Received: 20 July 2022 | Revised: 26 September 2022 | Accepted: 2 November 2022 | Published online: 2 November 2022

RESEARCH ARTICLE

Artificial Intelligence and Applications
2023, Vol. 1(4) 244–251
DOI: 10.47852/bonviewAIA2202332

BON VIEW PUBLISHING

Novel Thermal-Aware Green Scheduling in Grid Environment

Ahmed Abba Haruna[1], Lawan Jibril Muhammad[2,*] and Mansir Abubakar[3]

[1]*College of Computer Science and Engineering, University of Hafr Al Batin, Saudi Arabia*

[2]*Computer Science Department, Federal University Kashere, Nigeria*

[3]*Department of Mathematical Sciences, Al-Qalam University, Nigeria*

Abstract: The rising energy consumption of large-scale distributed computing systems raises operational expenses and has a negative impact on the environment (e.g. carbon dioxide emissions). The most expensive operating cost aspect in data centers is the electricity consumption for cooling purposes (DC). Inefficient cooling causes excessive temperatures, which leads to hardware breakdown. To solve this issue, novel thermal-aware green scheduling algorithms were developed to dramatically reduce cooling energy consumption costs while avoiding high thermal stress conditions such as big hotspots and thermal violations while preserving typical competitive performance. As a result of this research, the novel thermal-aware green scheduling algorithms can save cooling electricity usage during job execution when compared to nongreen scheduling methods. Thus, the green scheduling algorithms clearly outperform nongreen scheduling algorithms in terms of cooling power usage effectiveness.

Keywords: thermal-aware, scheduling, data center, cooling, Round Robin, slack time

1. Introduction

As large-scale distributed computing systems grow in size, adding more and more computing nodes and storage resources, their energy consumption exponentially increases (Pop et al., 2007). Recent studies have shown that the increase in server power density has led to a concomitant increase in DC heat density. Servers require approximately 1–1.5 watts of cooling for each watt of power used (Lawton, 2007). The ratio of cooling power to server power requirements will continue to increase as DC server densities increase. Such DCs typically use two or three times the amount of power overall as used for the IT equipment, mostly for cooling (Dietrich et al., 2007).

However, transitioning to green computing has involved a number of strategies, such as integrating new approaches for power and cooling with energy-efficient hardware, virtualization, software, and power and workload management (Dietrich et al., 2007). This optimizes the efficiency of DC operations by lowering costs and lessening the impact of computing on the environment. Based on the Arrhenius time-to-fail model (Hale, 1986), at every increase of 10°C temperature, system failure rate may be doubled. Consequently, resource management with thermal considerations is important for data center operations.

*Corresponding author: Lawan Jibril Muhammad, Computer Science Department, Federal University Kashere, Nigeria. Email: lawan.jibril@fukashere.edu.ng

Vanderster et al. (2007) present the concept of a task-temperature profile, which is defined as a rise in temperature as a task is completed. By predicting resource temperatures based on online task-temperature profiles, Wang et al. (2009, 2012) created a thermal-aware task scheduling method. Using past task-temperature profile knowledge and the resistor-capacitor thermal model, they compute an online task-temperature profile. Tang et al. (2006) also propose "Thermal-Aware Job Scheduling," which uses temperature data from onboard and ambient sensors to quantify hot air recirculation and accelerate the thermal evaluation process for high-performance data centers.

When evaluating thread migration, Mulas et al. (2008) examine the thermal and performance state of the destination core. According to their findings, tasks are migrated between two cores because the destination core is cooler and works at a lower frequency than the source core. Furthermore, the chip's temperature will be more uniformly distributed if the total power required to migrate a job is less than that required to avoid migration. The migration should be triggered when a core's temperature exceeds the average chip temperature. The migration policy outperforms the halt and go technique. However, when shifting task data, the source core's temperature may grow. As a result, transferring large workloads is costly in terms of energy.

Maruyama et al. (2013) proposed minimizing hotspots to reduce cooling energy consumption in Japan's Telecommunications Equipment Room. They used various thermal control measures in their studies to eliminate hotspots. Using their technique in

tropical regions (such as Malaysia) can be problematic, as removing the hot zone is a difficult task. As a result, the Universiti Teknologi PETRONAS (UTP) server is distributed across the DC, which consists of server rooms in different parts of the university and schedules jobs to run. However, it is vital to reduce cooling energy consumption while avoiding hot areas, thermal stress conditions, and violations when jobs are being scheduled for execution.

Specific studies have been conducted to implement a green scheduling strategy that can minimize hotspots and reduce cooling power demand in DC server rooms in tropical countries (such as Malaysia) (Badi et al., 2022; Li and Qi, 2008). However, none of these studies use the proposed scheduling approach and threshold mechanism used in this study, so there is potential for additional cooling energy savings at DC. Therefore, this study is an extension of the previous version (Yousif et al., 2011; Haruna et al., 2014) that was demonstrated using the Large Hadron Collider Computing Grid (LCG) benchmark trace (I. C. London, n.d.) in a similar network environment with competing delays, average processing times, and maximum delays. However, this work requires the inclusion of scalability tests in this study.

