

Thermal aware workload placement with task-temperature profiles in a data center

Lizhe Wang · Samee U. Khan · Jai Dayal

© Springer Science+Business Media, LLC 2011

Abstract Data centers now play an important role in modern IT infrastructures. Related research shows that the energy consumption for data center cooling systems has recently increased significantly. There is also strong evidence to show that high temperatures in a data center will lead to higher hardware failure rates, and thus an increase in maintenance costs. This paper devotes itself in the field of thermal aware workload placement for data centers. In this paper, we propose an analytical model, which describes data center resources with heat transfer properties and workloads with thermal features. Then two thermal aware task scheduling algorithms, TASA and TASA-B, are presented which aim to reduce temperatures and cooling system power consumption in a data center. A simulation study is carried out to evaluate the performance of the proposed algorithms. Simulation results show that our algorithms can significantly reduce temperatures in data centers by introducing endurable decline in system performance.

Keywords Thermal aware · Data center · Green computing · Task scheduling

L. Wang (✉)
Pervasive Technology Institute, Indiana University, 2719 E. 10th St., Bloomington, IN 47408, USA
e-mail: Lizhe.Wang@gmail.com

S.U. Khan
Department of Electrical and Computer Engineering, North Dakota State University, Fargo,
ND 58108-6050, USA

J. Dayal
School of Computer Science, School of Computing, George Institute of Technology, 266 First Drive,
Atlanta, GA 30332-0765, USA
e-mail: dayalsoap@gmail.com

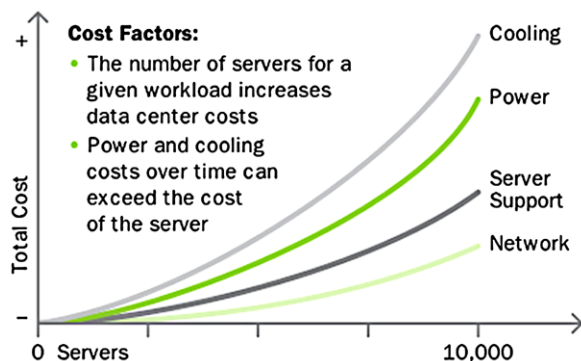
1 Introduction

Electricity usage is the most expensive portion of a data center's operational costs. In fact, the US Environmental Protection Agency (EPA) reported that 61 billion KWh, 1.5% of US electricity consumption, is used for data center computing [3]. Additionally, the energy consumption in data centers doubled between 2000 and 2006. Continuing this trend, the EPA estimates that the energy usage will double again by 2011. As shown in Fig. 1, it is reported that power and cooling cost is the most dominant cost in data centers [9, 14]. The cooling costs can be up to 50% of the total energy cost [22]. Even with more efficient cooling technologies in IBM's BlueGene/L and TACC's Ranger, cooling cost still remains a significant portion of the total energy cost for these supercomputers [15]. It is also noted that the life of a computer system is directly related to its operating temperature. Based on Arrhenius time-to-fail model [10], every 10°C increase of temperature leads to a doubling of the system failure rate. Hence, it is important to decrease data center operational temperatures for reducing energy cost and increasing reliability of compute resources.

Consequently, resource management with thermal considerations is important for data center operations. Some research has developed elaborate models and algorithms for data center resource management with computational fluid dynamics (CFD) techniques [5, 6]. Other work [17, 26], however, argues that the CFD based model is too complex and the computation is time consuming. It is thus not suitable for online scheduling. This has led to the development of some less complex online scheduling algorithms. Sensor-based fast thermal evaluation model [25, 27], Generic Algorithm and Quadratic Programming [18, 26], and the Weatherman—an automated online predictive thermal mapping [16] are a few examples.

In this paper, we propose a Thermal Aware Scheduling Algorithm (TASA) and a Thermal Aware Scheduling Algorithm with Backfilling (TASA-B) for data center task scheduling. The TASA schedules data workloads via tasks' temperature profiles to reduce temperatures of a data center. The TASA-B extends the TASA by allowing the backfilling of tasks under thermal constraints. Our work differs from the above systems in that we have developed some elaborate heat transfer models for data centers. Our model, therefore, is less complex than the CFD models and can be used

Fig. 1 Data center cost factors [9]



for online scheduling in data centers. It still can provide enough accurate description of data center thermal maps than [18, 26]. In detail, we study a temperature based workload model and a thermal based data center model. This paper then defines the thermal aware workload scheduling problem for data centers and presents the TASA and TASA-B for data center workloads. We use simulations to evaluate the TASA and TASA-B and discuss the trade-off between throughput, cooling cost, and other performance metrics.

Our unique contribution is listed as follows:

- We propose a general framework for thermal aware resource management for data centers. Our framework is not bound to any specific model, such as the RC-thermal model, the CFD model, or a task-temperature profile.
- We develop a new methodology for thermal aware workload scheduling: “predict temperature increase → distribute workloads → update thermal information.” The online task-temperature profile and the RC-model are used to predict resource temperature increase. New heuristics of TASA and TASA-B are developed and evaluated, in terms of performance loss, temperature reduced, and power saved.
- We implement a simulation study on the TASA and TASA-B and develop a discussion of their impact on the environment, for example, the CO₂ emission.

The rest of this paper is organized as follows. Section 2 introduces the related work and background of thermal aware workload scheduling in data centers. Section 3 presents mathematical models for data center resources and workloads. In Sect. 4, we present our thermal aware scheduling framework and formally defines the research issue of thermal aware scheduling in a data center. It flows the description of TASA and TASA-B in Sects. 5 and 6. We evaluate the algorithm with a simulation in Sect. 7. The paper is finally concluded in Sect. 8.

2 Related work and background

2.1 Research on thermal management for a data center

There are some existing research on thermal management in a data center. The most elaborate thermal aware schedule algorithms for tasks in data centers are with computational fluid dynamics (CFD) models [5, 6]. Some research [17, 26] declares that the CFD based model is too complex and is not suitable for online scheduling in a data center.

Weatherman [16] is a proof-of-concept implementation for automated modeling of data center thermal topology. It provides an online approach to predicting the heat profile for a given data center configuration with Artificial Neural Network (ANN). In our paper, we use task-temperature profiles for temperature prediction and thermal aware scheduling. The task-temperature profiling and calculation are less complex than the ANN based temperature prediction, thus suitable for on-line scheduling. Based on the temperature prediction, we develop a

thermal aware scheduling algorithm, which is not included in the implementation of [16].

Moore et al. in [17] develop a temperature-aware workload placement in data centers based on thermodynamic formulations, job power profiles, and information about inlet and outlet temperatures. We noticed that the power profiles in [17] are not easy to measure precisely for a large number types of jobs. It is also argued in [26] that the thermal model and power model in [17] are not accurate for data centers.

Lee et al. [13] propose a proactive control approach that jointly optimizes the air conditioner compressor duty cycle and fan speed to prevent heat imbalance to minimize the cost of cooling in data centers.

ASU researchers developed a software/hardware architecture for thermal management in data centers [18, 25, 26]. ASU's work solves the research problem of minimizing the peak inlet temperature within a data center through task assignment (MPIT-TA), consequently leading to minimal cooling requirement. Our implementation in this paper is different from ASU's work in that (1) we introduce task-temperature profile for task sorting, and (2) we do not introduce power profiles, cooling system, and inlet and outlet temperatures in our model. Therefore, our work is a lightweight implementation and suitable for online scheduling in a data center.

If we know how much temperature degrees increase to execute a task, it would be easier to develop a task scheduling algorithm to reduce data center temperature. However, this is not easy work. None of the above work directly models the relationship between task execution and ambient temperature, and use this relationship for task scheduling. The reason is that the task-temperature relationship modeling requires long time profiling and complex calculation, which is difficult or expensive to achieve. In this paper, we traced the workload of a real data center, and also use the RC model for task-temperature relationship calculation. With the calculated task-temperature profile, we developed scheduling algorithms to distribute incoming tasks to compute nodes based on the ambient temperature prediction.

