Integrating Cooling Awareness with Thermal Aware Workload Placement for HPC Data Centers [1]

Ayan Banerjee, Tridib Mukherjee, Georgios Varsamopoulos, Sandeep K. S. Gupta *

Impact Lab
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University
Tempe, Arizona, USA

Abstract

High Performance Computing (HPC) data centers are becoming increasingly dense; the associated power-density and energy consumption of their operation is increasing. Up to half of the total energy is attributed to cooling the data center; greening the data center operations to reduce both computing and cooling energy is imperative. To this effect, this paper integrates awareness of the dynamic behavior of the cooling unit with thermal awareness to develop new spatial workload placement algorithms for HPC data centers. The paper first proposes a coordinated *cooling-aware* job placement and cooling management algorithm, *Highest Thermostat Setting* (HTS). HTS is aware of dynamic behavior of the Computer Room Air Conditioner (CRAC) units and places the jobs to reduce the cooling demands from the CRACs. HTS also dynamically updates the CRAC thermostat set point to optimize cooling energy consumption. Further, the *Energy Inefficiency Ratio of SPatial job scheduling* (a.k.a. *job placement*) algorithms, also referred as *SP-EIR*, is analyzed by comparing the total (computing + cooling) energy consumption incurred by the algorithms with the minimum possible energy consumption, while assuming that the job start times are already decided to meet the Service Level Agreements (SLAs). This analysis is performed for two cooling models, constant and dynamic, to show how the common constant cooling model assumption in previous research misses out on opportunities to save energy. Simulation results based on power measurements and job traces from the ASU HPC data center show that: i) HTS has 15% lower SP-EIR compared to LRH, a thermal-aware spatial scheduling algorithm; and ii) in conjunction with FCFS-Backfill, HTS increases the throughput per unit energy by 6.89% and 5.56%, respectively, over LRH and MTDP (an energy-efficient spatial scheduling algorithm with server consolidation).

Key words: cooling aware, spatial job scheduling, energy efficiency, dynamic cooling model, energy efficiency metrics and bounds

* Corresponding author.
Email address: sandeep.gupta@asu.edu (Sandeep K. S. Gupta).
URL: http://impact.asu.edu/ (Sandeep K. S. Gupta).

[1] This work was funded in parts by NSF (CNS#0855277 and CSR#0834797), SFAz and Intel. This paper extends the conference version titled "Cooling-Aware and Thermal-Aware Workload Placement for Green HPC Data Centers" published in IGCC 2010 with: i) comprehensive thermodynamic details, ii) incorporation of different types of CRAC behavior in the scheduling decision making, iii) extensive simulation results, and iv) improved related works

1. Introduction

High Performance Computing (HPC) applications require high computation capabilities, often in the range of teraflops. A major issue in contemporary data centers, hosting such high computation facilities, is the high energy consumption in their operations. Indeed, the data centers' energy consumption amounted to nearly 2% of the total energy budget of the US in 2007 and is expected to reach 4% in 2011 [1]; as such, *greening* the data center operations has been of utmost interest over the years [2–10]. Up to half of this energy can be attributed to cooling the data centers (i.e. *cooling energy*) to keep the operating temperatures within manufacturer specified *redline* temperatures. This paper focuses on a cyber-physical oriented *coordinated job and cooling management* in HPC data centers to reduce the total (i.e. computing and cooling) energy consumption of the data centers.

The cooling energy depends on two factors: i) the *cooling demand*, which is driven by the power distribution and the redline temperature; and ii) the *cooling behavior*, i.e. the behavior of the Computer Room Air Conditioner (CRAC) unit (controlled by varying the thermostat setting), to meet the demand. A major concern in this regard is the possible *recirculation* and intermixing of hot air, generated by the servers running the jobs, with the cold air supplied from the CRAC [7]. Recirculation of hot air depends on the data center layout and can cause hot-spots; thus potentially increasing the cooling demand.

Techniques to reduce the power consumption include: i) thermal-aware workload (job) management in the cyber domain [5–8] that is aware of the heat recirculation; and ii) design layout and expensive physical infrastructure [11] to minimize, or even remove, any potential recirculation. However, cooling energy reduction is not guaranteed by thermal-aware job management unless coordinated with the dynamic management of the CRAC; such management should be **cooling-aware**, i.e. aware of the dynamic cooling behavior. As such, in many existing data centers the cooling is over-provisioned; resulting in a high *Power Usage Efficiency (PUE)* [2] . Further, there is no definite way to evaluate the *energy inefficiencies* of the job management algorithms and their dependencies on the heat recirculation and the cooling behavior of the CRAC. Hence, the cost-benefit trade-off of incorporating better physical and cooling infrastructure to reduce the recirculation and improve cooling is unknown.

This paper focuses on: 1) developing a cooling and thermal aware job placement algorithm for HPC datacenters taking into account the dynamic behavior of the CRAC, 2) theoretically analyzing the energy inefficiency of the spatial scheduling algorithm, when operated under different cooling behaviors, and 3) developing a spatio-temporal job scheduling and placement algorithm that is cognizant of the cooling behavior of the CRAC.

1.1. *Overview of Contributions and Results*

The **contributions** of this paper are summarized as follows.

[2] PUE is the ratio of total power consumption over the computing power consumption to service the jobs in a data center.

(i) *Highest Thermostat Setting (HTS)* algorithm is developed, which performs *cooling-aware* and *thermal-aware* spatial job scheduling, and integrates such scheduling with cooling management (dynamic variation of CRAC thermostat setting to meet the cooling demands). For the ASU HPC data center, HTS has an SP-EIR of 1.013, which is 15% lower than that for Least Recirculated Heat (LRH) algorithm, an energy-efficient and thermal-aware online spatial scheduling algorithm [12]. Simulations were performed to compare the performance per unit of energy consumption of HTS with that of LRH by separately using them with the same temporal scheduling algorithm. The simulation results are based on the layout, equipment profile, and actual job traces of the ASU HPC data center.

(ii) An *energy inefficiency* analysis of spatial job scheduling (i.e. job placement) algorithms is performed by taking into account the heat recirculation and cooling behavior. First, a metric, *Energy Inefficiency Ratio of SPatial scheduling (SP-EIR)*, is defined to compare the total energy consumption incurred by the algorithm with respect to the minimum possible energy consumption, while assuming that the job start times are already decided to meet the Service Level Agreements (SLAs). It is shown that the SP-EIR of any placement algorithm decreases (i.e. the energy consumption is reduced) with a decrease in the heat recirculation (i.e. with better data center design).

(iii) *Spatio-temporal* scheduling, i.e. deciding on *when and in which server* to execute a job, can be performed by integrating HTS with different temporal scheduling algorithms. This paper uses two different temporal scheduling algorithms: i) *First Come First Serve with backfilling* (FCFS-Backfill), the most commonly used scheduling algorithm in contemporary HPC data centers [13]; and ii) *Earliest Deadline First* (EDF), which has been used previously for energy efficiency while meeting the users' perception of job turn-around time (i.e. the deadline) [12]. Further, to ensure reduction in the computing energy, HTS can be augmented with power control techniques.

SP-EIR can be used by the data center designers to quantify the benefit of reducing the heat recirculation with respect to the cost of the infrastructure. Further, the increase in the CRAC thermostat set temperatures can decrease the SP-EIR. Cooling-aware job placement and coordinating with the dynamic CRAC management, as in HTS, can increase CRAC thermostat set temperatures; hence reducing SP-EIR and consequently the energy consumption. Indeed, simulation results show that HTS, when combined with EDF (i.e. EDF-HTS), can achieve up to 15% energy savings over EDF-LRH [12], an energy-efficient and thermal-aware online spatio-temporal scheduling algorithm if there is no power control in the servers. Even with power control, EDF-HTS achieves 9% energy savings over EDF-MTDP, i.e. EDF used in conjunction with Minimum Total Data center Power (MTDP) spatial scheduling, which performs thermal-aware server consolidation [7]. Further, HTS, when combined with FCFS-Backfill (i.e. FCFS-Backfill-HTS), can achieve up to 5.56% higher throughput per unit of energy consumption (measured in terms of number of jobs per second per Joule) over FCFS-Backfill-MTDP. Such improvement is significant when scaled to annual cost savings and

performance benefits of a data center.

1.2. *Overview of Approach*

Two different models of CRAC unit operation are considered: 1) *constant model*, where the CRAC acts as a source of cool air at a constant temperature, and 2) *dynamic model*, where the CRAC has multiple modes of operation and in each mode extracts heat at a definite rate from the data center. The constant model of the CRAC, though unrealistic, has been a common assumption in previous literature [5–8, 10, 12]. However, a recent empirical study on the cold supply air temperature from the CRAC and the hot input air temperature to the CRAC [14] suggests the presence of different modes in the CRAC operation. The temperature of the cold supply air from the CRAC depends on the CRAC's mode of operation and the thermostat set temperatures. The higher the thermostat set temperature the higher is the supply air temperature; thus producing less cooling and hence consumes less cooling energy. In this paper, we show the algorithm design and analysis for both CRAC models. An interesting fact in this regard is that the LRH spatial scheduling algorithm developed for constant cooling model [12] is a special case of the HTS algorithm. Hence, the more generic case of dynamic cooling is first considered.

In this regard, it is observed that, *in an over-cooled data center the energy consumption can be reduced by increasing the CRAC thermostat set temperatures.* An upper bound on the thermostat set temperatures is obtained from the required supply air temperature from the CRAC to ensure that all the servers can operate within their respective redline temperatures. HTS *is designed to place jobs in servers such that heat recirculation effect is reduced and thermostat set temperature requirements are increased.*

The spatial job scheduling is augmented with dynamic updates of the CRAC thermostat set temperatures. Figure 1 shows an example of cooling-aware spatial job scheduling in HTS. As shown in the figure, there are two jobs requested to be placed in two of the three available server. The leftmost server is in the portion of the data center with high heat recirculation. As such, placing a job in the leftmost server may lead to a hot spot. Traditional approach of non-thermal-aware spatial job scheduling can place a job in the leftmost server (Figure 1a); thus leading to hot spots and requiring low CRAC thermostat set temperature (18°C in the figure).

Even in a dynamic cooling environment CRAC thermostat is usually statically set to a low value, to counter worst case situations of hot spots (Figure 1a); thus cooling is over-provisioned most of the time. As shown in Figure 1b, cooling-aware job management has to avoid placing jobs in the leftmost server; and thus the required thermostat temperatures are increased. Further, coordination with the cooling management can enable dynamically setting the thermostat to high values.

Figure 2 conceptualizes the coordinated job and cooling management in data centers. The submitted workload is provided to the *job management module* where it first gets temporally scheduled which ensures that the jobs get the requested types of servers and are destined to meet the deadline. The temporally scheduled jobs are then dispatched

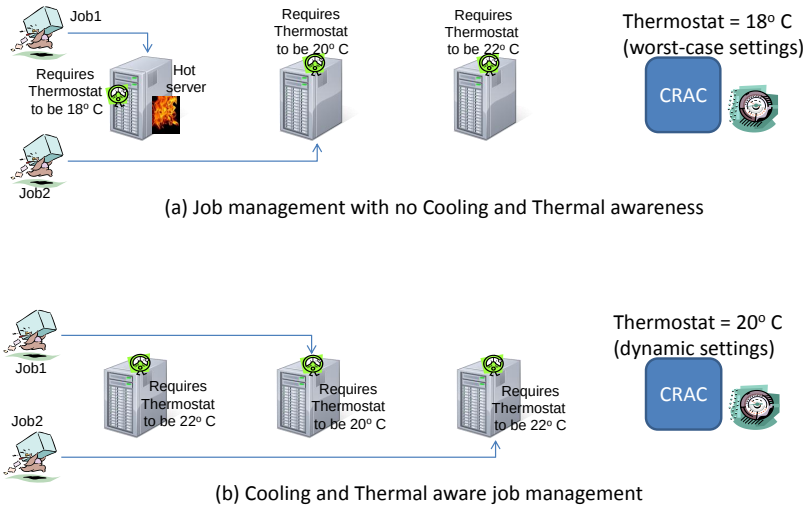

Fig. 1. Example of cooling-aware job management in HTS.

among the requested servers in the data center by the *spatial scheduling module*. The spatial scheduling attempts to place the jobs in servers such that: 1) the CRAC thermostat can be dynamically set, and 2) the total data center energy (computing + cooling) consumption is reduced. The combination of the spatial scheduling with dynamic thermostat setting is called the Coordinated Job and Cooling Management; an example of which is the HTS algorithm. We evaluate HTS from the following two perspectives:

(i) *Spatial perspective*: A generic expression for SP-EIR of any spatial job scheduling (i.e. job placement) technique is obtained with respect to the optimal energy-efficient approach that minimizes the energy consumption. The actual values of the SP-EIR depends on the spatial scheduling algorithm and the data center in question.