Therefore, in this study we are considering the evaluation of the proposed novel thermal-aware green scheduling algorithms using Sharcnet benchmark traces file (Iosup et al., 2008) with different processing demands were used to observe the performance of the proposed novel thermal-aware green scheduling algorithms by varying the number of jobs.

Hence, the rest of this research is structured as follows: The proposed scheduling method is described in detail in Section 2. Section 3 provides the results of algorithm implementation and discussion, Section 4 finishes the research work, and Section 5 presents future research options.

2. Proposed Scheduling Algorithm

In this paper, we aim to design and implement a novel green scheduling algorithm that is an extension of previous versions (Least Slack Time-Based Round Robin (LSTRR) scheduling algorithm, Least Slack Time Rate First-Based Round Robin (LSTRFRR) scheduling algorithm, then First Come First Served (FCFS) scheduling algorithm, and Round Robin (RR) scheduling algorithm) (Haruna et al., 2014) that have demonstrated to perform with competitive waiting time, average turnaround time, and maximum throughput. This is accomplished by including green elements: (I) thermal awareness based on an anticipated temperature profile, (ii) air conditioning control based on a temperature threshold, and (iii) allows for an increase in the amount of air conditioning temperature settings by 1°C or 1.5°C (Maruyama et al., 2013) (e.g. 19°C–20.5°C).

A thermal-aware profile was combined with the given scheduling algorithms based on an hourly predicted temperature of eight (8) DC's server rooms on the UTP campus from 0 to 23 h (LSTRR and LSTRFRR, then FCFS and RR). Taking load balance into account, the preemption of a job for execution is based on a cyclic system specified variable (time quantum (TQ)) for the LSTRR, LSTRFRR, and RR scheduling algorithms (RR approach). Each task was given a time limit of 60 min (1 h). This means that work cannot be performed beyond the 1 h time limit. This is done to ensure that all jobs are treated equally and to reduce the computational load of each server room while the job is running.

There are tasks that are completed every hour. Comparing the temperature profile to the arrival time of the work, the algorithm routes the job to the coldest server room in DC at that time. The scheduler evaluates the room temperature level hourly as soon as the job is assigned to the associated server room for execution. When the temperature exceeds the threshold, cooling (On-AC) is activated because there is not enough space cooling to cool the space for the entire job run at that time. However, if the temperature is below the threshold, cooling (Off-AC) is not required because there is enough room cooling to keep the room cool while performing the task at that time.

These approaches used minimum temperature and threshold limits. This is done to avoid situations where the server room is too hot to allocate workloads to the servers in that room. Additional cooling flows should be used as the server slows down and can overheat. As a result of the proposed new green scheduling algorithm, the AC temperature setting can be increased by 1.5°C (AT Masato, 2013). With these limitations, the proposed algorithm can prevent high heat stress situations such as serious hotspots and heat violations and reduce the incidence of hardware failures. In other words, the AC temperature setting to reduce power consumption increases.

2.1. Algorithm

Jobs are routed to a server room in DC with a set minimum temperature level at a given hour. A process ID, an arrival time, a burst time, and a deadline are all assigned to each job. When a scheduler receives a job, it double-checks the job's arrival time. Following that, the scheduler examines the temperature profiles of the server rooms in the University's DC at that specific time period of 1 h (between 0 and 23 h). During that hour, the scheduler will select the server room with the lowest temperature. The job is subsequently distributed to the server room servers that meet the conditions of the proposed scheduling algorithm.

TQ is a predetermined value that is used to execute jobs. A job is preempted for execution depending on a system determined variable (TQ) in a cyclic form, with load balancing and the temperature hourly profile in mind (RR approach). Each project was given a time limit of 60 min (1 h). This means that a work cannot run for longer than the 1-h time limit. This means that a job cannot last more than 1 h. This is done to ensure that all jobs are treated similarly and to lessen computational demands in each server room when workloads are running.

The scheduler checks the room temperature every hour after a job has been allocated a processor; if it exceeds the threshold temperature, cooling is utilized (On-AC). However, if the temperature is below the threshold, cooling is not required (Off-AC). If a job finishes running before the time limit expires, the job ends and is removed from the system, and the next job is dispatched based on the lowest priority of the ready queue. This method is repeated until the pool is completely exhausted.

Table 1
Key symbols for T_aware LSTRFRR

Set J_i	Be the *ith* Job
n	Quantity of jobs
TQ_i	Jobs i time quantum
A_{ti}	Jobs i arrival time
d_i	Jobs i deadline
α_i	Jobs i burst time
C_i	Jobs i completion time
D_i	Jobs i absolute deadline time
T_{REi}	Jobs i remaining execution time
T_{RDi}	Jobs i remaining absolute deadline time
Pr_i	Jobs i priority rate;
T_{hi}	Jobs i temperature time from *0-23 h*
T_i	Jobs i temperature at arrival time
T_{Ti}	*20C* is defined as job i threshold Temp; the maximum temperature server should be running.
PC_{Ei}	*100W* is defined as jobs i server power when executing
AEC_i	*10 is defined as* air-condition electricity correction of job i
COP_i	*3 is defined as* coefficient of performance, because standard electric chiller's performance is normally around *3*.
T_{AECi}	Jobs i total air-condition electricity correction of PC
T_{PCTTi}	Jobs i total air-condition electricity consumption for server at Threshold Temp
T_{AECi}	Jobs i total air-condition electricity correction
T_{PCi}	Jobs i total server electricity consumption
T_{di}	Jobs i temperature difference
T_{ECi}	Jobs i total electricity consumption
PUE_i	Jobs i power usage effectiveness
S-list	Sorted list