2.2 Data center organization

The racks in a typical data center with a standard cooling layout based on under-floor cold air distribution are back-to-back and laid out in rows on a raised floor over a shared plenum. Modular computer room air conditioning (CRAC) units along the walls circulate warm air from the machine room over cooling coils, and direct the cooled air into the shared plenum. As shown in Fig. 2, the cooled air enters the machine room through floor vent tiles in alternating aisles between the rows of racks. Aisles containing vent tiles are cool aisles; equipment in the racks is oriented so their intake draws inlet air from cool aisles. Aisles without vent tiles are hot aisles, providing access to the exhaust air and, typically rear panels of the equipment [23].

Thermal imbalances interfere with efficient cooling operation. Hot spots create a risk of redlining servers by exceeding the specified maximum inlet air temperature, damaging electronic components, and causing them to fail prematurely. Nonuniform

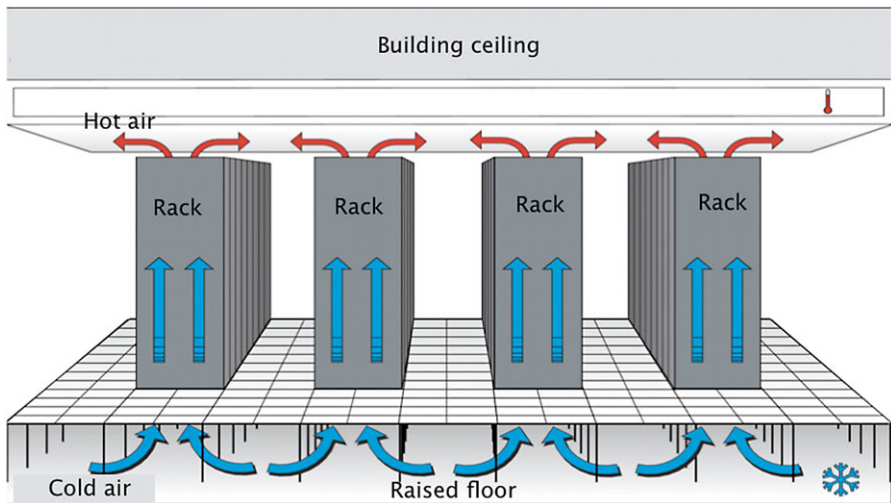


Fig. 2 Data center cooling system

equipment loads in the data center cause some areas to heat more than others, while irregular air flows cause some areas to cool less than others. The mixing of hot and cold air in high heat density data centers leads to complex airflow patterns that create hot spots. Therefore, objectives of thermal aware workload scheduling are to reduce both the maximum temperature for all compute nodes and the imbalance of the thermal distribution in a data center. In a data center, the thermal distribution and computer node temperatures can be obtained by deploying ambient temperature sensors, on-board sensors [25, 27], and with software management architectures like Data Center Observatory [12], Mercury and Freon [11], and LiquidN2 and C-Oracle[20].

2.3 Task-temperature profiling

Given a compute processor and a steady ambient temperature, a task-temperature profile is the temperature increase along with the task's execution. It has been observed that different types of computing tasks generate different amounts of heat, therefore, resulting with distinct task-temperature profiles [28].

Task-temperature profiles can be obtained by using some profiling tools. Figure 3 shows the task-temperature profile of a compute node in the Center for Computational Research (CCR) of State University of New York at Buffalo. The X -axis is the time and the Y -axis gives two values: the workload (task execution time in the passed hour) and the temperature for a compute node. Figure 3 clearly indicates the task and temperature correlation: normally as computation loads in term of task CPU time augment, compute node temperatures increases incidentally. Therefore, it is both constructive and realistic to assume that the knowledge of task-temperature profile is available based on the discussion [7, 31] that task-temperature can be well approximated using appropriate prediction tools and methods.

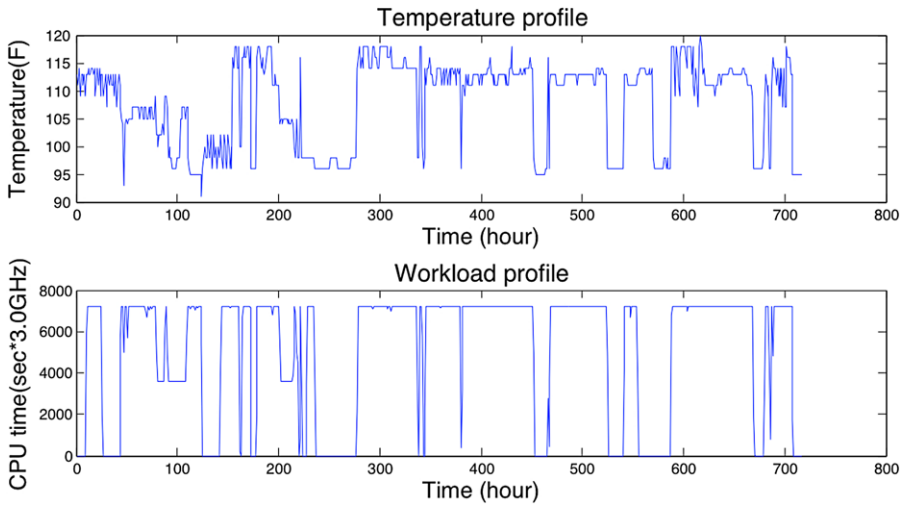


Fig. 3 Task-temperature profile of a compute node in buffalo data center

3 System models and problem definition

3.1 Compute resource model

This section presents formal models of data centers and workloads and a thermal aware scheduling algorithm, which allocate compute resources in a data center for incoming workloads with the objective of reducing temperature in the data center.

A data center *DataCenter* is modeled as:

$$DataCenter = \{Node, TherMap\} \quad (1)$$

where *Node* is a set of compute nodes, *TherMap* is the thermal map of a data center.

A thermal map of a data center describes the ambient temperature field in a 3-dimensional space. The temperature field in a data center can be defined as follows:

$$TherMap = Temp((x, y, z), t) \quad (2)$$

It means that the ambient temperature in a data center is a variable with its space location (x, y, z) and time t .

We consider a homogeneous data center: all compute nodes have identical hardware and software configurations. Suppose that a data center contains I compute nodes as shown in Fig. 4:

$$Node = \{node_i | 1 \leq i \leq I\} \quad (3)$$

The i th compute node is described as follows:

$$node_i = ((x, y, z), t^a, Temp(t)) \quad (4)$$

Fig. 4 Layout of compute nodes and ambient environment

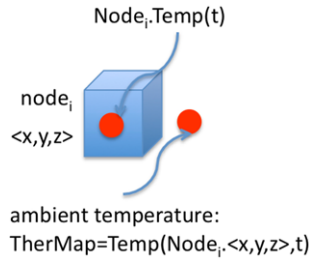
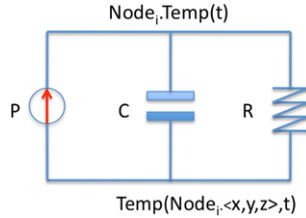


Fig. 5 RC-thermal model



$\langle x, y, z \rangle$ is $node_i$'s location in a 3-dimensional space. t^a is the time when $node_i$ is available for job execution. $Temp(t)$ is the temperature of $node_i$, t is time.

Figure 4 shows the layout compute nodes and their ambient environment. For simplicity, we assume the compute nodes and their ambient environment shares the same 3-dimensional position $\langle x, y, z \rangle$.

The process of heat transfer of $node_i$ is described with a RC-thermal model [21, 24, 32]. As shown in Fig. 5, P denotes the power consumption of compute node at current time t , C is the thermal capacitance of the compute node, R denotes the thermal resistance, and $Temp(node_i \cdot \langle x, y, z \rangle, t)$ represents the ambient temperature of $node_i$ in the thermal map. Therefore, the heat transfer between a compute node and its ambient environment is described in (5) (also shown in Fig. 6).

$$node_i \cdot Temp(t) = RC \times \frac{\partial node_i \cdot Temp(t)}{\partial t} + Temp(node_i \cdot \langle x, y, z \rangle, t) - RP \quad (5)$$

It is supposed that an initial die temperature of a compute node at time 0 is $node_i \cdot Temp(0)$, P and $Temp(node_i \cdot \langle x, y, z \rangle, t)$ are constant during the period $[0, t]$ (this can be true when calculating for a short time period). Then the compute node temperature $Node_i \cdot Temp(t)$ is calculated as (6).