(ii) *Spatio-temporal perspective*: The SP-EIR does not evaluate the overall spatio-temporal performance. Simulations were performed using empirical data from the ASU HPC data center to show the benefits of being aware of the CRAC dynamic behavior in HTS when combined with FCFS-Backfill and EDF temporal scheduling algorithms.

1.3. *Paper Organization*

The rest of the paper is organized as follows. Section 2 describes the system model of the data center. The cooling-aware job placement problem is defined in Section 3 followed by the HTS algorithm description in Section 4. Section 5 defines the SP-EIR metric for spatial job scheduling and describes its dependency on the heat recirculation and CRAC thermostat setting. Section 6 presents the simulation based evaluation of HTS from both spatial and spatio-temporal

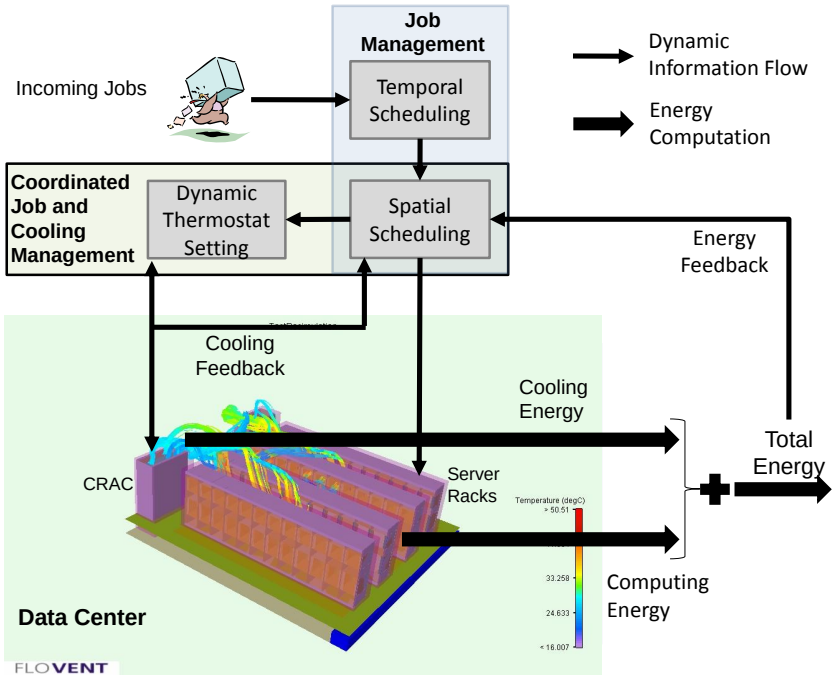

Fig. 2. Information flow during Coordinated Job and Cooling Management in HPC data centers.

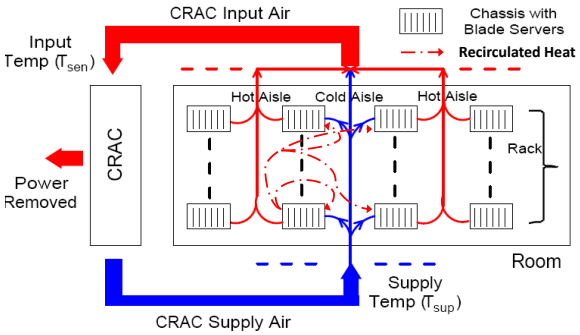

Fig. 3. Heat transfer mechanisms in data center

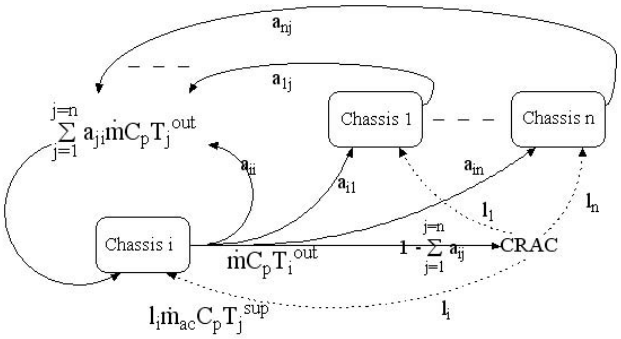

Fig. 4. Chassis level view of the air inlet, outlet and re-circulation.

perspectives. Section 7 presents the related work on energy-efficient job scheduling in data centers. Finally, Section 8 concludes the paper with a discussion on the various issues and future research directions.

2. System Model

This section provides the system model and assumptions to be used in designing the cooling aware spatial job scheduling algorithm (HTS) and computing algorithm energy inefficiency. First, the data center physical model is described in Section 2.1 followed by the job and machine environment in Section 2.2. Section 2.3 presents the constant and dynamic CRAC models and Section 2.4 presents their inter-dependencies with the data center job management and server power management.

Table 1
Scalar Symbols and Definitions.

Symbol	Definition
n	total number of chassis
h	inter event interval or event period
c_k^{tot}	the number of servers (blades) job k requires
r_{ac}	thermal capacity of air flowing out of the CRAC per unit time
r_{room}	thermal capacity of air in the data center room
f_{ii}	cold supply air fraction going from CRAC to chassis i
d_{ij}	heat recirculation coefficient from chassis i to j
x	mode of operation of CRAC, $x \in \{high, low\}$
t_{sw}	time taken by the CRAC to switch from one mode to the other
ω	idle chassis power consumption
α	power consumption of a chassis per unit of utilization
u	chassis utilization
N_h	total number of jobs in event period h
$T^{sup}(t)$	air temperature as supplied from the CRAC at time t
$T^{sen}(t)$	air temperature at the input of the CRAC at time t
T^{red}	manufacturer's redline inlet temperature
T_{high}^{th}	high thermostat setting for the CRAC
T_{low}^{th}	low thermostat setting for the CRAC
$(T_{high}^{th})^{max}$	upper bound on high thermostat setting for CRAC
$(T_{high}^{th})_i$	CRAC high thermostat setting requirement for server i
ΔT^{th}	temperature difference between the CRAC $high$ and low thermostat settings
P_{ex}^{x}	power extracted by the CRAC in mode x
P_j^{full}	power dissipation of chassis j at 100 % percent utilization. For any variable z, z^{full} denotes the value of z at 100 % utilization and z^{empty} denotes that at empty data center.
P_h^{comp}	total computing power at inter-event period h
$P^{AC}(t)$	power consumption of CRAC at time t
E_y	energy consumption for algorithm y
E_y^x	energy consumption of CRAC in mode x for algorithm y

2.1. *Physical Model*

The data center physical model includes the physical lay-out of (Section 2.1.1) and the computing equipment (server) safety (Section 2.1.2).

2.1.1. *Data Center Lay-out*

Contemporary HPC data centers use raised floors and lowered ceilings for cooling air circulation, with the computing equipment organized in rows of 42U racks arranged in an aisle-based layout, with alternating cold aisles and

hot aisles (Figure 3). The computing equipment is usually in blade-server form, organized in 7U chassis. Often, in data centers, server racks are provided with chiller doors, which cool down the hot air coming out of the blade servers before it enters the data center room [15].

The cooling of the data center room is done by the CRAC, a.k.a the *heating and ventilation air conditioner* (HVAC). They supply cool air into the data center through the raised floor vents. The cool air flows through the chassis inlet and gets heated up by convection from the computing equipments and hot air comes out of the chassis outlet. The hot air goes to the inlet of the CRAC which cools it down. However, depending on the design of the data center, parts of the hot air may recirculate within the room affecting the thermal map at various positions including the inlet of the CRAC and the chassis.

2.1.2. *Equipment Safety*

The CRAC has to supply cold air so that the inlet temperature of each chassis does not exceed the redline temperature (T^{red}), which otherwise would lead to throttling of the operation of the computing unit—an undesirable phenomenon with respect to HPC job performance (throughput and turnaround time). The inlet temperature of a chassis i is a result of heat contributed from two sources: 1) CRAC supply at a temperature of $\boldsymbol{T_{sup}}$ and 2) the recirculated heat from the outlet of all other chassis. Due to equipment placement within the data center the CRAC does not have uniform cooling effect everywhere. Hence, we assume that only l_i fraction of CRAC air supply reaches to each chassis i. Further, the heated air from the outlet of a chassis j gets recirculated and a fraction a_{ji} of it reaches the inlet of the chassis j as shown in Figure 4. Following conservation of mass principle we get,

$$\begin{aligned}
&\text{air flow rate through chassis } i = \text{fraction of air flowing from CRAC to chassis } i \times \text{air flow rate at CRAC outlet} \\
&+ \sum_{j=1}^{n} \big(\text{fraction of air recirculating from outlet of chassis } j \text{ to inlet of chassis } i \times \text{ air flow rate through chassis } j\big) \\
&\implies \text{air flow rate through chassis } i = l_i \times \text{air flow rate at CRAC outlet} + \sum_{j=1}^{n} \big(a_{ji} \times \text{ air flow rate through chassis } j\big)
\end{aligned}$$

It is assumed that the mass flow rate at the outlet of each chassis is the same and the specific heat is uniform throughout the room. Hence if r_{ac} and r are the thermal capacities per unit time of the air flowing out of the CRAC and the chassis i, respectively, then from the above result we get,

$$\text{Thermal capacity of air flowing through chassis } i = l_i \times r_{ac} + \sum_{j=1}^{n} \big(a_{ji} \times r\big) \tag{1}$$

If the inlet temperature at the chassis i is $T_i^{in}(t)$ at time t, then the amount of heat entering chassis i is the thermal capacity multiplied by $T_i^{in}(t)$. The heat input to chassis i from CRAC is similarly obtained by multiplying the thermal capacity of air flowing from CRAC to chassis i by the supply temperature. For recirculated heat, the thermal capacity of air coming out from chassis j and entering chassis i is to be multiplied by the outlet temperature of chassis j at time t, $T_j^{out}(t)$. Hence using the Equation 1 we get,

Heat input to chassis i in time dt = heat input from CRAC at chassis i in time dt +

recirculated heat to chassis i from all other chassis in time dt

$$\implies (l_i r_{ac} + \sum_{j=1}^{n} a_{ji} r) T_i^{in}(t)dt = l_i r_{ac} T^{sup}(t)dt + \sum_{j=1}^{n} a_{ji} r T_j^{out}(t)dt, \tag{2}$$

dt is a small time interval where temperature throughout the data center and power consumptions of chassis are assumed to be constant.

Further, the outlet temperature of the chassis i can be related to its inlet temperature and power dissipation using the principle of conservation of energy as follows:

Heat input to chassis i in time dt + Heat generated from chassis i in time dt = Heat output from chassis i in time dt

$$\implies (l_i r_{ac} + \sum_{j=1}^{n} a_{ji} r) T_i^{in}(t)dt + P_i(t)dt = r T_i^{out}(t)dt, \tag{3}$$

where $P_i(t)$ is the power generated by chassis i due to workload execution, which is constant in the time interval dt.