2.2. Terms or equation

The following are some of the terms we used to describe our approach as shown in Table 1:

i. *Time quantum* TQ_i: TQ_i represents the amount of time and specifies the defined time for each job to complete periodically.

ii. *Absolute deadline*: This means the time frame in which the task must be completed.

$$D_i = \sum d_i\ ,\ A_{ti} \tag{1}$$

iii. *Remaining execution time*: This refers to the time remaining until the job is completed.

$$T_{REi} = \alpha_i - A_{ti} \tag{2}$$

iv. *Remaining absolute deadline*: This refers to the remaining execution time of the completed job.

$$T_{RDi} = \left(\sum d_i\ ,\ A_{ti}\right) - A_{ti} \tag{3}$$

v. *Priority rate*: Defines which job in the ready queue to run first.

$$Pr_i = \frac{T_{REi}}{T_{RDi} - A_{ti}} \tag{4}$$

2.3. Performance metrics

The following measures are used to assess the performance of green scheduling methods:

Electricity Consumption:

This refers to the total amount of power consumed during the execution of each job. The estimated power consumption of the air conditioner was calculated using electricity consumption (EC) constraints.

$$T_{di} = max(T_{hi}, T_{i-}T_{Ti}) \tag{5}$$

$$T_{PCi} = J_i \times P_{CEi} \tag{6}$$

$$T_{PCTTi} = \left(T_{di} = 0 \begin{cases} 0 & \left(\max\left(T_{hi},\ \frac{T_{PCi}}{COP_i}\right)\right) \\ 1 & (T_{di} = 0, T_{hi}) \end{cases}\right) \tag{7}$$

If $T_{di} = 0$ in equation (8), that is true (meaning that air conditioning was not used because the room temperature was below the threshold temperature). If *1*, it is true (meaning that AC was used because the room temperature was higher than the threshold temperature). Otherwise, if it is *1*, it is false (meaning that AC was used because the room temperature was higher than the threshold temperature). As a result, there is not enough cooling to get the job done at that time.

$$T_{AECi} = \left(\frac{T_{di}}{1.5_{^\circ C}}\right)\left(AEC_i \times \frac{T_{PCTTi}}{PC_{Ei}}\right) \tag{8}$$

$$T_{ECi} = \sum T_{PCi}, T_{AECi} \tag{9}$$

Power Usage Effectiveness

This metric measures the total amount of energy used to complete a task. Get the power usage effectiveness (PUE) by summing the total AC power of the PC at the job threshold temperature, the total AC power compensation, and the total PC power of all jobs.

$$PUE_i = \frac{\sum T_{PCTTi},\ T_{AECi},\quad T_{PCi}}{T_{PCi}} \tag{10}$$

Power Usage Effectiveness:

It refers to the total power efficiency usage while job execution. The PUE is measured by summing the total air-condition electricity for PC at threshold temperature of jobs, the total air-condition electricity correction and the total PC electricity for all jobs.

$$PUE_i = \frac{\sum T_{PCTTi},\ T_{AECi},\quad T_{PCi}}{T_{PCi}} \tag{11}$$

2.4. Procedure

```
Pool of jobs with arrival time, burst time, and deadline
List of server rooms, each with forecasted temperature profile

Begin
    For each job

     Each job should be assigned to a server room
```

```
    If the job arrival time is the same as the room
    temperature time, Select the room with the lowest
    temperature.
    TQ_i = 60 mins
Using equations (1-4), calculate the slack time rate for
all tasks
  EndFor
  Sort the job list by equation in ascending order (4);
  Store in to (S-list)
  Calculate the difference in temperature between the
server room temperature and the temperature limit (5)
  If (T_di != 0)
  Calculate the total amount of electricity used by the
server (6)
  As a result, there is not enough cooling to finish the
work. Calculate the total air-conditioning
electricity usage for the server at the specified
temperature using AC (7)
  Calculate the total electricity correction for air
conditioning (8)
Else
   There is sufficient cooling to complete the task.
Turning off the air conditioning.
   Calculate the total amount of electricity used by the
server (6)
EndIf
   While (S-list is not empty)
   Execute jobs at the CPU level in response to demand.
      If (α_i > 0)
        α_i = α_i - TQ_i
        Put back into S-list
      EndIf
   EndWhile
   Compute total electricity consumption (9)
   Compute the power usage effectiveness (10)
End
```

3. Results and Discussions

This phase involved testing the proposed T_aware LSTRFRR scheduling algorithm with the proposed scheduling algorithm was compared to various green and nongreen scheduling algorithms from the work originally presented at the 2014 IEEE 3rd International Conference on Computer and Information Sciences (Yousif et al., 2011). However, a Sharcnet traces file (Iosup et al., 2008) with various processing demands was employed in this study to observe the performance of the suggested green scheduling methods by altering the number of jobs. By increasing the real demand, a scalability test for the scheduling methods was required (Kołodziej et al., 2013). As a result, two data sets were created, one utilizing 50 traces files processes and the other using 100.