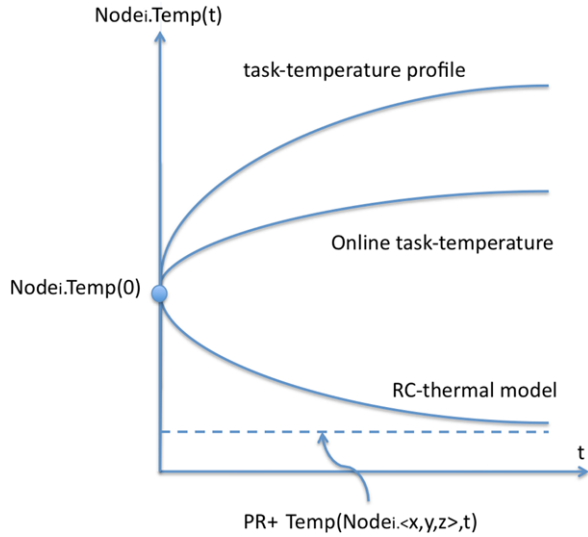
$$node_i \cdot Temp(t) = PR + Temp(node_i \cdot \langle x, y, z \rangle, 0) + (node_i \cdot Temp(0) - PR - Temp(node_i \cdot \langle x, y, z \rangle, 0)) \times e^{-\frac{t}{RC}} \quad (6)$$

3.2 Workload model

Data center workloads are modeled as a set of jobs:

$$Job = \{job_j | 1 \leq j \leq J\} \quad (7)$$

Fig. 6 Task-temperature profiles



J is the total number of incoming jobs. job_j is an incoming job, which is described as follows:

$$job_j = (p, t^{arrive}, t^{start}, t^{req}, \Delta Temp(t)) \tag{8}$$

where, p is the required number of compute nodes for job_j , t^{arrive} is the arrival time of job_j , t^{start} is the starting time of job_j , t^{req} is the required execution time of job_j , $\Delta Temp(t)$ is the task-temperature profile of job_j on compute nodes of a data center.

3.3 Online task temperature calculation

When a job job_j runs on certain compute node $node_i$, the job execution will increase the node's temperature $job_j \cdot \Delta Temp(t)$. In the mean time, the node also disseminates heat to ambient environment, which is calculated by (6). The online node temperature is the real temperature profile of the node. Thus, the online task temperature profile of $node_i$ is calculated as (9).

$$\begin{aligned}
 node_i \cdot Temp(t) &= node_i \cdot Temp(0) + job_j \cdot \Delta Temp(t) \\
 &- \{ PR + Temp(node_i \cdot \langle x, y, z \rangle, 0) + (node_i \cdot Temp(0) - PR \\
 &- Temp(node_i \cdot \langle x, y, z \rangle, 0)) \times e^{-\frac{t}{\kappa C}} \} \tag{9}
 \end{aligned}$$

3.4 Research issue definition

Based on the above discussion, a job schedule is a map from a job job_j to certain work node $node_i$ with starting time $job_j \cdot start$:

$$schedule_j : job_j \rightarrow (node_i, job_j \cdot t^{start}) \tag{10}$$

A workload schedule *Schedule* is a set of job schedules $\{schedule_j | job_j \in Job\}$ for all jobs in the workload:

$$Schedule = \{schedule_j | job_j \in Job\} \quad (11)$$

We define the workload starting time T_0 and finished time T_∞ as follows:

$$T_\infty = \max_{1 \leq j \leq J} \{job_j \cdot t^{\text{start}} + job_j \cdot t^{\text{req}}\} \quad (12)$$

$$T_0 = \min_{1 \leq j \leq J} \{job_j \cdot t^{\text{arrive}}\} \quad (13)$$

Then the workload response time T_{response} is calculated as follows:

$$T_{\text{response}} = T_\infty - T_0 \quad (14)$$

Assuming that the specified maximum inlet air temperature in a data center is $TEMP_{\text{max}}$, thermal aware workload scheduling in a data center could be defined as follows: given a workload set *Job* and a data center *DataCenter*, find an optimal workload schedule, *Schedule*, which minimizes T_{response} of the workload *Job*:

$$\min T_{\text{response}} \quad (15)$$

subject to:

$$\max_{1 \leq i \leq I} \{node_i \cdot Temp(t) | T_0 \leq t \leq T_\infty\} \leq Temp_{\text{max}} \quad (16)$$

It has been discussed in our previous work [30] that research issue of (15) is an NP-hard problem. To solve this research issue, in the next section, Sect. 4, we introduce a general concept framework for thermal aware workload scheduling. Sections 5 and 6 present our thermal aware scheduling algorithms, TASA and TASA-B.

4 Thermal aware workload scheduling framework

This section introduces the concept of the framework for thermal aware workload scheduling as shown in Fig. 7. The online task-temperature profile is the temperature profile obtained by running a task on a compute node and the ambient environment temperature. The online task-temperature profile is calculated with the input of the task-temperature profile, the RC-thermal model, and the thermal map. A thermal map is the current temperature field in the data center space. A data center monitoring system, for example, either a temperature sensor based monitoring service [26, 27] or by using compute fluid dynamics software [5, 6], can generate a thermal map for a data center. An alternative to cooling system operating patterns can change the thermal map of a data center.

Workloads of a data center are characterized with task compute requirements: number of required compute nodes, execution time, and a task-temperature profile.

With a data center resource model and a workload model, a thermal-aware scheduler can place workloads on the proper compute nodes with the objective to reduce

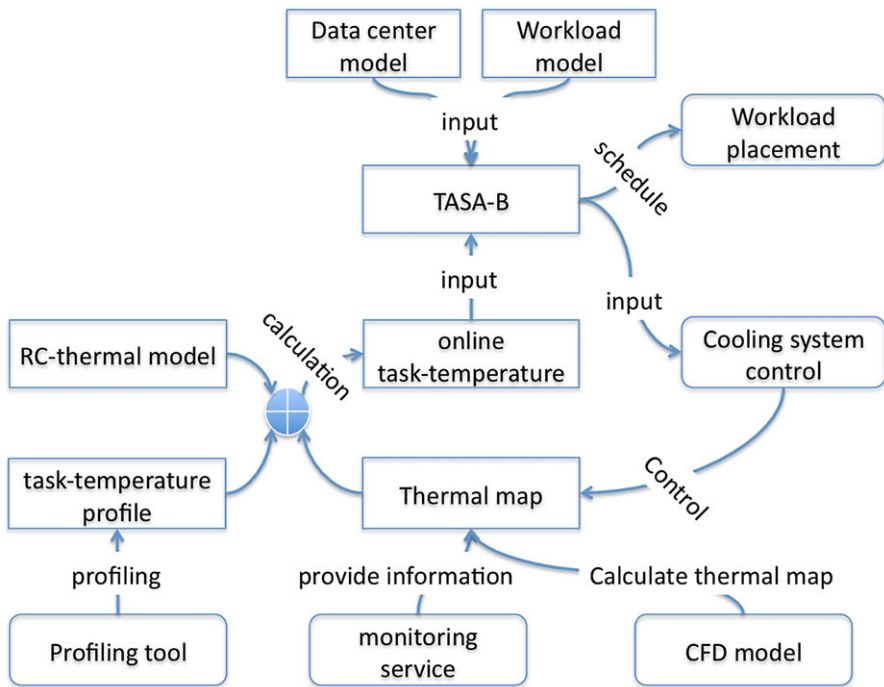


Fig. 7 Thermal aware workload scheduling framework

maximum node temperature. The workload placement can also predict future thermal maps and then give advise to future cooling system operations. Cooling system operations can change future thermal maps.

As shown in Fig. 8, with periodically updated data center information, the TASA-B executes periodically to accept incoming jobs and schedule them to compute nodes of multiple racks.

5 Thermal aware workload scheduling algorithm

This section discusses our Thermal Aware Scheduling Algorithm (TASA). The TASA is implemented in a data center scheduler and is executed periodically. Normally, a scheduling algorithm contains two stages: (1) job sorting, and (2) resource allocation. The TASA sorts incoming jobs in a decreasing order of predicted increased temperature and allocate jobs to resources with low temperatures. In other words, the TASA schedules “hot” jobs on “cold” compute nodes and tries to reduce the temperatures of compute nodes. The TASA is an online scheduling algorithm, which periodically updates resource information as well as the data center thermal map via temperature sensors.