Vectorizing Equations 2 and 3 and simultaneously solving them to determine T_i^{in} for each i we get,

$$\boldsymbol{T^{in}}(t) = \boldsymbol{F T^{sup}(t)} + \boldsymbol{D P}(t), \tag{4}$$

where $\boldsymbol{F} = (\boldsymbol{L R_{ac}} + \boldsymbol{B R})^{-1} \boldsymbol{R} (\boldsymbol{R} - \boldsymbol{A}' \boldsymbol{R})^{-1} \boldsymbol{L R_{ac}}$, $\boldsymbol{D} = (\boldsymbol{L R_{ac}} + \boldsymbol{B R})^{-1} \boldsymbol{R} [(\boldsymbol{R} - \boldsymbol{A}' \boldsymbol{R})^{-1} - \boldsymbol{R}^{-1}]$, $\boldsymbol{L}$ is a $n \times n$ diagonal matrix in which the i-th element of the i-th column contain the l_i for all i, $\boldsymbol{B}$ is the $n \times n$ diagonal matrix whose diagonal elements are $\sum_j a_{ji}$, $\boldsymbol{T^{sup}}(t)$ is a n dimensional vector $\{T^{sup}(t)\}_n$, $\boldsymbol{T^{out}}(t)$ is a n dimensional vector $\{T_i^{out}(t)\}_{i=1\ldots n}$, and $\boldsymbol{R}$ and $\boldsymbol{R_{ac}}$ are $n \times n$ diagonal matrix populated with the $\dot{m} C_p$ and $\dot{m}_{ac} C_p$ values in the diagonal respectively. Equation 4 shows that the chassis inlet temperature depends on the fraction of CRAC supply temperature reaching the chassis and the amount of heat recirculation in the data center.

For the safe operation of each chassis, the inlet temperature should be below the redline temperature. Thus, using Equation 4, the redline constraint can be expressed as follows,

$$\text{chassis inlet temp} \leqslant \text{redline temp}$$

$$\implies \boldsymbol{F T^{sup}}(t) + \boldsymbol{D P}(t) \leqslant \boldsymbol{T_{red}}, \tag{5}$$

where $\boldsymbol{T^{red}}$ is an n dimensional vector $\{T_i^{red}\}_n$, T_i^{red} is the redline temperature for chassis i, n is the total number of chassis in the data center.

2.2. *Job and Machine Environment*

Given the data center physical model, this section describes the job and machine environment in the data centers. The state-of-the-art in commercially available data center management software follows a conventional job queuing and

issuing paradigm that focuses on optimizing *performance policy metrics*, those usually being *throughput* and *turn-around time*. The user front-end of a data center is the *submission* interface, i.e. the interface of the *scheduler*, which decides when and where (i.e. what servers) the jobs to be run at.

A *job submission* usually provides: a) the executable, b) the input data, c) the number of servers it requires and the *estimated runtime*, and d) other constraints such as a priority, and specific *computing node* preferences. A *computing node* is a chassis containing multiple blade servers. The job run-times are normally overestimated by the users [12]. We consider user estimated job turnaround times as the jobs' **deadlines**. The scheduler aborts the jobs that do not complete by the deadline. Thus, a scheduling algorithm has to ensure meeting of the job deadlines to avoid job abortion.

There are two types of decision making for job scheduling: i) *temporal* (i.e. when to start the execution of the jobs), which directly impacts the job throughput and turnaround times; and ii) *spatial* (i.e. where to execute the jobs), which can also impact the job throughput and turnaround times if jobs are assigned to servers with low computing capabilities (e.g. processor speed). *To ensure no degradation in throughput and turnaround time, this paper focuses only on energy-aware spatial scheduling decision making among the servers requested by the users during job submission.*

Further, an event based decision making for job scheduling is considered. An **event** comprises of arrival of new jobs (*job arrival*), beginning of job execution (*job start*) and end of job execution (*job completion*). **Inter event interval**, also referred to as **event period** (denoted by the symbol h), is the time between two consecutive job start and completion events. Computing power in a data center changes when a job starts or ends execution on a machine. Therefore, the computing power in any inter-event interval is constant over time. The following sections will describe the behavior of the CRAC unit in the data center and the inter-dependency of the cooling behavior with the computing power consumption.

2.3. *CRAC Behavior*

Two types of CRAC behavior is considered in this paper.

2.3.1. *Constant cooling model*

In constant cooling model, the CRAC, at all times, keeps a constant supply temperature $T^{sup}(t)$ at its outlet regardless of the temperature at the inlet (i.e., the sensed temperature $T^{sen}(t)$). This means the entire power, generated in the data center room, gets removed by the CRAC. From the conservation of energy at the CRAC, we get:

$$\text{Heat output from CRAC + heat generated in the data center room = heat input to the CRAC from the room}$$

$$\implies r_{ac}T^{sup}(t)dt + P_h^{comp}dt = r_{ac}T^{sen}(t)dt, \tag{6}$$

This model is a common assumption is previous work [5–8, 12]. Such an assumption makes the analysis easier and allows steady state simulation of the data center using CFD tools such as Flovent (`http://www.mentor.com/`

products/). However, variations in the CRAC input (i.e. ceiling temperature) and output (i.e. supply temperature) over time from temperature sensor measurements in the ASU HPC data centers (Figure 5) suggest otherwise. The difference between these two temperatures (that determines the power extracted by the CRAC unit) clearly shows two distinct values indicating two operational modes. In reality, the CRAC has a dynamic behavior, which is captured by the next model.

2.3.2. *Dynamic cooling model*

The CRAC can have many different modes of operation. For simplicity, in this paper, we assume two operational modes viz. *high* and *low* modes. During its operation the CRAC oscillates between the *high* and *low* modes. In each mode, the CRAC extracts a constant amount of power P_{ex}^{high} and P_{ex}^{low}, respectively ($P_{ex}^{high} > P_{ex}^{low}$). From the energy conservation in the CRAC, we get,

$$\begin{aligned} &\text{Heat input to the CRAC from the room} = \text{heat extracted by the CRAC} + \text{heat output by the CRAC to the room} \\ \implies & r_{ac} T^{sen}(t) dt = P_{ex}^{x} dt + r_{ac} T^{sup}(t) dt, \end{aligned} \tag{7}$$

where the CRAC operates in mode $x \in \{low, high\}$.

A **mode epoch** is the time duration for which the CRAC operates in a particular mode. In a mode epoch, since the CRAC only extracts a constant amount of power P_{ex}^{x}, we expect a linear variation of the inlet temperature over time [14]. The rate of the linear variation is proportional to the difference in power generated in the room P_{h}^{comp}, and the power extracted by the CRAC. Hence we get,

$$\begin{aligned} &\text{CRAC inlet temperature at time t} = \text{CRAC inlet temperature at time 0} + \text{rate of change of temperature} \times t \\ \implies & T^{sen}(t) = T^{sen}(0) + \frac{\text{Rate of heat generation in room} - \text{Rate of heat extraction by CRAC}}{\text{Thermal capacity of room}} \times t \\ \implies & T^{sen}(t) = T^{sen}(0) + \frac{P_{h}^{comp} - P_{ex}^{x}}{r_{room}} \times t, \end{aligned} \tag{8}$$

where $T^{sen}(0)$ is the initial input temperature of the CRAC when the CRAC moved to mode x and t is the duration for which the CRAC is in mode x. If $T^{sen}(t)$ goes above the high threshold temperature T_{high}^{th}, the CRAC switches from *low* to *high* mode and when $T^{sen}(t)$ goes below the low threshold temperature T_{low}^{th}, the CRAC switches from *high* to *low* mode ($T_{high}^{th} > T_{low}^{th}$). Generally the difference between the two threshold temperature is kept constant and only the high threshold temperature is varied to modulate the CRAC behavior.

Further, once the input temperature goes above T_{high}^{th} or goes below T_{low}^{th} the CRAC takes an additional t_{sw} amount of time to switch between modes. During this t_{sw} time the CRAC operates in the same mode as in the previous mode epoch. Hence, at the end of each mode period the maximum and minimum CRAC inlet temperatures are given as follows:

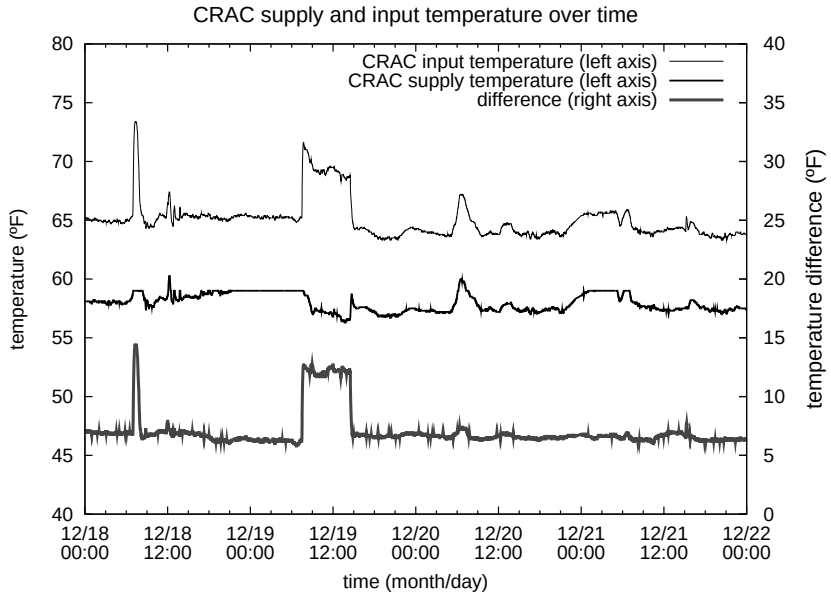

Fig. 5. Variations in the CRAC input and output temperature based on actual sensor measurements in the ASU HPC data center. The difference in these temperatures indicates the two operational modes of the CRAC.

max CRAC inlet temp = CRAC high threshold temp + temp change due to operation in low mode for t_{sw} time

$$\implies T^{sen}_{max} = T^{th}_{high} + \frac{P^{comp}_h - P^{low}_{ex}}{r_{room}} t_{sw},$$

min CRAC inlet temp = CRAC low threshold temp + temp change due to operation in high mode for t_{sw} time

$$\implies T^{sen}_{min} = T^{th}_{low} + \frac{P^{comp}_h - P^{high}_{ex}}{r_{room}} t_{sw}. \tag{9}$$

2.3.3. *CRAC Power Consumption*

For the dynamic mode the CRAC power consumption depends on the CRAC power mode (i.e. the power extracted by the CRAC) and Coefficient Of Performance (COP) of the CRAC. The COP of the CRAC to supply air at temperature $T^{sup}(t)$ at an instance t is normally given by $COP(T^{sup}(t)) = \frac{T^{sup}(t)}{T^{sen}(t) - T^{sup}(t)}$, where $T^{sen}(t)$ is the CRAC input temperature (Figure 3) at the instance t [16]. The above assumption on the COP of the CRAC unit enables the computation of the energy dissipated by a cooling unit in the operating mode x. The COP is given by $\frac{r_{ac} T^{sup}(t)}{P^x_{ex}}$. The power consumption to run the CRAC at time t is given by:

$$P^{AC}(t) = \frac{\text{Rate of heat extraction by CRAC}}{\text{Coefficient of performance}} = \frac{P^x_{ex}}{COP(T^{sup}(t))} = \frac{(P^x_{ex})^2}{r_{ac} T^{sup}(t)}. \tag{10}$$

Any technique to reduce the CRAC power consumption has to operate in lower modes, reducing the P^x_{ex}, and increase the $T^{sup}(t)$ as far as possible. For the constant cooling model of the CRAC the power consumption can be obtained on replacing P^x_{ex} by the total power dissipated in the data center at an event period P^{comp}_h.

2.4. *Inter-dependency of Cooling and Job Management*

Given the data center physical model, machine environment, and the CRAC behavior in the previous subsections, this subsection summarizes the inter-dependencies among the cooling, job, and power management. In this section we focus on the dynamic cooling model for the CRAC as the dependencies of the cooling, job and power management for constant cooling has been discussed in our previous work [12].