The nongreen scheduling algorithms (FCFS, RR, LSTRR, and LSTRFRR) were compared to the green scheduling algorithms (T_aware FCFS, T_aware RR, T_aware LSTRR, and T_aware LSTRFRR) (Haruna et al., 2014).

3.1. Using 50 Sharcnet traces

Figure 1 shows how Sharcnet traces were used to plan 50 jobs for execution. Because all of the jobs came at 0, 3, 4, and 11 a.m., they were all completed in server room 2. In comparison to the

Figure 1
Electricity consumption (W) based on 50 Sharcnet traces in server room 2

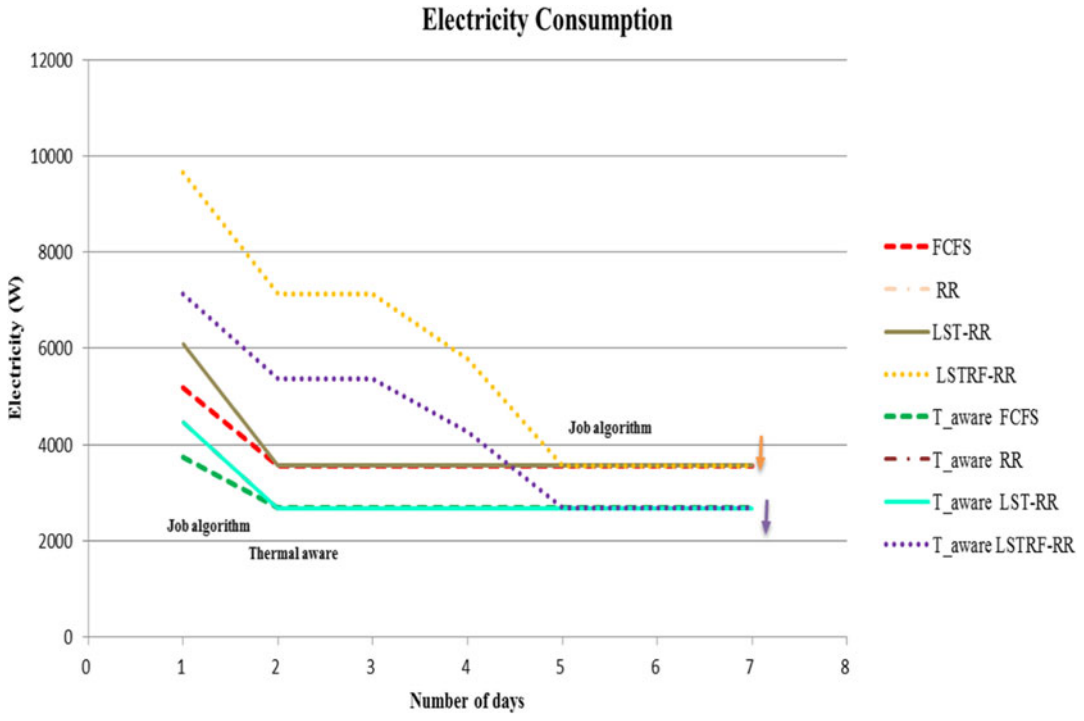


Figure 2
Power usage effectiveness (PUE) based on 50 Sharcnet traces in server room 2