Based on (1) temperatures of ambient environment and compute nodes which can be obtained from temperature sensors, and (2) on-line task-temperature profiles, the

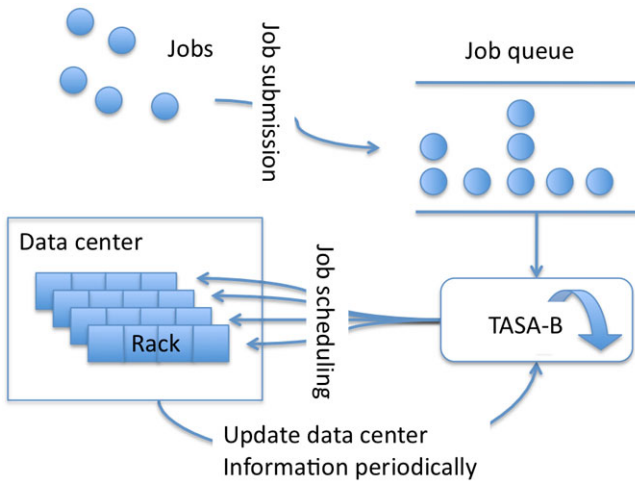


Fig. 8 The TASA-B execution context

compute node temperature after job execution can be predicated with (9). The TASA algorithm schedules jobs based on the temperature prediction.

Now, we introduce the TASA in detail. Algorithm 1 presents the Thermal Aware Scheduling Algorithm (TASA). Lines 1–4 initialize variables. Line 1 sets the initial time stamp to 0. Lines 2–4 set compute nodes available time to 0, which means all nodes are available from the beginning.

Lines 5–29, of Algorithm 1 schedule jobs periodically with an interval of T^{interval} . Lines 5 and 6 update thermal map $TherMap$ and current temperatures of all nodes from the input ambient sensors and on-board sensors. Then line 7 sorts all jobs with decreased $job_j \cdot \Delta Temp(job_j \cdot t^{\text{req}})$: jobs are sorted from “hottest” to “coolest.” Line 8 sorts all nodes with increasing node temperature at the next available time, $node_i \cdot Temp(node_i \cdot t^a)$: nodes are sorted from “coolest” to “hottest” when nodes are available.

Lines 9–14 cool down the over-heated compute nodes. If a node’s temperature is higher than a predefined temperature $TEMP^{\text{max}}$, then the node is cooled for a period of T^{cool} . During the period of T^{cool} , there is no job scheduled on this node. This node is then inserted into the sorted node list, which keeps the increased node temperature at the next available time.

Lines 16–26 allocate jobs to all compute nodes. Related research [31] indicates that based on the standard model for the microprocessor thermal behavior, for any two tasks, scheduling the “hotter” job before the “cooler” one, results in a lower final temperature. Therefore, line 16 gets a job from sorted job list, which is the “hottest” job and line 17 allocates the job with a number of required nodes, which are the “coolest.” Lines 18–20 find the earliest starting time of the job on these nodes. After that, line 24 calculates the temperature of the next available time for these nodes. Then these nodes are inserted into the sorted node list, which keeps the increased node temperature at the next available time.

Algorithm 1 Thermal Aware Scheduling Algorithm (TASA)

```

01  $t = 0$ 
02 For  $i = 1$  TO  $I$  DO
03    $node_i \cdot t^a = 0$ ;
04 ENDFOR

05 update thermal map  $TherMap$ 
06 update  $node_i \cdot Temp(t)$ ,  $node_i \in Node$ 

07 sort  $Job$  with decreased  $job_j \cdot \Delta Temp(job_j \cdot t^{req})$ 
08 sort  $Node$  with increased  $node_i \cdot Temp(node_i \cdot t^a)$ 

09 FOR  $node_i \in Node$  DO
10   IF  $(node_i \cdot Temp(node_i \cdot t^a) \geq TEMP^{max})$  TEHN
11      $node_i \cdot t^a = node_i \cdot t^a + T^{cool}$ 
12     calculate  $node_i \cdot Temp(node_i \cdot t^a)$  with (6)
13     insert  $node_i$  into  $Node$ , keep the increased order
      of  $node_i \cdot Temp(node_i \cdot t^a)$  in  $Node$ 
14   ENDIF
15 ENDFOR

16 FOR  $j = 1$  TO  $J$  DO
17   get  $job_j \cdot p$  nodes from sorted  $Node$  list,
      which are  $\{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
18    $t_0 = \max\{node_k \cdot t^a\}$ 
       $node_k \in \{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
19   FOR  $node_k \in \{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
20      $node_k \cdot t^a = t_0 + job_j \cdot t^{req}$ 
21   ENDFOR
22   schedule  $job_j$  on  $\{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
23    $job_j \cdot t^{start} = t_0$ 
24   calculate  $node_k \cdot Temp(node_k \cdot t^a)$  with (6) & (9)
       $node_k \in \{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
25   insert  $\{node_{j1}, node_{j2}, \dots, node_{jp}\}$  into  $Node$ 
      keep the increased order of  $node_i \cdot Temp(node_i \cdot t^a)$ 
       $node_i \in Node$ 
26 ENDFOR

27  $t = t + T^{interval}$ 
28 Accept incoming jobs
29 go to 05

```

Algorithm 1 waits a for period of T^{interval} and accepts incoming jobs. It then proceeds to the next scheduling round.

6 Thermal aware workload scheduling algorithm with backfilling

Algorithm 2 Thermal Aware Scheduling Algorithm with Backfilling (TASA-B)

```

01 walltime  $\leftarrow$  0
02 For  $i = 1$  TO  $I$  DO
03    $node_i \cdot t^a \leftarrow 0$ ;
04 ENDFOR
05 extranodeset  $\leftarrow \emptyset$ 
06 update thermal map TherMap
07 update  $node_i \cdot Temp(t)$ ,  $node_i \in Node$ 
08 sort Job with decreased  $job_j \cdot \Delta Temp(job_j \cdot t^{\text{req}})$ 
09 sort Node with increased  $node_i \cdot Temp(node_i \cdot t^a)$ 
10 FOR  $node_i \in Node$  DO
11   IF ( $node_i \cdot Temp(node_i \cdot t^a) \geq TEMP_{\text{max}}$ ) TEHN
12      $node_i \cdot t^a \leftarrow node_i \cdot t^a + T^{\text{cool}}$ 
13     calculate  $node_i \cdot Temp(node_i \cdot t^a)$  with (6)
14     insert  $node_i$  into Node keep the increased order
        of  $node_i \cdot Temp(node_i \cdot t^a)$  in Node
15   ENDIF
16 ENDFOR
17 FOR  $j = 1$  TO  $J$  DO
18   IF  $job_j \cdot p = 1$  THEN
19     schedule  $job_j$  with backfilling by Algorithm 4
20   ENDIF
21   IF  $job_j \cdot p > 1$  or
         $job_j$  cannot be scheduled with Algorithm 4 THEN
22     schedule  $job_j$  with Algorithm 3
23   ENDIF
24 ENDFOR
25 walltime  $\leftarrow$  walltime +  $T^{\text{interval}}$ 
26 Accept incoming jobs
27 go to 06

```

This section discusses our Thermal Aware Scheduling Algorithm with Backfilling (TASA-B). The TASA-B is implemented based on the TASA. Jobs can be backfilled if they do not violate (1) the starting time of scheduled jobs, and (2) maximum temperature of resources which have been allocated to the scheduled jobs; (1) is the requirement for normal backfilling algorithm. A scheduled job, for example job A, is allocated resources based on the resources' temperature profile. If a backfilled job,