It should be noted that as the data center utilization increases, power consumption at the chassis increases; requiring lower supply temperature to meet the redline (Equation 5) [12]. The supply temperature is maximum for 0% utilization and minimum for 100% utilization. Generally, however, if there is heat recirculation, the heat input to the chassis increases; thus requiring lower $T^{sup}(t)$ to keep the temperature within the redline temperature [12]. Therefore, it is important to predict the maximum supply temperature from the CRAC. In a particular CRAC mode, the supply temperature also changes linearly at the same rate as the CRAC input temperature (Equation 7). It should be noted that the maximum temperature can be reached when the power extraction from the CRAC is low, i.e. when it is operating in the low mode and the inlet temperature is maximum. Hence combining Equations 7 and 9 we get,

$$T^{sup}_{max} = T^{th}_{high} + \frac{P^{comp}_{h} - P^{low}_{ex}}{r_{room}} t_{sw} - \frac{P^{low}_{ex}}{r_{ac}}, \tag{11}$$

It can be concluded from Equation 11 that the maximum supply temperature from the CRAC depends on the high thermostat settings and the computing power in the data center. Hence, job management (i.e. scheduling and placement) and server power management, both of which determine the computing power consumption, in conjunction with the CRAC management (i.e. dynamically updating the thermostat settings) need to be performed in such a way that for the maximum supply temperature, the redline temperature is not violated (Equation 5).

We assume a programmable thermostat where the set temperatures can be dynamically changed. However, the CRAC maintains a constant difference, ΔT^{th}, between the *high* and *low* thermostat settings [3] (i.e. $\Delta T^{th} = T^{th}_{high} - T^{th}_{low}$ and $T^{th}_{high} > T^{th}_{low}$). Depending on this difference, the minimum possible supply temperature, T^{sup}_{min}, from the CRAC can be determined (in the same way as T^{sup}_{max} in Equation 11) when the input temperature reaches the low thermostat set point and the CRAC operates at a high mode (i.e. the power extracted by the CRAC is higher):

$$T^{sup}_{min} = T^{th}_{low} + \frac{P^{comp}_{h} - P^{high}_{ex}}{r_{room}} t_{sw} - \frac{P^{high}_{ex}}{r_{ac}} \Longrightarrow T^{sup}_{min} = T^{th}_{high} - \Delta T^{th} + \frac{P^{comp}_{h} - P^{high}_{ex}}{r_{room}} t_{sw} - \frac{P^{high}_{ex}}{r_{ac}}. \tag{12}$$

Equation 12 shows how an increase in the high thermostat set temperature can increase the supply temperature; thus potentially reducing the CRAC power consumption (from Equation 10). This paper designs a cooling-aware spatial job scheduling algorithm, HTS, that allows higher thermostat set temperatures and hence lower SP-EIR when integrated with dynamic updates to the CRAC thermostat settings.

[3] In the rest of the paper, changing the high thermostat set temperature refers to changing both the set temperatures.

3. Problem Definition

This paper deals with energy-efficient spatial scheduling of HPC jobs on the heterogeneous parallel machine environment in the data centers. The problem is defined as:

> *Given a heterogeneous parallel machine environment and a job environment such that a job k requires c_k^{tot} number of processors, find the spatial distribution of jobs (a vector of chassis numbers in which the job k is executed), and the thermostat set points of the CRAC with every job start and completion to* ***minimize*** *the total energy consumption such that the job throughput and turnaround times are within the user expectations.*

It should be noted that the problem has two major decision-making aspects: i) spatial job scheduling (i.e. placement of the jobs to the appropriate servers), and ii) CRAC thermostat control (i.e. determining the thermostat set temperatures). Spatial job scheduling is similar to the Knapsack problem, which is again NP-complete [17]. The requirement for minimizing the energy consumption makes the problem harder. There has been various heuristics proposed over the years ranging from using low-power servers [2–4] to reducing heat recirculation [7, 12]. A heuristic spatial job-scheduling algorithm, called LRH, was developed in [12] that has two distinct characteristics: i) LRH statically ranks the nodes so that the choice of nodes to place a job can be very fast (ideal situation of dynamically re-calculating the nodes' ranks can increase the complexity); and ii) the static rank of a node is assigned based on the node's *contribution* in heat recirculation, i.e. the total recirculated heat from the node to all other nodes.

The ranking mechanism in LRH however does not consider the *impact* of the cool air from the CRAC to the nodes. This paper proposes the HTS algorithm that enables *cooling-aware* node ranking based on: i) the supplied cool air from the CRAC with different modes of operations, and ii) the recirculated hot air from all the other nodes. Further, HTS performs coordinated cooling management by dynamically updating the CRAC thermostat set temperatures.

4. Highest Thermostat Setting (HTS)

This section presents the Highest Thermostat Setting (HTS) algorithm. As described in the previous sections, the energy inefficiency for any algorithm depends on the thermostat settings of the CRAC. With a high setting on the thermostat, the SP-EIR reduces. The HTS integrates the control of the CRAC thermostat settings with spatial job scheduling (i.e. job placement) so that the thermostat can be set as high as possible without violating the redline temperatures and the SLAs.

4.1. *Algorithm Design*

The principal intuition behind the algorithm is to: i) *statically rank the servers* from best to worst (according to the potential load, i.e. the required thermostat setting, incurred on the CRAC); ii) *place (i.e. assign)* temporally scheduled jobs to the best ranked servers; iii) *dynamic determination of the required CRAC thermostat setting* after the job placement is performed; and iv) dynamically set the thermostat to the required value. Figure 6 presents the intuitive

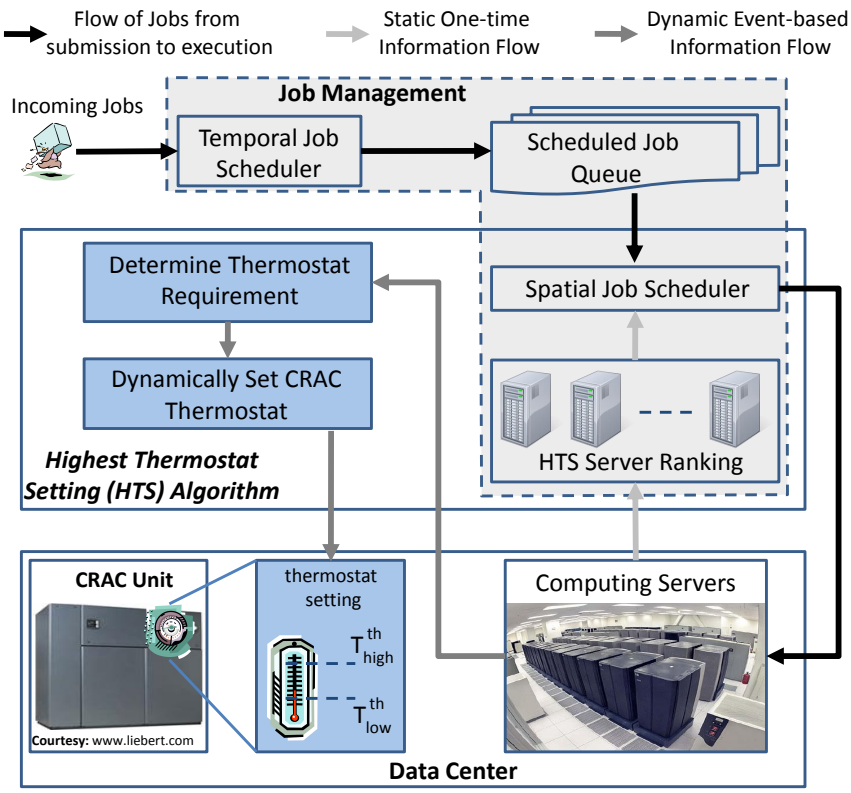

Fig. 6. Architecture and work-flow of HTS.

operational flow showing the aforementioned four operations and their inter-dependencies. The following subsections describe these four operations.

4.1.1. *Server Ranking*

The ranking of the servers is necessary for the job placement to assign jobs based on the server ranks. To counter the energy inefficiencies because of the dependency on the CRAC thermostat setting, the HTS algorithm ranks the servers based on their requirement on the CRAC thermostat to keep the inlet temperature within the redline temperature. The servers are ranked from highest to lowest thermostat temperature requirement. The jobs are then placed according to the server ranking; thus allowing the CRAC thermostat setting to be increased.

As shown in Equation 20, the power consumption of the servers themselves impact the upper bound on the CRAC thermostat settings. Dynamically ranking the servers depending on the placement is however not efficient, since the placement itself is dependent on the ranking. Further, finding the optimal placement is NP-complete and may require hours of operation [12]. As such HTS performs a static ranking of the servers based on full utilization of the data center, which yields the thermostat setting requirement for a server i as follows:

Algorithm 1 HTS integrated in the spatio-temporal job scheduling

procedure INITIALIZATION( )
Group nodes with respect to power specifications.
Sort groups with respect to computing efficiency (i.e. MIPS/watt).
Perform server rankings, $\boldsymbol{R}$, according to the requirement of thermostat set temperature to meet the redline for 100% utilization (Equation 13).
end procedure

procedure HTS( )
Place job to available node(s) with the lowest rank in $\boldsymbol{R}$.
Determine the power distribution vector,$\boldsymbol{P_h}$.
Set the CRAC thermostat using SETTHERMOSTAT($\boldsymbol{P_h}$).
end procedure

procedure SETTHERMOSTAT($\boldsymbol{P_h}$)
Set high thermostat setting (T_{th}^{high}) as
$F^{-1}T_{red} - \left[\frac{P_h^{comp}-P_{ex}^{low}}{r_{room}}t_{sw} - \frac{P_{ex}^{low}}{r_{ac}}\right] - F^{-1}D\boldsymbol{P_h}$ (Eq. 20)
end procedure

procedure UPONJOBCOMPLETION( )
Dispatch the next job in this group's queue using HTS ().
Determine the power distribution vector,$\boldsymbol{P_h}$.
Remove the job from the queue.
Set the CRAC thermostat using SETTHERMOSTAT($\boldsymbol{P_h}$).
end procedure

procedure UPONJOBARRIVAL( )
if job comes with node restrictions **then**
Insert the job in the queue of the specified node group based on a scheduling policy (e.g. FCFS or EDF).
else
Insert the job in the most energy-efficient group queue. based on a scheduling policy (e.g. FCFS or EDF).
end if
for each node group, from the most to least efficient node **do**
if job's finish estimation > deadline **then**
(1) Insert the job in an "opening" having enough free servers for enough time. Continue with next job.
(2) If Step 1 fails, push-fit the job at an earlier "time" if shifting jobs still make the deadline.
Continue with next job.
(3) If Step 2 fails, add the job to next group's queue.
end if

if required nodes in this group are idle **then**
Dispatch the job in this group's queue using HTS ().
Remove the job from the queue.
end if
end for
end procedure

CRAC high threshold requirements for chassis i + temperature rise due to switching time of the CRAC,

- difference in temperature due to CRAC power extraction = maximum allowed inlet temperature of chassis i,

$$\Longrightarrow (T_{high}^{th})_i + \frac{(P_h^{comp})^{full} - P_{ex}^{low}}{r_{room}}t_{sw} - \frac{P_{ex}^{low}}{r_{ac}} = T_i^{in},$$

$$\Longrightarrow (T_{high}^{th})_i = \frac{T^{red} - \sum_j d_{ij}P_j^{full}}{\sum_j f_{ij}} + \frac{P_{ex}^{low}}{r_{ac}} - \frac{(P_h^{comp})^{full} - P_{ex}^{low}}{r_{room}}t_{sw}, \quad (13)$$

where P_j^{full} is the power consumption of chassis j at 100% utilization. The servers are then statically ranked in the decreasing order of $(T_{high}^{th})_i$. The server ranking, is an one-time initialization process (performed in procedure *Initialization* in Algorithm 1). The ranks are represented by ranking vector $\boldsymbol{R}$.

