Liebert[®] DSE[™] precision cooling system sales brochure. <http://tinyurl.com/c7e8qxz>, 2012.
- [15] FAN, X., WEBER, W.-D., AND BARROSO, L. A. Power provisioning for a warehouse-sized computer. In *Proc. ACM/IEEE Intl. Symp. Computer Architecture (ISCA)* (2007).
- [16] FEDERAL ENERGY REGULATORY COMMISSION. U.S. electric power markets. <http://www.ferc.gov/market-oversight/mkt-electric/overview.asp>, 2011.
- [17] GAO, P. X., CURTIS, A. R., WONG, B., AND KESHAV, S. It's not easy being green. In *Proc. ACM SIGCOMM* (2012).
- [18] GOIRI, I. N., BEAUCHEA, R., LE, K., NGUYEN, T. D., HAQUE, M. E., GUITART, J., TORRES, J., AND BIANCHINI, R. Greenslot: Scheduling energy consumption in green datacenters. In *Proc. SC* (2011).
- [19] GOIRI, I. N., LE, K., NGUYEN, T. D., GUITART, J., TORRES, J., AND BIANCHINI, R. GreenHadoop: Leveraging green energy in data-processing frameworks. In *Proc. ACM EuroSys* (2012).
- [20] HAN, D., AND YUAN, X. A note on the alternating direction method of multipliers. *J. Optim. Theory Appl.* 155 (2012), 227–238.
- [21] HE, B. S., TAO, M., AND YUAN, X. M. Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM J. Optim.* 22 (2012), 313–340.
- [22] HESTENES, M. R. Multiplier and gradient methods. *Journal of Optimization Theory and Applications* 4, 5 (1969), 303–320.
- [23] HONG, M., AND LUO, Z.-Q. On the linear convergence of the alternating direction method of multipliers, August 2012.
- [24] INTEL INC. Reducing data center cost with an air economizer, August 2008.
- [25] KOHAVI, R., HENNE, R. M., AND SOMMERFIELD, D. Practical guide to controlled experiments on the web: Listen to your customers not to the hippo. In *Proc. ACM SIGKDD* (2007).
- [26] LE, K., BIANCHINI, R., NGUYEN, T. D., BILGIR, O., AND MARTONOSI, M. Capping the brown energy consumption of Internet services at low cost. In *Proc. IGCC* (2010).
- [27] LIN, M., WIERMAN, A., ANDREW, L. L. H., AND THERESKA, E. Dynamic right-sizing for power-proportional data centers. In *Proc. IEEE INFOCOM* (2011).
- [28] LIU, Z., CHEN, Y., BASH, C., WIERMAN, A., GMACH, D., WANG, Z., MARWAH, M., AND HYSER, C. Renewable and cooling aware workload management for sustainable data centers. In *Proc. ACM Sigmetrics* (2012).
- [29] LIU, Z., LIN, M., WIERMAN, A., LOW, S. H., AND ANDREW, L. L. Greening geographical load balancing. In *Proc. ACM Sigmetrics* (2011).
- [30] MADHYASTHA, H. V., ISDAL, T., PIATEK, M., DIXON, C., ANDERSON, T., KRISHNAMURTHY, A., AND VENKATARAMANI, A. iPlane: An information plane for distributed services. In *Proc. USENIX OSDI* (2006).
- [31] NARAYANA, S., JIANG, J. W., REXFORD, J., AND CHIANG, M. To coordinate or not to coordinate? Wide-Area traffic management for data centers. Tech. rep., Princeton University, 2012.
- [32] NIU, D., XU, H., LI, B., AND ZHAO, S. Quality-assured cloud bandwidth auto-scaling for video-on-demand applications. In *Proc. IEEE INFOCOM* (2012).
- [33] PELLEY, S., MEISNER, D., WENISCH, T. F., AND VANGILDER, J. W. Understanding and abstracting

- total data center power. In *Proc. Workshop on Energy Efficient Design (WEED)* (2009).
- [34] QURESHI, A., WEBER, R., BALAKRISHNAN, H., GUTTAG, J., AND MAGGS, B. Cutting the electricity bill for Internet-scale systems. In *Proc. ACM SIGCOMM* (2009).
- [35] RAO, L., LIU, X., XIE, L., AND LIU, W. Minimizing electricity cost: Optimization of distributed Internet data centers in a multi-electricity-market environment. In *Proc. IEEE INFOCOM* (2010).
- [36] REISS, C., TUMANOV, A., GANGER, G. R., KATZ, R. H., AND KOZUCH, M. A. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In *Proc. ACM SoCC* (2012).
- [37] TELEGEOGRAPHY RESEARCH. Global Internet geography executive summary. http://bpastudio.csudh.edu/fac/lpress/471/hout/telegeographygig_execsumm.pdf, 2008.
- [38] URDANETA, G., PIERRE, G., AND VAN STEEN, M. Wikipedia workload analysis for decentralized hosting. *Elsevier Computer Networks* 53, 11 (July 2009), 1830–1845.
- [39] XU, H., FENG, C., AND LI, B. Temperature aware workload management in geo-distributed datacenters. Tech. rep., University of Toronto, <http://iqua.ece.toronto.edu/~henryxu/share/geodc-preprint.pdf>, 2013.
- [40] ZHOU, R., WANG, Z., MCREYNOLDS, A., BASH, C., CHRISTIAN, T., AND SHIH, R. Optimization and control of cooling microgrids for data centers. In *Proc. IEEE ITherm* (2012).