Algorithm 3 Thermal Aware Scheduling Algorithm for job_j

```

1  get  $job_j \cdot p$  nodes,  $nodeset_j$ , from sorted Node set:
    $nodeset_j \leftarrow \{node_{j1}, node_{j2}, \dots, node_{jp}\}$ 
2   $t_0 \leftarrow \max\{node_k \cdot t^a \mid node_k \in nodeset_j\}$ 
    $node_{max1}$  is the node which gives  $\max\{node_k \cdot t^a\}$ 
3  schedule  $job_j$  on  $nodeset_j$ 
4   $job_j \cdot t^{start} \leftarrow t_0$ 
5   $Temp^{bfmax} \leftarrow \max\{node_k \cdot Temp(t^a) \mid node_k \in nodeset_j\}$ 
    $node_{max2}$  is the node which gives  $\max\{node_k \cdot Temp(t^a)\}$ 
6  FOR  $node_k \in nodeset_j - \{node_{max1}, node_{max2}\}$ 
7    $node_k \cdot bf \leftarrow yes$ 
8    $node_k \cdot t^{bfsta} \leftarrow node_k \cdot t^a$ 
9    $node_k \cdot t^{bfend} \leftarrow t_0$ 
10   $node_k \cdot Temp^{bfsta} \leftarrow node_k \cdot Temp^{node_k \cdot t^a}$ 
11   $node_k \cdot Temp^{bfend} \leftarrow Temp^{bfmax}$ 
12  insert  $node_k$  into an extra node set, extranodeset
   keep the increased order of  $node_k \cdot t^{bfsta}$ 
13 ENDFOR
14 FOR  $node_k \in nodeset_j$ 
15   $node_k \cdot t^a \leftarrow t_0 + job_j \cdot t^{req}$ 
16  calculate  $node_k \cdot Temp(node_k \cdot t^a)$  with (6) & (9)
17 ENDFOR
18 insert every node in  $nodeset_j$  into Node
   keep the increased order of  $node_i \cdot Temp(node_i \cdot t^a)$ 
    $node_i \in Node$ 

```

e.g., job B, violates the maximum temperature of scheduled resources, the resource temperature profiles for scheduling job A are violated. Therefore, maximum temperature for resources allocate to job A should be not violated by executing the backfilled job B. This is why we keep Rule (2) in the TASA-B.

Algorithm 2 presents the TASA-B. We now give a detailed explanation. TASA-B's implementation is similar with TASA except that lines 18–20 try to schedule a job on some backfilled nodes with the backfilling Algorithm 3 if the job only requires 1 processor. If the job requires more than 1 processor or the job cannot be scheduled on backfilled nodes, lines 21–23 schedule the jobs on some nodes with thermal aware scheduling Algorithm 4. We restrict the backing jobs on 1 processor to reduce the complexity of TASA-B, thus it is suitable for online scheduling. Our previous work of multiple resource coreservation algorithm [29] can be used for backfilling jobs that require more than 1 processor. This discussion goes beyond this paper's topic.

Algorithm 4 schedules job_j on some compute nodes with thermal considerations. Firstly, Algorithm 4 gets a set of nodes $nodeset_j$ for job_j , which are the coolest nodes. Line 2 locates the earliest available time t_0 of all nodes in $nodeset_j$.

Then job_j is scheduled on the node set $nodeset_j$ and the job starting time is set as the latest available time of nodes in $nodeset_j$, which is t_0 of $node_{max}$ (lines 3–4).

Algorithm 4 Thermal Aware Backfilling Algorithm for job_j

```

1 FOR  $node_k \in \text{extranodeset}$  DO
2   IF  $node_k \cdot t^{\text{bfend}} < \text{walltime}$  THEN
3      $node_k \cdot bf \leftarrow no$ 
4     remove  $node_k$  from  $\text{extranodeset}$ 
5     go to  $node_{k+1}$ 
6   ENDIF
7    $node_k \cdot t^{\text{bfsta}} \leftarrow \max\{\text{walltime}, node_k \cdot t^{\text{bfsta}}\}$ 
8   IF  $node_k \cdot t^{\text{bfend}} - node_k \cdot t^{\text{bfsta}} \geq job_j \cdot t^{\text{req}}$  and
 $node_k \cdot Temp(node_k \cdot t^{\text{bfsta}} + job_j \cdot t^{\text{req}}) \leq node_k \cdot Temp^{\text{bfend}}$ 
   THEN
9     schedule  $job_j$  on  $node_k$ 
10     $job_j \cdot t^{\text{start}} \leftarrow node_k \cdot t^{\text{bfsta}}$ 
11     $node_k \cdot t^{\text{bfsta}} \leftarrow job_j \cdot t^{\text{start}} + job_j \cdot t^{\text{req}}$ 
12     $node_k \cdot Temp^{\text{bfstar}} \leftarrow node_k \cdot Temp(job_j \cdot t^{\text{start}} + job_j \cdot t^{\text{req}})$ , which is calculated with
(6) & (9)
13  ENDIF
14  IF  $node_k \cdot t^{\text{bfsta}} \geq node_k \cdot t^{\text{bfend}}$  THEN
15     $node_k \cdot bf \leftarrow no$ 
    remove  $node_k$  from  $\text{extranodeset}$ 
16  ENDIF
17 ENDFOR

```

Line 5 calculates the maximum temperature at time t_0 for all nodes in $nodeset_j$, which is used for later backfilling algorithms. For example, if some jobs want to be backfilled on these nodes, these jobs should not delay the execution of any previously scheduled jobs, and should not affect the maximum node thermal profiles, which were used to schedule previous jobs.

Lines 6–13 update compute node information for backfilling. We use a conservative backfilling algorithm, which means that all previously scheduled jobs should be not affected. Therefore, the temperature profiles and resource available time that determine the schedule of job_j should not be affected. In specific, t_0 and $Temp^{\text{bfmax}}$ should be not changed by the backfilled jobs. Figures 9 and 10 show the available backfilling periods for $nodeset_j$. Then $node_{\text{max}1}$ and $node_{\text{max}2}$ cannot be backfilled as they have no available time or temperature slot to accommodate any backfilled jobs.

Line 7 sets nodes in $nodeset_j$ (except $node_{\text{max}1}$ & $node_{\text{max}2}$) with a flag $node_k \cdot bf = yes$, which means these nodes can be backfilled. Line 8 sets the starting time for backfilling of these nodes, which is the current available time for the nodes. Line 9 sets the end time for backfilling of these nodes, which is the earliest starting time of the job, job_j , determined by the $node_{\text{max}}$. Line 10 sets the start temperature for backfilling of these nodes, which are the temperatures of nodes at time of current available time. Line 11 sets the upper temperature limit for backfilling. If some jobs are backfilled on these nodes, the node temperature should not exceed the

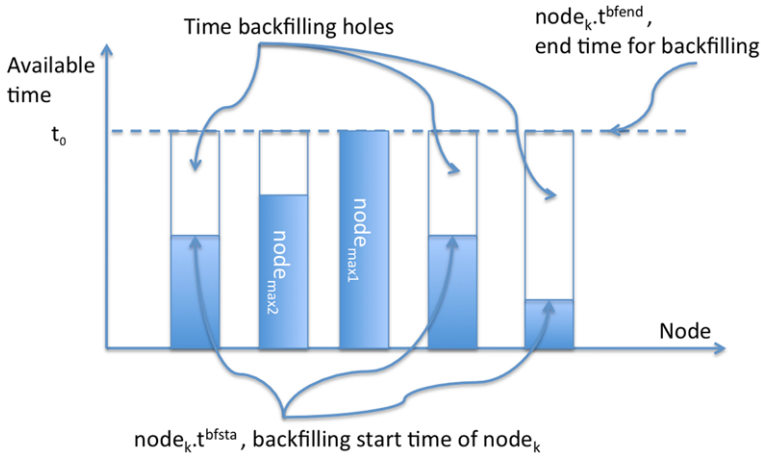


Fig. 9 Available time slots of thermal aware backfilling algorithm

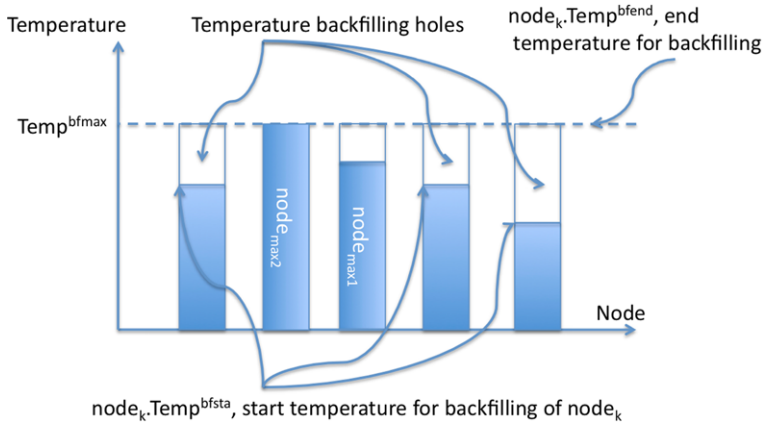


Fig. 10 Available temperature slots of thermal aware backfilling algorithm

node temperature limit $Temp^{bfmax}$, which was used to determine the which nodes are allocated for the job job_j . Finally, line 12 organizes all compute nodes for backfilling into a node set, which is sorted by the available backfilling starting time.