4.1.2. *Job Placement*

As shown in Figure 6, the Spatial Job Scheduler takes the jobs from the scheduled job queue [4] and places them to the servers based on their ranks (procedure *HTS* in Algorithm 1). Such rank-based job placement can be easily incorporated in the current job management softwares. For example, the widely used Moab job management software allows setting up server priorities in the software's configuration [18]. The server ranking presented in the previous section can be used to prioritize the servers in Moab. The priority-based job assignment can then be enabled for job placement.

[4] Here the assumption is that the jobs are already temporally scheduled, i.e. the jobs' start times are decided by some temporal scheduling algorithm.

4.1.3. *Dynamic Thermostat Setting*

After placing the jobs, EDF-HTS sets the thermostat setting to the highest possible value given by Equation 20. As shown in Figure 6, first the required thermostat set temperature is determined followed by actually setting the thermostat to the required value. Equation 20 is used to determine the required thermostat setting after the job placement is performed. Unlike the server ranking in Section 4.1.1 (where the data center was assumed to be fully utilized), the thermostat requirement is computed based on the actual server utilization after job placement.

4.2. *Energy inefficiency of HTS*

Since HTS performs cooling-aware spatial-scheduling and coordinates with the cooling management, the SP-EIR is lower than the most efficient spatial-scheduling algorithms in the literature. For the representative ASU HPC data center, the theoretical maximum SP-EIR of HTS is 1.013 (obtained a synthetic workload in the range of 0-100% data center utilization), which is around 15% lower than the LRH algorithm, the most energy-efficient online job placement algorithm [12]. Section 6 presents the results in further detail.

4.3. *Spatio-temporal Job Scheduling with HTS*

Given the HTS algorithm, which integrates spatial job scheduling with cooling management, this section describes how HTS can fit into the overall job management involving spatio-temporal job scheduling in data centers. Algorithm 1 presents how HTS is integrated with the spatio-temporal job scheduling. An event based scheduling approach is taken where decision making is performed for the following three events:

– **Arrival Event**: Temporal scheduling decision is taken when a job arrives, i.e. the job is submitted by the user to the queue. FCFS-Backfill or the EDF approach is for temporal scheduling. After this decision, if some job is projected to finish after its deadline, then it is scheduled to execute at an earlier time, thus having the data center run multiple jobs at the same time. The procedure UponJobArrival in Algorithm 1 performs the temporal scheduling.
– **Start Event**: Two decisions are involved when a job is scheduled to start execution: 1) determination of the placement of the job, which is done following the HTS approach (procedure HTS in Algorithm 1); and 2) determination of the thermostat setting of CRAC. Given the placement of jobs, HTS sets the T^{th}_{high} of the CRAC to its upper bound, which can be obtained from Equation 20.
– **End Event**: When a job finishes execution, the placement changes since the job is removed from the server and HTS resets the thermostat setting of the CRAC to a new value following the upper bound in Equation 20.

Note that if we consider the HTS ranking in Equation 13 for constant cooling case, then t_{sw} will be zero. $T^{th}_{high} - \frac{P^{low}_{ex}}{r_{ac}}$ can be considered as the supply temperature of the CRAC. Thus, the ranking strategy simplifies to ordering the servers in decreasing order of supply temperature requirements (Equation 14).

$$(T^{sup}_{min})_i = \frac{T^{red} - \sum_j d_{ij} P^{full}_j}{\sum_j f_{ij}}, \tag{14}$$

This implies sorting the servers in increasing order of the recirculated heat $\sum_j d_{ij} P^{full}_j$, which is same as the LRH ranking [12]. Hence, we see that for constant cooling case HTS reduces to the LRH ranking strategy. Hence, constant cooling is a special case of the dynamic model.

4.4. *Time Complexity*

Upon a job arrival the job queue is searched to find a proper start time of the job. This has a complexity of $O(N_h)$ at the worst, where N_h is the total number of jobs in the event period. For spatial scheduling, since the static server ranking is used, it is only required to place the job in the required number of servers, which has a complexity of $O(n)$, where n is number of chassis. Thus, the total complexity for spatio-temporal scheduling of a job $O(n + N_h)$. Since there are a total N_h job arrivals in an event period h, the complexity of spatio-temporal scheduling in the event period is $O(N_h(n + N_h))$. The complexity for only spatial scheduling in the event period is given by $O(nN_h)$.

5. SP-EIR for Spatial Job Scheduling

This section defines the SP-EIR of the spatial scheduling algorithms to compare the quality of the solutions. It should be noted that spatial job scheduling do not impact the job performance as long any affinity of the HPC jobs to the servers are maintained. As such, the quality of the solution provided by an algorithm *Alg* is evaluated with respect to the objective of minimizing the total energy of the schedule.

The *SP-EIR* of *Alg* is defined as the ratio, $\frac{E_{Alg}}{E_{opt}}$, of the total energy consumption, E_{Alg}, for *Alg*, to the total energy consumption, E_{opt}, for the optimal algorithm that minimizes the total energy consumption. Note that the CRAC oscillates between the *high* and *low* modes during its operation as (Section 2.3). The SP-EIR for an algorithm *Alg* can then be obtained by calculating the energy consumption for *Alg* and finding its ratio to the energy consumption in optimal case. The SP-EIR for any algorithm *Alg* can thus be given as:

$$\begin{aligned} \frac{E_{Alg}}{E_{opt}} &= \frac{\text{energy consumption of } Alg \text{ under all CRAC modes}}{\text{energy consumption of } opt \text{ under all CRAC modes}} \\ \implies \frac{E_{Alg}}{E_{opt}} &= \frac{E^{high}_{Alg} + E^{low}_{Alg}}{E^{high}_{opt} + E^{low}_{opt}}, \end{aligned} \tag{15}$$

where E^x_{Alg} and E^x_{opt} are the energy consumptions of CRAC in mode x for algorithm *Alg* and the optimal case, respectively. The energy consumption at a particular CRAC mode can be obtained as follows, by integrating the CRAC power consumption, Equation 10, at that mode over the period for which the CRAC remains in the mode.

$$E^x = \int_0^{t_x} \frac{(P^x_{ex})^2}{r_{ac} T^{sup}(t)}\, dt, \tag{16}$$

where t_x is the time for the CRAC to remain in a particular mode x. From the linear relationship of $T^{sup}(t)$ with $T^{sen}(t)$ from Equation 7 and the variation of $T^{sen}(t)$ over time, Equation 11, the following closed form expression for the energy consumption of the CRAC in a mode period can be obtained after performing the integration in Equation 16:

$$E^x = \left| P^x_{ex} \frac{\Delta T^x_{ac}}{\dot{T}^{sen}_x} \left[ln\{ \frac{(T^{sup}_{min})}{(T^{sup}_{max})} \} \right] \right|, \tag{17}$$

where ΔT^x_{ac} is the temperature difference maintained by the CRAC between its inlet and the supply in a particular mode x (this can be obtained from Equation 7), $\dot{T}^{sen}_x$ is the rate of change of CRAC input temperature obtained from Equation 8. The values of (T^{sup}_{min}) and (T^{sup}_{max}) can be obtained from Equation 12 and Equation 11 in terms of T^{sen}_{max} and T^{sen}_{min} respectively. The *SP-EIR* depends on these energy consumption values and hence is directly related to the ratio of maximum to minimum supply temperature. The closed form equation for *SP-EIR* can be obtained by substituting Equation 17 in 15.

It can be observed from Equation 17 that the CRAC energy consumption in a mode period nears zero as the ratio of T^{sen}_{max} to T^{sen}_{min} approaches unity. The optimal algorithm will thus have the ratio $\frac{T^{sen}_{max}}{T^{sen}_{min}}$ as close to unity as possible. It can be easily observed as the ratio between the maximum and minimum supply temperature in a mode period gets closer to unity the *SP-EIR* also improves. The ratio of the maximum to minimum supply temperature in a mode period can be obtained from Equations 11 and 12,

$$\frac{(T^{sup}_{max})^x}{(T^{sup}_{min})^x} = \frac{T^{th}_{high} + \frac{P^{comp}_h - P^{low}_{ex}}{\alpha m_{room} C_p} t_{sw} - \frac{P^x_{ex}}{r_{ac}}}{T^{th}_{high} - \Delta T^{th} + \frac{P^{comp}_h - P^{high}_{ex}}{\alpha m_{room} C_p} t_{sw} - \frac{P^x_{ex}}{r_{ac}}}. \tag{18}$$

From Equation 18, it can be seen that the ratio depends on: 1) high threshold temperature setting of the CRAC, higher the threshold setting more close the ratio to unity and 2) the difference in the power extracted by the CRAC in its different modes, higher the difference between P^{high}_{ex} and P^{low}_{ex} lower is the ratio. However, if the switching time of the CRAC is low the effect of extracted power in different modes on the ratio is negligible. The *SP-EIR* hence depends on the dynamic behavior of the CRAC and gets better if the threshold temperature is increased to a high value.

For the constant cooling case, during an event period the $T^{sup}(t)$ and the power extracted by the CRAC P^{comp}_h remains constant. Hence, integrating Equation 10, the energy consumption of the CRAC in an event period is given by,

$$E^{const}_{CRAC} = \frac{(P^{comp}_h)^2}{r_{ac} T^{sup}(t)} \times h, \tag{19}$$

Different spatial scheduling algorithms affect the $T^{sup}(t)$ in an event period. The *SP-EIR* for constant cooling case thus only depends on the ratio of the supply temperature requirements imposed by an algorithm *alg* to that by the optimal algorithm *opt*. Hence for constant cooling, the supply temperature requirement solely depends on the heat recirculation in the data center and the *SP-EIR* does not depend on the dynamic behavior of the CRAC. In such a case the HTS and LRH algorithms will have the same energy inefficiency, which is not observed in our simulation results (Section 6).

5.1. *Thermostat set point limits and its dependency on heat recirculation*

As discussed in Section 5, an algorithm with *SP-EIR* near to unity will attempt to set the thermostat to a high value. However, the redline constraint in the data center puts an upper limit on the thermostat setting that can be obtained from Equation 5. Hence there is an upper bound on the maximum $T^{sup}(t)$ in a mode epoch, which can be obtained from Equation 11; and using this maximum $T^{sup}(t)$ in the redline constraint, an upper bound on the *high* thermostat setting can be obtained as follows:

$$(T^{th}_{high})^{max} \leqslant min(\boldsymbol{F}^{-1}(\boldsymbol{T_{red}} - \boldsymbol{D}\boldsymbol{P}_h)) + \frac{P^{low}_{ex}}{r_{ac}} - \frac{P^{comp}_h - P^{low}_{ex}}{r_{room}} t_{sw} \tag{20}$$

Notice here that the highest T^{th}_{high} that an algorithm can set depends on the recirculation in the data center, the computing power consumption (P^{comp}_h), and the placement vector, $\boldsymbol{P}_h$. Given a utilization of the data center, the optimal algorithm will always attain the highest allowable T^{th}_{high} setting.

For a data center with no recirculation, $\boldsymbol{D}$ becomes a matrix of all zeroes and the upper bound on the T^{th}_{high}only depends on the utilization of the data center. It follows from Equation 20 that the upper bound on T^{th}_{high} decreases with increase in the recirculation in the data center. Hence, with the increase in the recirculation the SP-EIR also decreases.

An interesting observation in this regard is that the SP-EIR depends solely on the recirculation in the data center assuming that Alg controls the thermostat set point. If investments are made in making the data center free of recirculation, then greater energy savings can be achieved. The tradeoff between the infrastructure investments and the energy savings has to be considered by the data center designers. The following section presents how to compute an upper bound of the SP-EIR for any algorithm given the recirculation in a data center.