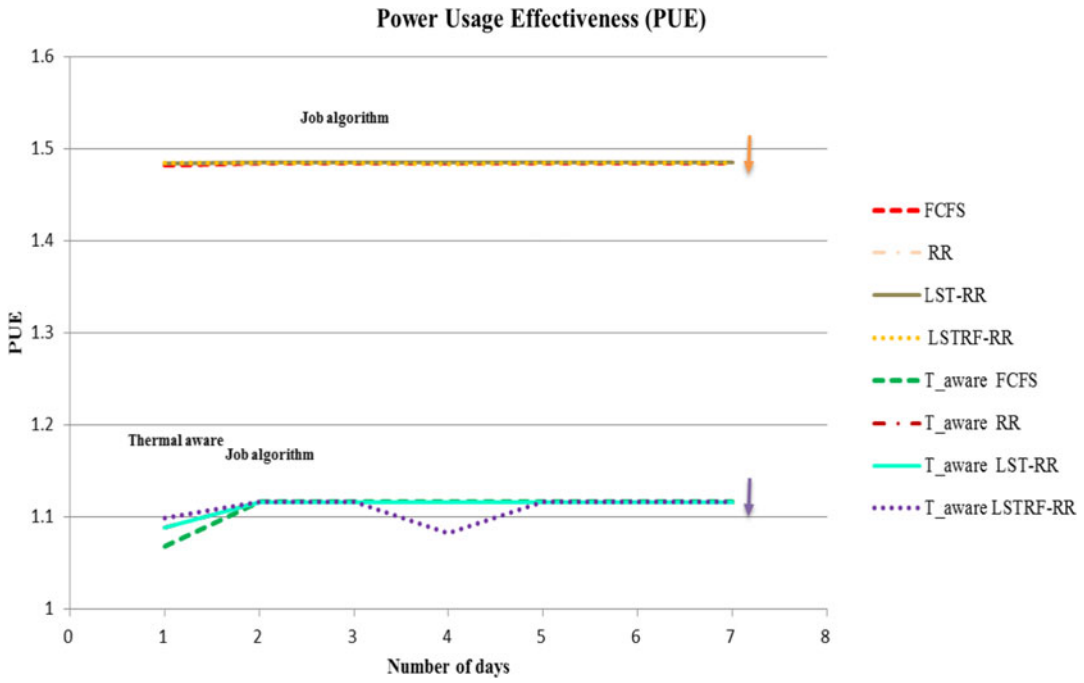
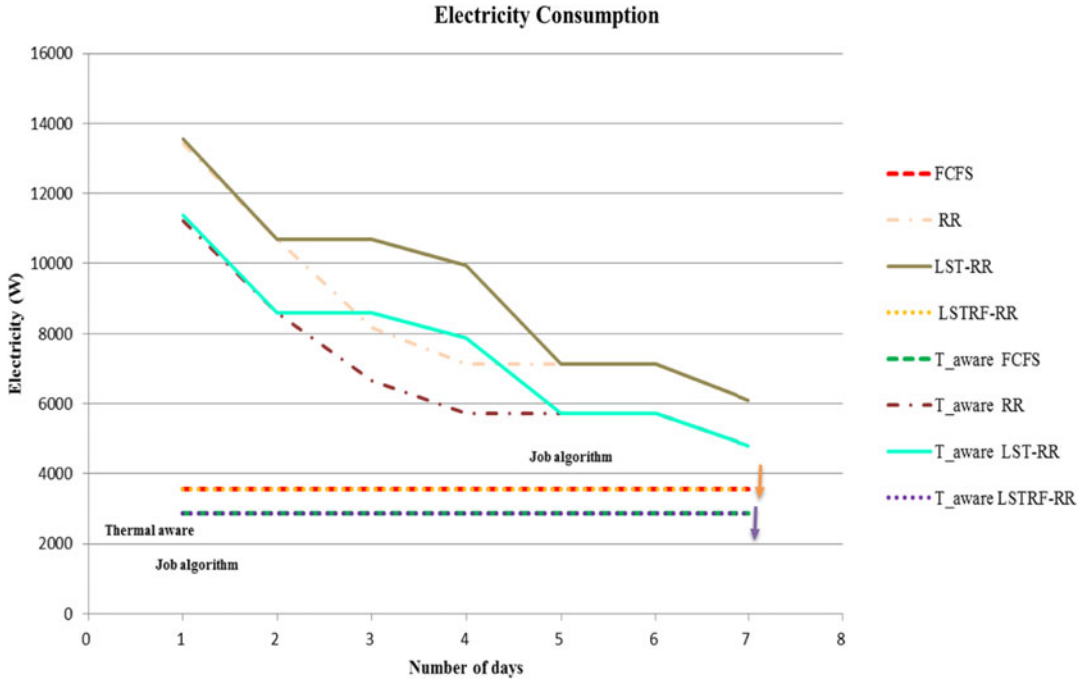


Figure 3
Electricity consumption (W) based on 100 Sharcnet traces in server room 6



other server rooms, server room 2 has the lowest temperature at 0, 3, 4, and 11 h (see Figure 2).

Figure 1 shows the cooling power consumption of 50 jobs that ran for a week (1st to 7th day) using the green scheduling algorithm and the nongreen green scheduling algorithm. Compared to nongreen scheduling algorithms, green scheduling algorithms save cooling power consumption.

Figure 1 shows that the amount of cooling electricity consumed decreases from day 1 to day 2. The reason for this is that the server can quickly complete important subtasks. However, from day 2 to day 7, the cooling usage remained unchanged. This is owing to the fact that the task/subtask will take longer than 7 days to complete. The simulation lasts 7 days.

T aware LSTRFRR saves 25.31% (10217.43/40377.03 ×100) of cooling electricity usage, according to an exhaustive examination of the 50 jobs executed during the week (1–7 days). T aware LSTRR reduces cooling EC by 25.22% (6926.08/27465.22 × 100), T aware FCFS reduces cooling EC by 25.42% (6752.93/26570.28 × 100), and T aware RR scheduling algorithm reduces cooling EC by 25.22% (6926.08/27465.22 × 100).

From day 1 to day 7, the performance of the nongreen scheduling method is the same as 1.48 PUE as shown in Figure 2. The green scheduling algorithm, on the other hand, has the same performance from day 1 with PUE 1.09 and grows to day 7 with PUE 1.11. The 1st day PUE of the T aware FCFS is 1.06, and the 4th day PUE of the T aware LSTRFRR is 1.08. It is clear that the green scheduling algorithm has a better PUE for cooling than the nongreen scheduling algorithm.