Lines 14–17 set up available time for nodes in $nodeset_j$ and calculate temperatures of their next available time. Line 18 finally inserts these nodes into the *Node* list with proper node temperatures of the next available time.

Algorithm 4 schedules the job_j with backfilling. To decrease the algorithm complexity, we only consider the jobs that require 1 processor. Algorithm 4 checks all nodes in $extranodeset$, which are the nodes that are sorted by increased available time for backfilling. If the end time for backfilling of a node has passed, this node cannot be backfilled and should be removed from $extranodeset$ (lines 2–6). Line 7 adjusts the start time for backfilling if it has passed. Line 8 checks whether the node's time slot and temperature slot can accommodate the job job_j . If job_j can be back-

filled on $node_k$ (line 10), then the backfilling information of $node_k$ is updated (lines 11, 12). Lines 14–15 check whether $node_k$ can be further backfilled.

7 Simulation and performance evaluation

7.1 Simulation environment

We simulate a real data center environment based on the Center for Computational Research (CCR) of State University of New York at Buffalo. All jobs submitted to CCR are logged during a 30-day period, from 20 Feb. 2009 to 22 Mar. 2009. CCR's resources and job logs are used as input for our simulation of the TASA and TASA-B.

CCR's computing facilities include a Dell x86 64 Linux Cluster consisting of 1056 Dell PowerEdge SC1425 nodes, each of which has two Irwindale processors (2MB of L2 cache, either 3.0 GHz or 3.2 GHz) and varying amounts of main memory. The peak performance of this cluster is over 13 TFlop/s.

The CCR cluster has a single job queue for incoming jobs. All jobs are scheduled with a First Come First Serve (FCFS) policy, which means incoming jobs are allocated to the first available resources. There were 22,385 jobs submitted to CCR during the period from 20 Feb. 2009 to 22 Mar. 2009. Figures 11, 12, and 13 show the distribution of job execution time, job size (required processor number), and job arrival rate in the log. We can see that 79% jobs are executed on one processor and the job execution time ranges from several minutes to several hours.

Figure 3 shows the task-temperature profiles in CCR. In the simulation, we take the all 22,385 jobs in the log as input for the workload module of TASA and TASA-B. We also measure the temperatures of all computer nodes and ambient environment with off-board sensors. Therefore, the thermal map of data centers and task-temperature profiles are available. Online temperatures of all computer nodes can also be accessed from CCR Web portal.

Fig. 11 Job execution time distribution

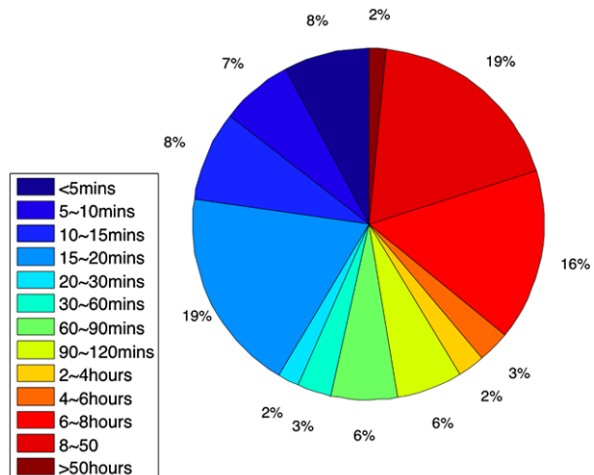
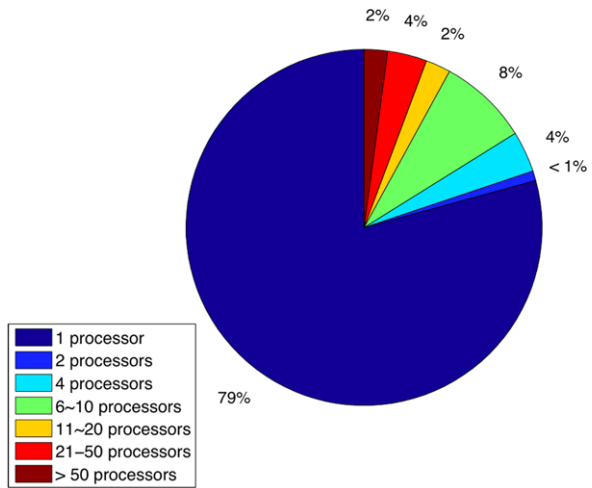
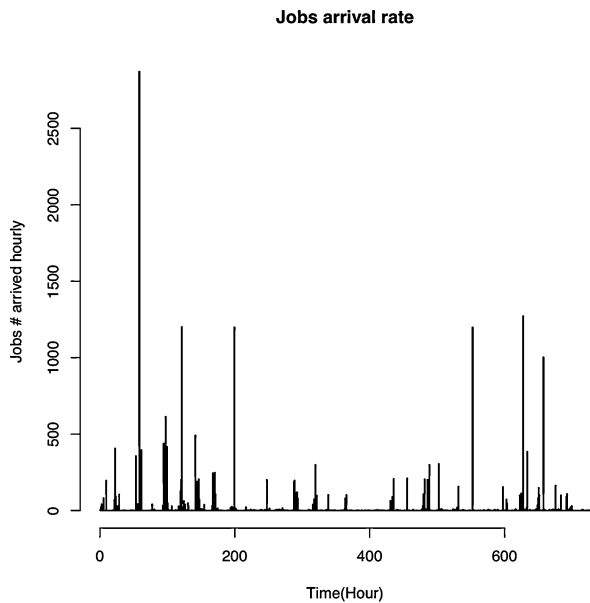


Fig. 12 Job size distribution**Fig. 13** Job arrive rate distribution

In the following section, we simulate the TASA and TASA-B based on the task-temperature profile, job information, thermal maps, and resource information obtained in CCR log files. We evaluate the thermal aware scheduling algorithms (TASA and TASA-B) by comparing them with the original job execution information logged in the CCR, which is scheduled by FCFS. In the simulation of TASA and TASA-B, we set the maximum temperature threshold of CCR to 125 F.

7.2 Experiment results and discussion

7.2.1 Data center average temperature

Firstly, we consider the average temperature in a data center. We use $\nabla Temp_{tasa}^{avg}$ and $\nabla Temp_{tasa-b}^{avg}$ to show the maximum temperature reduced by TASA and TASA-B.

$$\nabla Temp_{tasa}^{avg} = Temp_{fcfs}^{avg} - Temp_{tasa}^{avg} \quad (17)$$

$$\nabla Temp_{tasa-b}^{avg} = Temp_{fcfs}^{avg} - Temp_{tasa-b}^{avg} \quad (18)$$

where, $Temp_{fcfs}^{avg}$ is the average temperature in a data center when FCFS is employed, $Temp_{tasa}^{avg}$ is the average temperature in a data center when TASA is employed, and $Temp_{tasa-b}^{avg}$ is the average temperature in a data center when TASA-B is employed.

In the simulation, we obtained $\nabla Temp_{tasa}^{avg} = 16.1$ F, $\nabla Temp_{tasa-b}^{avg} = 14.6$ F. Therefore, TASA reduces 16.1 F and TASA-B reduces 14.6 F of the average temperatures of the 30-day period in CCR.