5.2. *Bound on Energy inefficiency*

The maximum value of $\frac{E_{Alg}}{E_{opt}}$ gives the upper bound on the SP-EIR for any algorithm. In order to get the upper bound we need to maximize the numerator and minimize its denominator. It can be observed that if T^{th}_{high} for Alg is decreased then the numerator increases since the cooling energy consumption would increase with higher thermostat settings. Similarly, if T^{th}_{high} for opt is increased the denominator decreases, hence increasing the SP-EIR. So an upper bound on the SP-EIR can be obtained if T^{th}_{high} for Alg is set to the lowest possible value, i.e. at 100 % utilization while T^{th}_{high} is set to the highest possible value, i.e. at 0 % utilization. Thus, the upper bound on the SP-EIR can be given as follows:

$$\frac{E_{Alg}}{E_{opt}} \leqslant \frac{\text{energy consumption of } Alg \text{ under 100\% utilization}}{\text{energy consumption of } Opt \text{ under 0\% utilization}}, \tag{21}$$

The numeric value of the upper bound can be calculated for a data center by plugging in the specific values of the recirculation coefficient for the data center. For the ASU HPC data center, the upper bound for any algorithm is 1.69. Given the SP-EIR, the following section defines the problem addressed in this paper.

6. Simulation Study

The previous sections presented: i) a new integrated job scheduling algorithm, EDF-HTS, that combines dynamic control of thermostat set temperatures with spatio-temporal job scheduling under linear cooling model; and ii) an analytical study to determine energy inefficiencies of any spatial job scheduling algorithm in data centers. This section performs simulation study to show that HTS indeed achieves better SP-EIR than other spatial scheduling algorithms, and EDF-HTS reduces the total data center energy consumption. In this simulation study, we only present simulation results for dynamic CRAC behavior. The simulation results for constant CRAC model were presented in [12] and hence are omitted to avoid repetition.

6.1. *Evaluation Methodology*

Evaluations are performed for two cases: 1) Idle servers kept on and 2) Idle servers turned off. For each of these cases we do an analysis on the SP-EIR and the total energy consumption of the algorithms.

6.1.1. *SP-EIR*

The SP-EIR of HTS is compared with that of two spatial job scheduling algorithms: i) LRH, which tries to reduce the heat recirculation in the data center and is shown to have the best energy savings when used in conjunction with the EDF temporal scheduling algorithm [12], and ii) MTDP, which performs server consolidation in the data center (turning servers *on* or *off*) [7]. For a fair comparison, HTS and MTDP are compared for *idle chassis turned off* case only. The CRAC thermostat setting is assumed to be set to a constant value when either LRH or MTDP algorithm is used. This value is usually set to handle the worst case situation, i.e. for 100% data center utilization. Further, we consider two more cases for a fair comparison: i) thermostat setting to the highest value allowable for the *m*aximum data center utilization for a particular job trace considered in the simulation (LRH and MTDP, when used with such thermostat settings, are referred as LRHm and MTDPm, respectively), and ii) *d*ynamically varying the thermostat according to the requirement (LRH and MTDP, when used with such dynamic thermostat settings, are referred as LRHd and MTDPd, respectively). The computation of E_{opt}, in order to calculate the SP-EIR, is non-trivial since the most energy efficient solutions are heuristic in nature without guaranteeing optimality [12]. In the simulation, the lower bound on the optimal energy consumption is calculated by minimizing the value of $max_i(\boldsymbol{DP})$ for any utilization. The minimization of $\boldsymbol{DP}$ is done by assigning a separate power distribution to each row in the $\boldsymbol{D}$ matrix so as to minimize the value of $\sum_j d_{ij}P_j$ for all i.

6.1.2. *Total Energy Consumption*

The HTS is used in conjunction with both FCFS-Backfill and EDF temporal scheduling algorithm (referred as FCFS-Backfill-HTS and EDF-HTS, respectively). While FCFS-Backfill is the most common practice in the contemporary data

centers to improve throughput and utilization, EDF has been shown to be more energy-efficient [12]. The EDF-HTS and FCFS-Backfill-HTS are verified with the following spatio-temporal job scheduling algorithms.

(i) FCFS-Backfill-LRH: This algorithm uses FCFS-Backfill, the most widely used temporal scheduling algorithm in current data centers. It also tries to maximize the throughput and utilization of the data center. For spatial scheduling LRH algorithm is used.

(ii) EDF-LRH: This algorithm uses the EDF for temporal scheduling and LRH for spatial scheduling. To the best of our knowledge, this is the best online energy-efficient spatio-temporal job scheduling algorithm for HPC data centers [12]. LRHm and LRHd are also used with EDF (referred as EDF-LRHm and EDF-LRHd, respectively). However, none of these consider controlling the cooling set points in the scheduling decisions. Thus, their comparison with EDF-HTS will hint towards the benefits of doing integrated cooling control.

(iii) EDF-MTDP: Comparison of EDF-HTS with EDF-MTDP, EDF-MTDPm, and EDF-MTDPm will highlight the advantages of doing integrated cooling over cooling-oblivious server consolidation.

6.1.3. *Throughput per Unit Energy*

This is defined as the number of jobs serviced (i.e. completed) per unit time per unit of energy consumed. The throughput (i.e the number of jobs completed per unit time) depends on the temporal scheduling algorithm since the spatial scheduling in performed among the requested servers of the jobs (Section 4.3). Apart from the aforementioned spatio-temporal scheduling algorithms, the FCFS-Backfill-MTDP (i.e. MTDP spatial scheduling in conjunction with the FCFS-Backfill temporal scheduling) is also compared with the FCFS-Backfill-HTS algorithm. Comparison of FCFS-Backfill-MTDP with FCFS-Backfill-HTS will show the advantages of coordinated job and cooling management over server consolidation in terms of the throughput per unit energy.

6.2. *Simulation Setup*

FloVENT [19], a CFD simulation software is used to conduct thermal simulations to obtain heat distribution matrix. Based on the ASU HPCI data center physical layout, a data center simulation model is created with physical dimensions 9.6 m × 8.4 m × 3.6 m, which has two rows of industry standard 42U racks arranged in a typical cold aisle and hot aisle layout. The cold air is supplied by one CRAC unit with the flow rate 8 m^3/s. The cold air rises from raised floor plenum through vent tiles, and exhausted hot air returns to the air conditioner through ceiling vent tiles. There are ten racks and each rack is equipped with five 7U (12.25-inch) chassis. There are two different types of computing equipment in the data center. Among the fifty chassis, there are thirty Dell PowerEdge 1955 (i.e. three racks) and twenty Dell PowerEdge 1855 chassis.

Table 2
Power Consumption Parameters

	ω	α
PowerEdge 1855	1820	72
PowerEdge 1955	2420	175

6.3. *Equipment Power Consumption*

Power measurements of Dell Power Edge 1855 and 1955 blade servers were performed using the DUALCOM [20] power meter from CyberSwitching Inc. Using the power measurements of the blade systems and performing linear regressions on the data, the idle chassis power consumption (ω) and the single fully-utilized server power consumption (α) values for the simulation runs [21, 22], as given in Table 2 were computed. The simulations assume that the jobs are CPU-intensive. The estimated power consumption of the resulting linear function has an error of 0.4–9% from the actual measurements. For a different utilization $u < 100\%$ the power consumption was scaled following a linear equation:

$$P = \omega + (u/100)\alpha. \tag{22}$$

6.4. *Data center job profile*

We used the ASU data center job traces of around one and half year for the simulation [5] . The job traces provide: i) the job arrival times, ii) their corresponding deadlines, iii) the number of processors required (c_k^{tot}), and iv) the job start and finish times using the FCFS-Backfill scheduling. From this job log, a set of time-contiguous jobs are selected for each simulation run based on the peak utilization during that interval.

6.5. *CRAC cooling model*

The CRAC had two operating modes. The temperature difference between the high and the low threshold is kept at a constant value of 15 °C. For EDF-LRH the high threshold temperature of the CRAC was kept at 30 °C while the low threshold was kept at 15 °C. The power consumption of the CRAC at the high mode was 350 KW which is greater than the maximum computing power of the data center. The power consumption in the low mode was kept at 100 KW which is equal to the idle power consumption of the ASU HPC data center. The CRAC mode switching time was kept at 3 seconds.

6.6. *Results*

EDF-HTS, when augmented with idle server turn-off, is the most energy-efficient algorithm while FCFS-Backfill-HTS, when augmented with idle server turn-off, has the maximum throughput per unit of energy consumption.

[5] The job traces are available online at `http://impact.asu.edu/WorkLoad/Log.zip`.

Table 3
Percentage of Total Energy Savings in EDF-HTS for Different Utilization (Idle Chassis kept On)

Data center Utilization	Percentage of energy savings of EDF-HTS with respect to				
	FCFS-Backfill-LRH	FCFS-Backfill-HTS	EDF-LRH	EDF-LRHm	EDF-LRHd
5%	12.41	10.65	3.70	3.32	0.87
40%	5.70	3.27	1.85	1.49	0.83
80%	3.30	0.85	1.40	1.31	0.73

Table 4
Percentage of Total Energy Savings in EDF-HTS for Different Utilization (Idle Chassis turned Off)

Data center Utilization	Percentage of energy savings of EDF-HTS with respect to							
	FCFS-Backfill-LRH	FCFS-Backfill-HTS	EDF-LRH	EDF-LRHm	EDF-LRHd	EDF-MTDP	EDF-MTDPm	EDF-MTDPd
5%	23.78	21.30	12.53	12.30	5.17	8.54	8.17	5.17
40%	21.50	17.22	16.00	15.56	10.84	9.00	8.73	5.73
80%	15.80	10.81	9.03	8.86	0.66	3.81	3.47	0.47

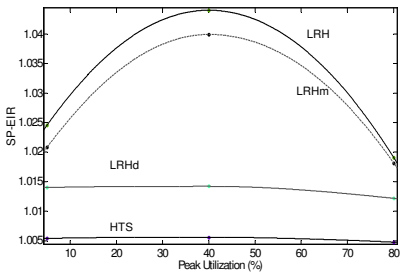

Fig. 7. Energy inefficiency of the total energy of HTS and LRH when idle chassis are kept on. The plots are interpolated from the energy consumption for 5%, 40%, and 80% peak utilization in ASU HPC data center.

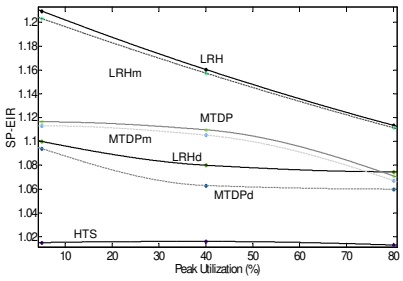

Fig. 8. Energy inefficiency of the total energy of HTS, LRH, and MTDP when idle chassis are turned off. The plots are interpolated from the energy consumption for 5%, 40%, and 80% peak utilization in ASU HPC data center.

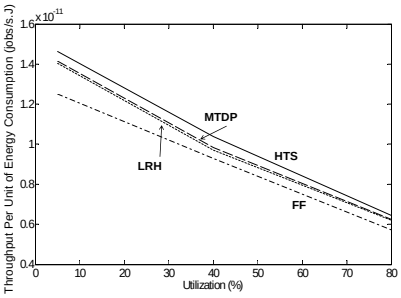

Fig. 9. Job throughput per unit of energy consumption when different spatial scheduling is used with FCFS-Backfill temporal scheduling. The plots are interpolated from the throughput per energy consumption for 5%, 40%, and 80% peak utilization in ASU HPC data center.

The percentage savings in energy obtained by EDF-HTS with respect to other algorithms for idle on and idle off cases are shown in the Table 3 and Table 4, respectively. For lower peak utilization, the savings of EDF-HTS with respect to EDF-LRH is high while it decreases for higher utilization. This is because at low peak utilization there are more options to place a job and achieve higher thermostat settings. Further, it can be observed that if the thermostat setting is kept constant (as in EDF-LRH and EDF-LRHm) then the energy consumption is higher than that of EDF-LRHd. Moreover, keeping the thermostat setting for the maximum data center utilization for a given set of jobs (as in EDF-LRHm) is beneficial than setting it for 100% utilization (as in EDF-LRH).