3.2. Using 100 Sharcnet traces

Figures 3 and 4 show the results of an examination of 100 jobs run in server room 6. Figures 5 and 6 show the results of 100 jobs run in server room 2. Server room 2 jobs arrived at 0, 3, 4, and 11 a.m., while server room 6 jobs arrived at 7, 8, 9, 10, 14, 16, 17, and 18 a.m. Figure 2 shows that server room 2 has the lowest temperature at 0, 3, 4, and 11 h, whereas server room 6 has the lowest temperature at 7, 8, 9, 10, 14, 16, 17, and 18 h.

Figure 3 shows that the green scheduling algorithm can save cooling power compared to the nongreen scheduling algorithm.

The cooling electricity required by the green scheduling algorithms and the nongreen scheduling algorithms is decreasing from day 1 to day 7 as shown in Figure 3. The reason for this is that the server swiftly completes important subtasks. From day 1 to day 7, the cooling consumption of green scheduling algorithms and nongreen scheduling strategies remained consistent. This is owing to the fact that the task/subtask will take longer than 7 days to complete. The simulation period is 7 days.

T aware LSTRFRR saves 19.74% (4925.17/24946.64 × 100) of cooling electricity usage, according to the experimental investigation. T aware LSTRR saves 19.30% (12585.71/65227.21 × 100), T aware FCFS saves 19.74% (4925.17/24946.64 × 100), and T aware RR scheduling method saves 18.91% (11297.76/59741.50 × 100) of cooling electricity use over the course of a week (from day 1 to day 7).

Throughout the week, Figure 4 shows that the green scheduling algorithms outperform the nongreen scheduling algorithms in terms of PUE.

When compared to nongreen scheduling algorithms, Figure 5 shows that green scheduling strategies can save cooling EC. Figure 5 shows that the amount of cooling electricity consumed (by green and nongreen scheduling techniques) is fairly high on day 1, but decreases on day 2. The reason for this is that the server swiftly completes important subtasks. However, the cooling consumption remained constant from day 2 to day 7 because the task/subtask required more than 7 days to finish, and the simulation duration was 7 days.

The result analysis shows that T aware LSTRFRR saves 29.37% (4896.65/16674.98 × 100) of cooling electricity usage over the course of a week (from day 1 to day 7). T aware LSTRR reduces cooling EC by 24.29% (8739.04/35975.88 × 100), T aware FCFS reduces cooling EC by 25.51% (8596.60/33696.85 × 100), and T aware RR scheduling algorithm reduces cooling EC by 24.29% (8739.04/35975.88 × 100) per week.

Figure 6 shows that the nongreen scheduling scheme maintains a constant PUE of 1.48 from day 1 to day 7. Similarly, Green's scheduling algorithm works consistently from day 1, with a PUE of 1.14, a drop of 1.11 by day 2, and a PUE of 1.07 by day 7. It is clear that the green scheduling algorithm is more successful in utilizing the cooling capacity than the nongreen scheduling algorithm.

Figure 4
Power usage effectiveness (PUE) based on 100 Sharcnet traces in server room 6

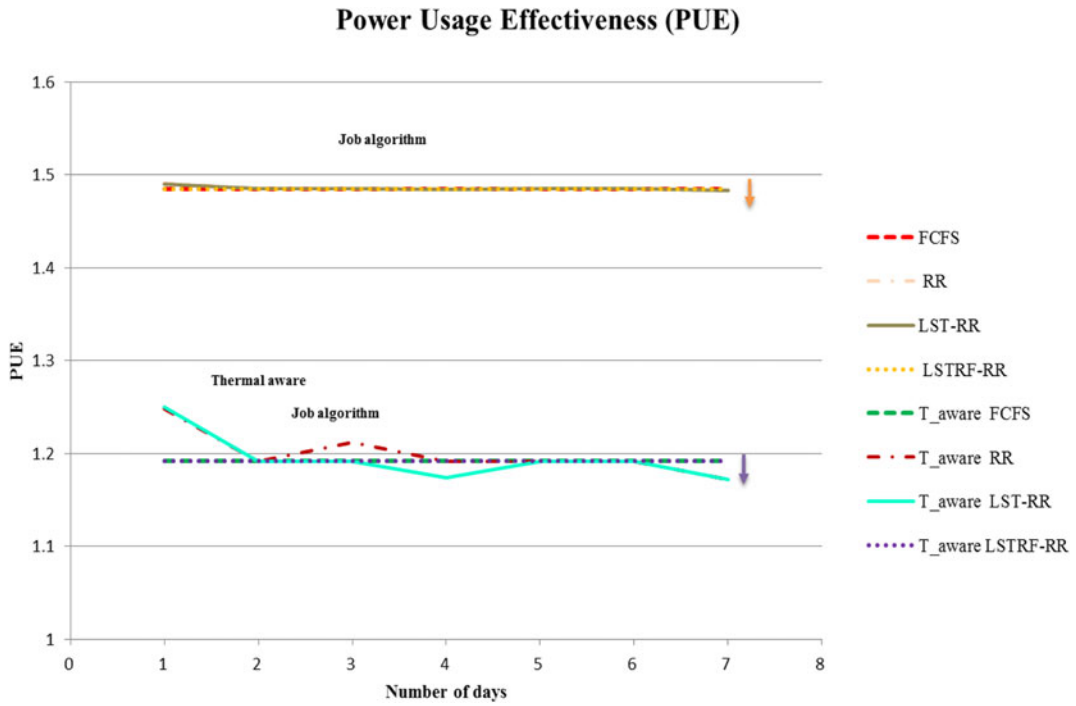


Figure 5
Electricity consumption (W) based on 100 Sharcnet traces in server room 2