7.2.2 Data center maximum temperature

We also consider the maximum temperature in a data center as it correlates with the cooling system operating level. We use $\nabla Temp_{tasa}^{max}$ and $\nabla Temp_{tasa-b}^{max}$ to represent the maximum temperature reduced by TASA and TASA-B.

$$\nabla Temp_{tasa}^{max} = Temp_{fcfs}^{max} - Temp_{tasa}^{max} \quad (19)$$

$$\nabla Temp_{tasa-b}^{max} = Temp_{fcfs}^{max} - Temp_{tasa-b}^{max} \quad (20)$$

where, $Temp_{fcfs}^{max}$ is the maximum temperature in a data center when FCFS is employed, $Temp_{tasa}^{max}$ is the maximum temperature in a data center when TASA is employed, and $Temp_{tasa-b}^{max}$ is the maximum temperature in a data center when TASA-B is employed.

In the simulation, we got $\nabla Temp_{tasa}^{max} = 6.1$ F, $\nabla Temp_{tasa-b}^{max} = 4.9$ F. Therefore, TASA reduces 6.1 F and TASA-B reduces 4.9 F of the maximum temperatures of the 30-day period in CCR.

7.2.3 Job response time

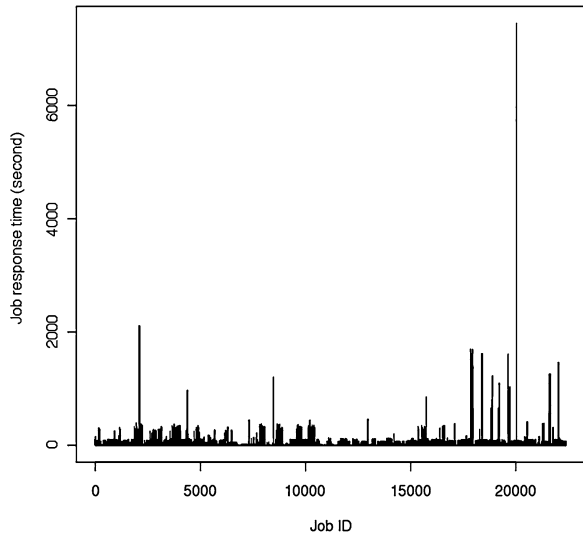
We have reduced power consumption and have increased the system reliability, both by decreasing the data center temperatures. However, we must consider that there may be trade-offs by an increased response time.

The response time of a job $job_j \cdot t^{res}$ is defined as job execution time ($job_j \cdot t^{req}$) plus job queueing time ($job_j \cdot t^{start} - job_j \cdot t^{arrive}$), as shown below:

$$job_j \cdot t^{res} = job_j \cdot t^{req} + job_j \cdot t^{start} - job_j \cdot t^{arrive} \quad (21)$$

To evaluate the algorithm from the view point of users, job response time indicates how long it takes for job results to return to the users.

Fig. 14 Job response time of FCFS



As the thermal aware scheduling algorithm intends to delay scheduling jobs to some over-hot compute nodes, it may increase the job response time. Figure 14 shows the response time of the FCFS, Figures 15 and 16 show the response time of the TASA and TASA-B, respectively.

In the simulation, we calculate the overall job response time overhead as follows:

$$\zeta_{\text{tasa}} = \frac{1}{J} \sum_{1 \leq j \leq J} \frac{\text{job}_j \cdot t_{\text{tasa}}^{\text{res}} - \text{job}_j \cdot t_{\text{fcfs}}^{\text{res}}}{\text{job}_j \cdot t_{\text{fcfs}}^{\text{res}}} \quad (22)$$

$$\zeta_{\text{tasa-b}} = \frac{1}{J} \sum_{1 \leq j \leq J} \frac{\text{job}_j \cdot t_{\text{tasa-b}}^{\text{res}} - \text{job}_j \cdot t_{\text{fcfs}}^{\text{res}}}{\text{job}_j \cdot t_{\text{fcfs}}^{\text{res}}} \quad (23)$$

ζ_{tasa} and $\zeta_{\text{tasa-b}}$ are the overhead of TASA and TASA-B, respectively, which are the average values of response time increase of TASA and TASA-B over FCFS.

In the simulation of TASA, we got the $\zeta_{\text{tasa}} = 13.9\%$, which means that we reduce the 6.1 F of maximum temperature and the average temperature by 16.1 F in CCR data center by paying a cost of increasing 13.9% job response time. In the simulation of TASA-B, we obtained the $\zeta_{\text{tasa-b}} = 11\%$, which means that we reduced the maximum temperature by 4.9 F and the average temperature by 14.6 F in CCR's data center by paying a cost of increasing 11% job response time.

7.2.4 Impact on environment: CO₂ emission

It is reported that every 1 F of maximum temperature reduced in a data center, 2% power supply of cooling system can be saved [8, 19]. Therefore, TASA can save up to 12.2% power supply of CCR's cooling system, which is up to 6.1% of CCR's total power. It is estimated that CCR's total power consumption is around 80000 kW/Hour.

Fig. 15 Job response time of TASA

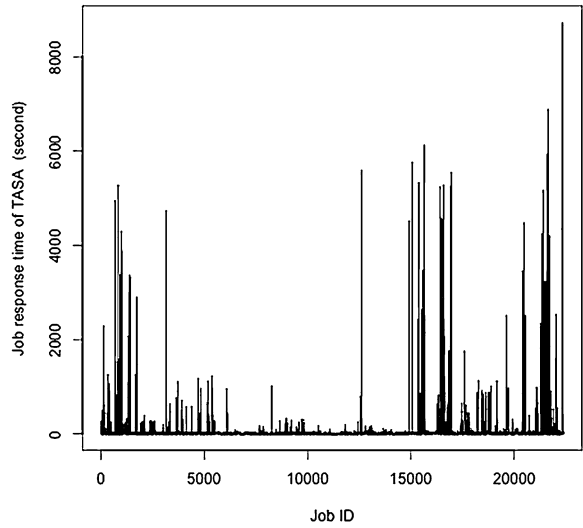
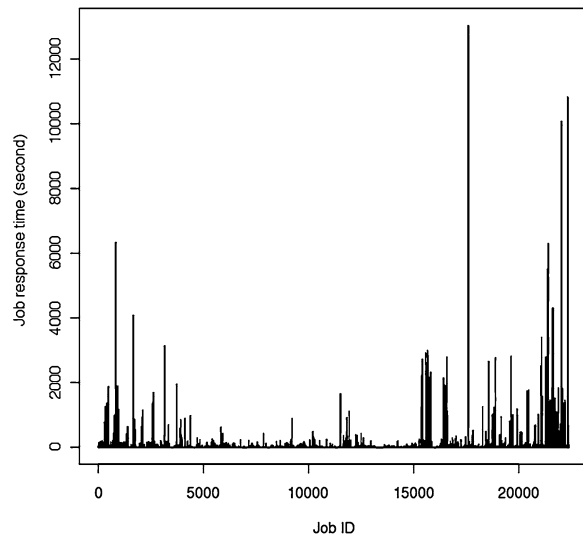


Fig. 16 Job response time of TASA-B



Thus, the TASA can save around 5000 kW/Hour power consumption. In the same way, the TASA-B can save around 4000 kW/Hour power consumption.

Table 1 shows the environmental effect of power consumption. The left column shows the power sources; the second column shows the CO₂ emission of different power sources [1]. The right column shows the power source distribution in New York state [4]. Based on Table 1, we can coarsely estimate that CCR costs 33,000 kg CO₂ emission every hour. Therefore, TASA and TASA-B can reduce 1900 kg CO₂ and 1600 kg CO₂ emission per hour, respectively. We want to give a direct image on this number. For example, Lizhe has a car, whose mark is SUBARU Legacy AWD 4

Table 1 Environment effect of power consumption

Power source	CO ₂ emission (g/kWh)	Power source distribution
Oil	881	16%
Coal	963	14%
Natural gas	569	22%
Nuclear	6	29%
Hydroelectric	4	17%
Wind Power	3–22	≤2%

Table 2 Comparison of TASA and TASA-B

Performance metrics	TASA	TASA-B
Reduced average temperature	16.1 F	14.6 F
Reduced maximum temperature	6.1 F	4.9 F
Increased job response time	13.9%	11%

Cyl. Suppose that every month, Lizhe drives this car for about 1500 km. In 6 months, this car emits nearly 1900 kg CO₂ [2].