The SP-EIR of the HTS and LRH algorithms are plotted against the utilization in Figures 7 and 8 for the idle on and idle off case, respectively. With increase in the utilization the SP-EIR increases, reaches a maximum and then again goes down. This behavior is expected from the formulation where at 0% and 100% utilization both the algorithm will be optimal.

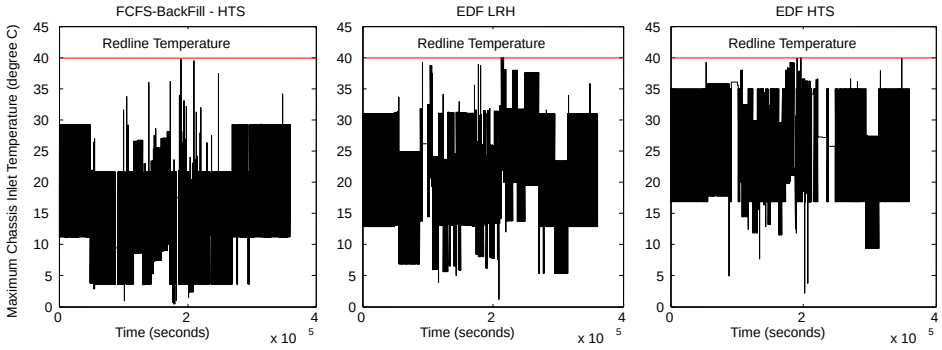

Fig. 10. Maximum chassis inlet temperature for 80 % utilization under dynamic cooling model for different energy management policies. FCFS-Backfill-HTS, EDF-LRH and EDF-HTS are compared in this figure to represent the effect of only cooling aware, only job and thermal aware and job thermal and cooling aware scheduling mechanisms. It can be seen that EDF-HTS runs the data center hottest yet avoiding redline violations and hence achieves the most energy efficiency

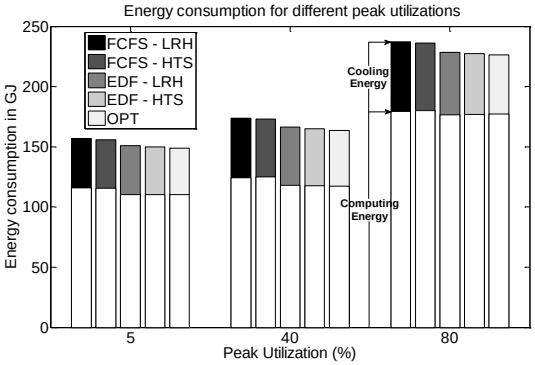

Fig. 11. Total energy consumption comparison between EDF-LRH, EDF-HTS and Lower bound on Optimal solution at different data center peak utilizations for idle chassis kept on

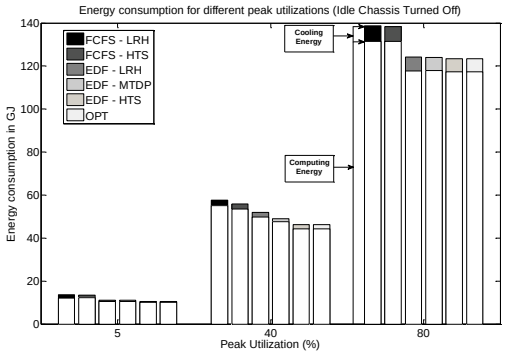

Fig. 12. Total energy consumption comparison between EDF-LRH, EDF-HTS and Lower bound on Optimal solution at different data center peak utilizations for idle chassis turned off case

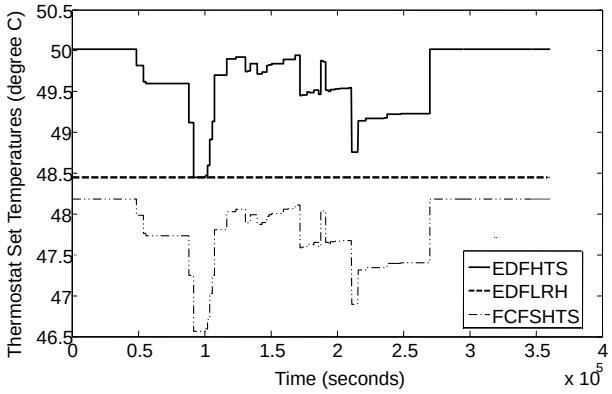

Fig. 13. CRAC thermostat set temperatures with respect to time when different management policies are employed (peak utilization = 80%).

FCFS based algorithms perform poorer than the EDF based algorithms with respect to the total energy consumption, because of the temporal spreading of the jobs in EDF [12]. Tables 3 and 4 give the comparison of the FCFS based approaches with EDF-HTS for the idle on and idle off cases, respectively. Figures 11 and 12 show the data center energy consumption for the different algorithms along with the break up of the computing energy and cooling energy. Future research is needed to investigate the energy savings when dynamic CRAC control is integrated with server consolidation.

On the other hand, FCFS-Backfill based algorithms can achieve higher throughput per unit energy over the EDF based algorithms. This is mainly because of very high job throughput since jobs are not temporally spread in FCFS-Backfill. Figure 9 shows the variation in throughput per unit energy for FCFS-Backfill based algorithms when the idle servers are turned off. The energy consumption increases considerably with increase in utilization; thus causing reduction in the throughput per unit energy for higher utilization.

As shown in Figure 9, FCFS-Backfill-HTS achieves the highest throughput per unit energy for all utilizations and can increase the throughput per unit energy by up to 5.56% over the FCFS-Backfill-MTDP algorithm. When compared to FCFS-Backfill-LRH, FCFS-Backfill-HTS can achieve up to 6.89% higher throughput per unit energy consumption.

Further, the relative benefits of incorporating cooling awareness, job awareness and thermal awareness can be obtained from the simulations. To this effect we choose: 1) the FCFS-Backfill-HTS algorithm, which just incorporates cooling awareness to the existing FCFS-Backfill scheduling algorithm, 2) the EDF-LRH algorithm, which incorporates thermal awareness and ensures that maximum number of jobs meet their respective deadline, and 3) the EDF-HTS algorithm, which integrates all three aspects. Figure 10 shows the maximum chassis inlet temperature as a function of time for the three algorithms. It can be seen that the data center runs hotter for EDF-LRH than FCFS-Backfill-HTS and the hottest for EDF-HTS. This shows that EDF-HTS benefits from ranking the servers according to their threshold temperature requirements. This benefit is not utilized by EDF-LRH which was designed based on constant cooling assumptions. Figure 13 shows the CRAC threshold temperatures set by the three algorithms. From the figure it is clearly seen that EDF-LRH misses out on adjusting the CRAC threshold. Further, higher CRAC threshold set temperatures can be achieved if the slack in jobs are utilized to spread them apart in time as seen from the comparison between FCFS-Backfill-HTS and EDF-HTS.

7. Related Work

In this section, the work done in this paper is compared with existing approaches for data center energy efficiency. Reduction in the data center energy consumption has been tackled by both academia and industry from different domains. For example, mechanical engineers focus on modification in physical layout of data centers [11], liquid cooling (`http://www.42u.com/liquid-cooling-article.htm`) and introduction of chiller doors (`http://www.42u.com/42u-rack-cooling.htm`); electrical engineers focus on designing low power servers and

introducing new power states in processors; and computer scientists focus on energy efficient spatial scheduling of parallel workload on heterogeneous servers. This paper focused on cyber-physical decision making through: i) spatio-temporal job scheduling to reduce cooling demands; and ii) dynamic variations in the CRAC thermostat settings.

Research in the area of energy-efficient job scheduling can be divided into three categories: 1) *thermally oblivious*, in which, job scheduling decisions are not aware of the heating effects in the data center; 2) *thermal-aware*, which considers the thermal behavior of the data center but does not consider its cooling behavior; and 3) *cooling-aware*, which considers **both** the thermal behavior of the data center and also considers its cooling behavior.

Thermally oblivious management strategies include work in [2] where Ranganathan et al propose a load balancing scheme based on the power consumption of machines so as to minimize overall energy consumption of the data center while meeting SLAs. In [23] and [24], duty cycling of servers (turning off and on) is proposed to achieve energy efficiency. A unification of load balancing and duty cycling of servers is further proposed in [3]. Researchers have also proposed the use of dynamic voltage scaling of servers to reduce energy [25] while maintaining SLA. In this regard, ideas from game theory has been employed to account for the trade off in energy and performance [26, 27]. However, all these works do not consider the thermal effects and the associated cooling energy expenditure in the data center.

In [28], the thermal behavior of the data center is taken into account for job placement algorithms. In this work, the thermal effects in the data center are abstracted to heat recirculation between servers that cause local hot spots. Job placement algorithms are then proposed to avoid servers in the hot spots. In [29], a similar approach is taken to avoid hot spots while placing jobs. In [30], an RC thermal model is used to characterize the thermal effects of individual server on the environment. However, it ignores the heat recirculation from other servers which may result in under-estimation of temperature rise. Although these techniques are aware of the thermal effects in the data center, they do not consider its cooling behavior.

Integration of cooling control (controlling cooling parameters such as supply temperature and set points of the CRAC) with energy management in data centers is of recent focus. In [12], Mukherjee et al propose job scheduling algorithms that consider the thermal effects in a data center and try to schedule as well as place jobs so as to reduce the cooling demands. Similar efforts has been undertaken by Zahra et al [8] for the Internet data centers where active servers are so chosen that the supply temperature requirements are increased. This in turn causes reduction in cooling energy. Parolini et al have also proposed an integrated approach to job scheduling and cooling control in [6]. Data center is modeled as a Markov decision process where the actions taken to move from one thermal state to another is controlled by the CRAC thermostat settings. However, all these techniques do not model the dynamic behavior of the CRAC. The HTS algorithm proposed in this paper considered the CRAC dynamic behavior in performing cooling-aware, thermal-aware, and energy-efficient job management. Table 5 shows this classification and highlights the novelty in HTS.

Further, given these three types of scheduling algorithms, [9] gives a mechanism to decide on which algorithm to

Table 5
Classification of previous research on data center energy efficiency

Techniques for Energy Efficiency	Thermally Oblivious	Thermal Aware	Cooling Aware	Cooling Model
Ensemble level power management [2], Load balancing [3, 23, 24], CPUMISER [25], Game theoretic approach [26, 27]	✔			-
MinHR, XInt, Minimax Optimization [31], Temperature based allocation without recirculation [29, 30]		✔		-
SCINT and LRH [12], TASP and TAWD [8], and MDP based scheduling [6]		✔	✔	Constant
HTS		✔	✔	Dynamic

use for different management requirements such as , job, thermal or cooling management. Recent works have further, evaluated the energy efficiency of these algorithms through simulations in presence of energy proportional servers in the data centers [10] under constant cooling environment. However, a theoretical analysis on how energy inefficient these algorithms can get in presence of heat recirculation and dynamic cooling behavior, has not been performed.

In this regard, this paper also provides a theoretical framework to mathematically evaluate the performance of energy management algorithms in terms of their inefficiency in energy consumption under heat recirculation and dynamic cooling behavior. It defines a new metric, *SP-EIR*, which determines the degree of energy inefficiency of a given algorithm with respect to the best possible cooling and thermal aware job placement (optimal). Previous research in algorithm performance evaluation have mostly considered energy efficiency of thermal and cooling unaware algorithms. Lin et al [32] have performed competitive bound analysis of online algorithms for dynamically selecting servers to turn off so as to save energy. However, such an analysis is only for energy aware scheduling algorithms and does not consider the thermal effects in the data center or the cooling behavior. Further, for HPC data centers the workload is often known before hand so a competitive bound analysis for online job scheduling is not applicable. Rather an offline analysis using the *SP-EIR* metric, as in this paper, is more relevant. Offline approximation bound analysis of thermal aware job scheduling algorithms for homogeneous data centers have been considered [33]. However, such an analysis again falls short of considering cooling behavior and thermal profiles of heterogeneous data center. Qinghui et al [34] proposes a theoretical framework for the development and evaluation of scheduling algorithms for distributed cyber physical systems with data center as a specific example. Although it incorporates the thermal profiles of the data center in the algorithm development, cooling behavior is still ignored. Further, Gupta et al [35] propose a theoretical formulation of the scheduling for energy efficiency problem in cyber-physical systems with data center as a specific case study. It incorporates the heat recirculation and cooling behavior into the formulation but does not give any metric for algorithm performance evaluation.