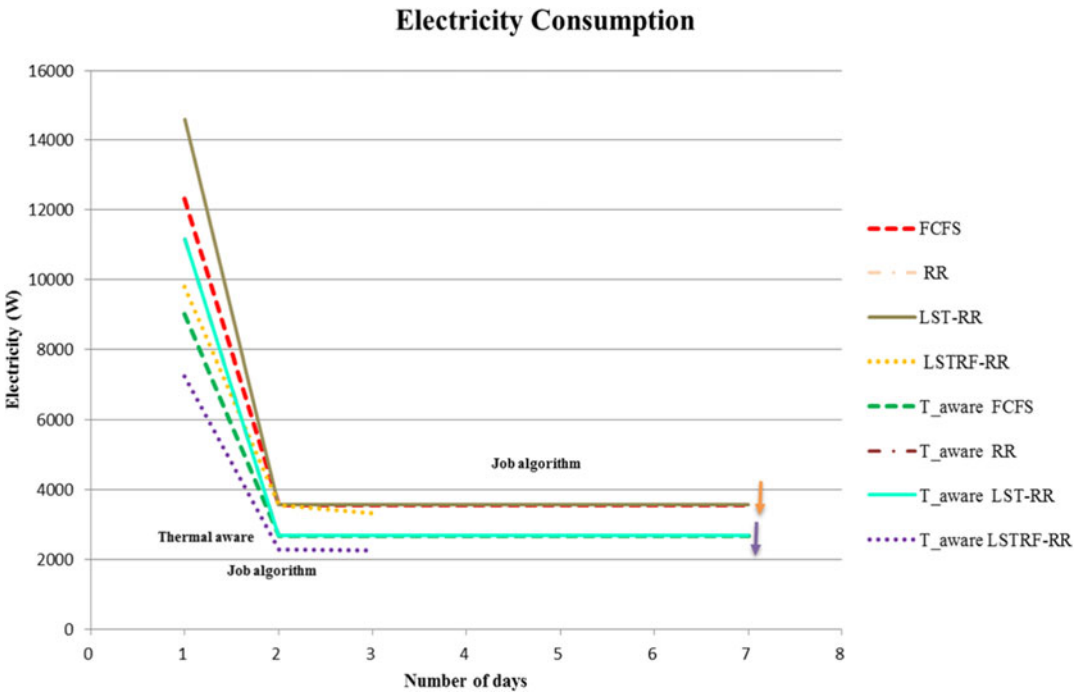
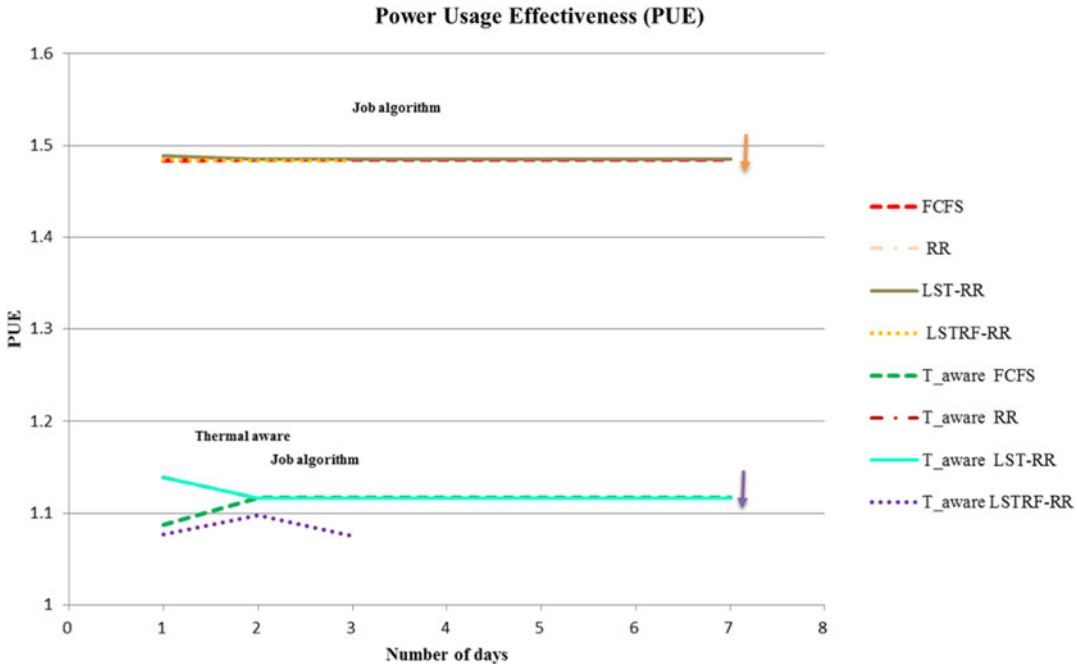