7.2.5 Backfilling vs. no backfilling

Table 2 compares the simulation results between the TASA and the TASA-B. From Table 2, we can see that TASA can reduce more average temperature and maximum temperature than TASA-B, however, it results in a larger average job response time. This is easy to understand. The TASA-B puts some jobs into the backfilling slots while the TASA leave these backfilling slots alone, thus resulting in more running jobs and a larger reduced temperature. The various scheduling algorithm almost gives the same system utilization.

7.3 Further discussion

In our research, we assume that we can get task-temperature profiles for task scheduling. This can be done via task profiling and workload benchmarking. This requirement might require some addition work or incur some overhead for task scheduling. Firstly, if a data center's workloads always change their job execution patterns, like task execution time and data access, task-temperature profiles might be not accurate and will affect the performance of task scheduling algorithms. Furthermore, to save task-temperature profiles requires additional memory and storage access for online scheduling.

8 Conclusion

With the advent of Cloud computing, data centers are becoming more important in modern cyber infrastructures for high performance computing. However current data centers can consume a city worth of power during operation, due to factors such as resource power supply, cooling system supply, and other maintenance. The large power consumption produces huge CO₂ emissions and significantly contributes to the growing environmental issue of global warming. Green computing, a new trend for high-end computing, attempts to alleviate this problem by delivering both high performance and reduced power consumption, effectively maximizing total system efficiency.

Currently power supply for cooling systems can occupy up to 40%–50% of total power consumption in a data center. This paper presents thermal aware scheduling algorithms for data centers to reduce the temperatures in data center. Specifically, we present the design and implementation of energy-efficient scheduling algorithms, TASA and TASA-B, which allocate workloads based on their task-temperature profiles and allocate suitable resources for job execution with or without job backfilling, respectively. The algorithms are studied through a simulation based on real operation information from CCR's data center. In the simulation, TASA can reduce 16.1 F of average temperature and 6.1 F maximum temperature with an overhead of increased job response time of 13.9%. TASA-B can reduce 14.6 F of average temperature and 4.9 F maximum temperature with an overhead of increased job response time of 11%. Test results and a performance discussion justify the design and implementation of the thermal aware scheduling algorithms.

References

1. Carbon dioxide emissions from the generation of electric power in the United States. Web page. http://www.eia.doe.gov/cneaf/electricity/page/co2_report/co2report.html. Access on Feb. 2011
2. CO₂ emission calculator. Website. <http://www.falconsolution.com/co2-emission/>. Access on Feb. 2011
3. Report to congress on server and data center energy efficiency. Web page. http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf. Access on Feb. 2011
4. U.S. power grid visualization. Web page. <http://www.npr.org/news/graphics/2009/apr/electric-grid/>. Access on Feb. 2011
5. Beitelmal AH, Patel CD (2007) Thermo-fluids provisioning of a high performance high density data center. *Distrib Parallel Databases* 21(2–3):227–238
6. Choi J, Kim Y, Sivasubramaniam A, Srebric J, Wang Q, Lee J (2008) A CFD-based tool for studying temperature in rack-mounted servers. *IEEE Trans Comput* 57(8):1129–1142
7. Chrobak M, Dürr C, Hurand M, Robert J (2008) Algorithms for temperature-aware task scheduling in microprocessor systems. In: *AAIM*, pp 120–130
8. California Energy Commission. Web page. http://www.consumerenergycenter.org/tips/business_summer.html, Access on Feb. 2011
9. The Green Grid. The green grids opportunity: decreasing datacenter and other IT energy usage patterns. Technical report, the Green Grid. http://www.thegreengrid.org/~media/WhitePapers/Green_Grid_Position_WP.ashx?lang=en, Access on Feb. 2011
10. Hale PW (1986) Acceleration and time to fail. *Qual Reliab Eng Int* 2(4):259–262
11. Heath T, Centeno AP, George P, Ramos L, Jaluria Y (2006) Mercury and freon: temperature emulation and management for server systems. In: *ASPLOS*, pp 106–116

12. Hoke E, Sun J, Strunk JD, Ganger GR, Faloutsos C (2006) InteMon: continuous mining of sensor data in large-scale self-infrastructure. *Oper Syst Rev* 40(3):38–44
13. Lee EK, Kulkarni I, Pompili D, Parashar M (2010, online first) Proactive thermal management in green datacenters. *J Supercomput.* doi:[10.1007/s11227-010-0453-8](https://doi.org/10.1007/s11227-010-0453-8)
14. Lee YC, Zomaya AY (2010) Energy efficient utilization of resources in cloud computing systems. *J Supercomput.* doi:[10.1007/s11227-010-0421-3](https://doi.org/10.1007/s11227-010-0421-3)
15. Li K. Energy efficient scheduling of parallel tasks on multiprocessor computers. *J Supercomput.* (2010, online first). doi:[10.1007/s11227-010-0416-0](https://doi.org/10.1007/s11227-010-0416-0)
16. Moore J, Chase J, Ranganathan P (2006) Weatherman: automated, online, and predictive thermal mapping and management for data centers. In: *The third IEEE international conference on autonomic computing*
17. Moore JD, Chase JS, Ranganathan P, Sharma RK (2005) Making scheduling “cool”: temperature-aware workload placement in data centers. In: *USENIX annual technical conference, general track*, pp 61–75
18. Mukherjee T, Tang Q, Ziesman C, Gupta SKS, Cayton P (2007) Software architecture for dynamic thermal management in datacenters. In: *COMSWARE*
19. Patterson MK (2008) The effect of data center temperature on energy efficiency. In: *Proceedings of the 11th intersociety conference on thermal and thermomechanical phenomena in electronic systems*, pp 1167–1174
20. Ramos L, Bianchini R (2008) C-oracle: predictive thermal management for data centers. In: *HPCA*, pp 111–122
21. Rosinger PM, Al-Hashimi BM, Chakrabarty K (2005) Rapid generation of thermal-safe test schedules. In: *DATE*, pp 840–845
22. Sawyer R (2004) Calculating total power requirements for data centers. Technical report, American Power Conversion
23. Sharma RK, Bash CE, Patel CD, Friedrich RJ, Chase JS (2007) Smart power management for data centers. Technical report, HP Laboratories
24. Skadron K, Abdelzaher TF, Stan MR (2002) Control-theoretic techniques and thermal-rc modeling for accurate and localized dynamic thermal management. In: *HPCA*, pp 17–28
25. Tang Q, Gupta SKS, Varsamopoulos G (2007) Thermal-aware task scheduling for data centers through minimizing heat recirculation. In: *CLUSTER*, pp 129–138
26. Tang Q, Gupta SKS, Varsamopoulos G (2008) Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: a cyber-physical approach. *IEEE Trans Parallel Distrib Syst* 19(11):1458–1472
27. Tang Q, Mukherjee T, Gupta SKS, Cayton P (2006) Sensor-based fast thermal evaluation model for energy efficient high-performance datacenters. In: *Proceedings of the fourth international conference on intelligent sensing and information processing*, pp 203–208
28. Vanderster DC, Baniyadi A, Dimopoulos NJ (2007) Exploiting task temperature profiling in temperature-aware task scheduling for computational clusters. In: *Asia-Pacific computer systems architecture conference*, pp 175–185
29. Wang L, Cai W, Jie W, See S (2004) Models and heuristics for resource co-reservation in computational grids. *Neural Parallel Sci Comput* 12(3):261–288
30. Wang L, von Laszewski G, Dayal J, He X, Younge AJ, Furlani TR (2009) Towards thermal aware workload scheduling in a data center. In: *Proceedings of the 10th international symposium on pervasive systems, algorithms and networks (ISPAN2009)*, Kao-Hsiung, Taiwan, 14–16 Dec
31. Yang J, Xiuyi Z, Chrobak M, Youtao Z, Jin L (2008) Dynamic thermal management through task scheduling. In: *ISPASS*, pp 191–201
32. Zhang S, Chatha KS (2007) Approximation algorithm for the temperature-aware scheduling problem. In: *ICCAD*, pp 281–288