8. Conclusions & Future Work

Integration of cooling and thermal awareness for spatial scheduling of jobs in HPC data centers is discussed in the paper. First, a cooling and thermal aware job placement algorithm HTS is proposed, which ranks the servers in decreasing order of their CRAC threshold temperature requirements and then adjusts the CRAC threshold temperature to the highest possible value without violating the redline constraint. Second, an energy inefficiency analysis of the existing spatial scheduling algorithms under constant and linear cooling modes is performed, where the *SP-EIR* metric is introduces. This metric shows how much energy savings a spatial scheduling algorithm can have with respect to the best known cooling and thermal aware scheduling algorithm. In this regard, constant cooling is mathematically shown to be a special case of dynamic cooling. Further, the *SP-EIR* is shown to depend on the high threshold temperature, power extraction in different modes and switching time of the CRAC. It is concluded that any algorithm that fails to control the threshold temperature misses out on opportunities to save cooling energy. Third, the proposed HTS spatial scheduling algorithm is augmented with EDF temporal scheduling. Extensive simulations are performed to compare the EDF-HTS spatio-temporal scheduling against other common approaches such as FCFS-Backfill-FF, FCFS-Backfill-LRH, FCFS-Backfill-HTS, and EDF-LRH. Further, in simulations, potential for energy efficiency by turning off servers has been studied. In this regard EDF-HTS has been compared with EDF-MTDP, which performs intelligent server consolidation.

Simulation results show HTS can reduce SP-EIR by up to 15% over the most energy-efficient scheduling algorithm, LRH. HTS, when augmented with idle server turn-offs, can further achieve up to 9% energy-savings compared to the MTDP, which performs spatial scheduling based on thermal-aware server consolidation. HTS can achieve up to 5.56% higher throughput per unit energy consumption than MTDP, when both HTS and MTDP are used with FCFS-Backfill, a widely used temporal scheduling algorithm in data centers.

The cooling-aware spatial job scheduling in HTS (and integrating it with cooling management) is developed as part of a *BlueTool* research infrastructure funded by the National Science Foundation (NSF) [6] . A miniature data center test-bed is being developed to develop, test and evaluate energy reduction policies in the data centers. Current and future work in this regard are geared towards coordinated job scheduling and cooling management that can capture the behavior of novel future cooling units designed to scavenge energy from alternate sources (e.g. solar power).

Many enterprise data centers further host Internet services (e.g. search engines, data retrieval) where the applications are mostly *short-duration transactions* (unlike the HPC jobs considered in this paper, which are normally long running). In such transaction based data centers, job management usually involves distribution of large number of workloads depending on their arrival rate. HTS can be adapted for such data centers in two ways: i) provisioning of the servers based on the HTS server ranking mechanism; and ii) cooling-aware workload distribution similar to the

[6] http://www.nsf.gov/awardsearch/showAward.do?AwardNumber=0855277

spatial scheduling for HPC data centers. Indeed, we have been working on thermal-aware workload distribution in transaction-based data centers without any dynamic setting of the CRAC thermostat.

Dynamically setting the thermostat may not have instantaneous impact and may take effect with a delay, principally because of saturating the heat capacity of the agent in the internal cooling cycle (several seconds, perhaps minutes). As such, in many cases (especially in case of transaction based data centers with frequent changes in the thermostat requirements), it may be important to make sure that any dynamic update of the CRAC thermostat does not become stale by the time it takes effect. Future work in this regard need to perform *management decision making* that employs proper energy-management policies depending on the state of the data center while meeting the SLAs and server redlines.

Further, the results presented in this paper are simulation based since actual experiments in physical data centers may require shutting down servers, which in turn may affect SLAs. In an ongoing project named BlueTool' [36] (`http://impact.asu.edu/BlueTool/wiki/index.php/Main_Page`) funded by NSF infrastructure grant (CNS#0855277) Impact Lab is developing a data center testbed. Such a testbed can be used to evaluate the proposed algorithms in real scenarios and is considered as an important future work.

References

[1] U. E. P. Agency, "Report to congress on server and data center energy efficiency public law 109-431," ENERGY STAR Program, 2007.

[2] P. Ranganathan, P. Leech, D. Irwin, and J. Chase, "Ensemble-level power management for dense blade servers," in *IEEE Proceedings of the 33rd International Symposium on Computer Architecture (ISCA'06)*, Boston, MA, May 2006, pp. 66–77.

[3] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Load balancing and unbalancing for power and performance in cluster-based systems," in *In Workshop on Compilers and Operating Systems for Low Power*, 2001.

[4] P. Bohrer, E. N. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, R. Rajamony, and L. C. Mcdowell, "The case for power management in web servers," 2002.

[5] J. Moore, J. Chase, P. Ranganathan, and R. Sharma, "Making scheduling "cool": Temperature-aware resource assignment in data centers," in *2005 Usenix Annual Technical Conference*, April 2005.

[6] L. Parolini, B. Sinopoli, and B. H. Krogh, "A unified thermal-computational approach to data center energy management," in *Fourth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*, San Francisco, California, USA, April 2009.

[7] E. Pakbaznia and M. Pedram, "Minimizing data center cooling and server power costs," in *ISLPED '09: Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design*. New York, NY, USA: ACM, 2009, pp. 145–150.

[8] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta., "Thermal aware server provisioning and workload distribution for internet data centers," in *ACM International Symposium on High Performance Distributed Computing (HPDC10)*, Jun. 2010.

[9] T. Mukherjee, A. Banerjee, G. Varsamopoulos, and S. K. S. Gupta, "Model-driven co-ordinated management of data centers," *(Elsevier) Computer Networks, Special Issue on Managing Emerging Computing Environments, accepted (2010)*, 2010.

[10] G. Varsamopoulos, Z. Abbasi, and S. K. S. Gupta, "Trends and effects of energy proportionality on server provisioning in data centers," in *International Conference on High performance Computing Conference (HiPC2010)*, Dec. 2010.

[11] M. Stansberry, "Hot-aisle/cold-aisle containment and plenum strategies go big-time," http://searchdatacenter.techtarget.com/news/article/0,289142,sid80_gci1320452,00.html, 2008.

[12] T. Mukherjee, A. Banerjee, G. Varsamopoulos, S. K. S. Gupta, and S. Rungta, "Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers?" *Computer Networks*, June 2009. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2009.06.008

[13] E. Krevat, J. G. Casta nos, and J. E. Moreira, "Job scheduling for the bluegene/l system," in *JSSPP '02: Revised Papers from the 8th International Workshop on Job Scheduling Strategies for Parallel Processing*. London, UK: Springer-Verlag, 2002, pp. 38–54.

[14] G. Varsamopoulos, A. Banerjee, and S. Gupta, "Energy efficiency of thermal-aware job scheduling algorithms under various cooling models," in *IC^3 '2009*, Noida, India, Aug.

[15] S. R. LaPlante, N. Aubry, L. Rosa, P. Levesque, B. S. Aboumrad, D. Porter, C. Cavanaugh, and J. Johnston, "Liquid cooling of a high density computer cluster," [online], 2006. [Online]. Available: \irl{http://www.electronics-cooling.com/articles/2006/2006_nov_a1.php}

[16] M. J. Moran and H. N. Shapiro, *Fundamentals of Engineering Thermodynamics, 6th Edition*. Wiley, 2007.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 2nd ed. Cambridge, London: McGraw-Hill Book Company, 2001, 1. editon 1993.
[18] "Moab grid suite of ClusterResources Inc." http://www.clusterresources.com/. [Online]. Available: http://www.clusterresources.com/
[19] A. Flomerics Ltd, "Flovent version 2.1," Hampton Court, Surrey, KT8 9HH, England, 1999. [Online]. Available: http://www.flomerics.com/
[20] Cyber Switching, "DUALCOM user manual," [online], http://www.cyberswitching.com/pdf/DualcomManual.pdf.
[21] T. Mukherjee, G. Varsamopoulos, S. K. S. Gupta, and S. Rungta, "Measurement-based power profiling of data center equipment," in *Proc. IEEE Conference on Clustered and Grid Computing (Cluster 2007), Workshop on Green Computing (GreenCom'07)*, Austin, TX, Sep. 2007.
[22] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-Efficient, Thermal-Aware Task Scheduling for Homogeneous, High Performance Computing Data Centers: A Cyber-Physical Approach," *IEEE Transactions on Parallel and Distributed Computing (TPDS)*.
[23] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Dynamic cluster reconfiguration for power and performance," pp. 75–93, 2003.
[24] J. Hikita, A. Hirano, and H. Nakashima, "Saving 200kw and $200 k/year by power-aware job/machine scheduling," in *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, April 2008, pp. 1–8.
[25] R. Ge, X. Feng, W. chun Feng, and K. Cameron, "Cpu miser: A performance-directed, run-time system for power-aware clusters," in *Parallel Processing, 2007. ICPP 2007. International Conference on*, 2007, p. 18.
[26] S. Khan and I. Ahmad, "A cooperative game theoretical technique for joint optimization of energy consumption and response time in computational grids," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 346 –360, 2009.
[27] R. Subrata, A. Y. Zomaya, and B. Landfeldt, "Cooperative power-aware scheduling in grid computing environments," *Journal of Parallel and Distributed Computing*, vol. 70, no. 2, pp. 84 – 91, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B6WKJ-4XCJ4MF-1/2/debdc2f8%0e1ffeba4ea6a9a2e2d25112
[28] Q. Tang, T. Mukherjee, S. Gupta, and P. Cayton, "Sensor-based fast thermal evaluation model for energy efficient high-performance datacenters," in *Intelligent Sensing and Information Processing, 2006. ICISIP 2006. Fourth International Conference on*, 15 2006-Dec. 18 2006, pp. 203–208.
[29] C. Bash and G. Forman, "HPL-2007-62 cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center," HP Laboratories Palo Alto, Tech. Rep. HPL-2007-62, Aug. 2007.
[30] *Towards Thermal Aware Workload Scheduling in a Data Center*. Kao-Hsiung, Taiwan: IEEE, 12/2009 2009. [Online]. Available: http://cyberaide.googlecode.com/svn/trunk/papers/09-greenit-ispan1/vonL%aszewski-ispan1.pdf
[31] Q. Tang et al, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *IEEE TPDS*, vol. 19, no. 11, pp. 1458–1472, 2008.
[32] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Proc. IEEE INFOCOM*, Shanghai, China, 10-15 Apr 2011.
[33] H. Aissi, C. Bazgan, and D. Vanderpooten, "General approximation schemes for min-max (regret) versions of some (pseudo-)polynomial problems," *Discrete Optimization*, vol. 7, no. 3, pp. 136–148, 2010.
[34] Q. Tang, G. Varsamopoulos, and S. K. S. Gupta., "A unified methodology for scheduling in distributed cyber-physical systems," *ACM Transactions on Embedded Computing Systems*, 2010, special Issue on the Verification of Cyber-Physical Software Systems.
[35] S. K. S. Gupta, T. Mukherjee, G. Varsamopoulos, and A. Banerjee, "Research directions in energy-sustainable cyber-physical systems," *(Elsevier) Sustainable Computing, invited paper accepted for publication*, 2010.
[36] S. Gupta, G. Vasamopoulos, A. Haywood, P. Phelan, and T. Mukherjee, *BlueTool: Using a computing systems research infrastructure tool to design and test green and sustainable data centers*. Handbook of energy aware and green computing, publisher Chapman and Hall/CRC press, Taylor and Fracis group LLC, To Appear, 2011.