Figure 6
Power usage effectiveness (PUE) based on 100 Sharcnet traces in server room 2



4. Conclusions

In this study, a detailed performance evaluation is performed using the Sharcnet benchmark trace to test the effectiveness of the proposed green scheduling method. The proposed green scheduling method significantly reduces cooling power consumption, such as large hotspots, while maintaining competitive performance when running workloads or jobs while preserving space on the DC server. The proposed green scheduling methods significantly reduces high thermal stress conditions and avoids thermal overshoot (by applying minimum temperature and threshold). Therefore, the research finding

concludes that green scheduling algorithms can reduce cooling power consumption during job execution compared to nongreen scheduling algorithms (50 and 100 jobs, respectively). Moreover, it is also evident that the green scheduling algorithms are better with cooling power usage effectiveness (PUE) compared to the nongreen scheduling algorithms.

5. Future Research Work

However, in a situation when resource users' jobs miss the deadline, the present scheduling systems based on controlled resource scheduling provide little or no compensation. The lack of knowledge about performance aspects and paying resource users may discourage them from submitting work to the grid. As a result, future research should take into account resource users' performance goals and incentivize them to stay on the grid by including an economy model into a regulated grid (Aram et al., 2005; Haruna et al., 2021; Wang et al., 2009). The economic model's requirements are intended to satisfy both resource providers and their users in a controlled resource scheduling framework. The cost of all grid resources is determined by the resource's performance.

Conflicts of Interest

Lawan Jibril Muhammad is an editorial board member for *Artificial Intelligence and Applications*, and was not involved in the editorial review or the decision to publish this article. The authors declare that they have no conflicts of interest to this work.

References

Aram, G., Czajkowski, K., & Lerman, K. (2005). Resource allocation in the grid with learning agents. *Journal of Grid Computing*, *3*, 91–100. https://doi.org/10.1007/s10723-005-9003-7

Badi, I., Muhammad, L. J., Abubakar, M., & Bakır, M. (2022). Measuring sustainability performance indicators using FUCOM-MARCOS methods. *Operational Research in Engineering Sciences: Theory and Applications*, *5*(2), 99–116. https://doi.org/10.31181/oresta040722060b.

Dietrich, J., Schmidt, R., Hogan, M., & Allen, G. (2007). *The green data center*.

Hale, P. W. (1986). Acceleration and time to fail. *Quality and Reliability Engineering International*, *2*(4), 259–262. https://doi.org/10.1002/qre.4680020409

Haruna, A. A., Jung, L. T., Arputharaj, V., & Muhammad, L. J. (2021). Incentive-scheduling algorithms to provide green computational data center. *SN Computer Science*, *2*, 252. https://doi.org/10.1007/s42979-021-00633-5

Haruna, A. A., Zakaria, N. B., Jung, L. T., Pal, A., Naono, K., & Okitsu, J. (2014). Performance comparison of least slack time based heuristics for job scheduling on computational grid. In *2014 International Conference on Computer and Information Sciences*, 1–6. https://doi.org/10.1109/ICCOINS.2014.6868841

Iosup, A., Li, H., Jan, M., Anoep, S., Dumitrescu, C., Wolters, L., & Epema, D. H. (2008). The grid workloads archive. *Future Generation Computer Systems*, *24*(7), 672–686. https://doi.org/10.1016/j.future.2008.02.003

Kołodziej, J., Khan, S. U., Wang, L., Byrski, A., Min-Allah, N., & Madani, S. A. (2013). Hierarchical genetic-based grid scheduling with energy optimization. *Cluster Computing, 16*, 591–609. https://doi.org/10.1007/s10586-012-0226-7

Lawton, G. (2007). Powering down the computing infrastructure. *Computer*, *40*(2), 16–19. https://doi.org/10.1109/MC.2007.69

Li, F. F. and Qi, D. Y. (2008). Research on grid resource allocation algorithm based on fuzzy clustering. In *2008 Second International Conference on Future Generation Communication and Networking*, 162–166. http://doi.org/10.1109/FGCN.2008.88

I. C. London. (n.d.) *HEP e-Science group*. Retrieved from: http://lcg.web.cern.ch/LCG/.

Maruyama, M., Takahashi, A., Yajima, H., Takeuchi, A., Yamashita, N., Matsumoto, M., & Tajima, K. (2013). Reducing electric power consumption for air conditioning by improving temperature distribution in telecom equipment rooms. *NTT Technical Review*, *11*(10), 1–6.

Mulas, F., Pittau, M., Buttu, M., Carta, S., Acquaviva, A., Benini, L., & Atienza, D. (2008). Thermal balancing policy for streaming computing on multiprocessor architectures. In *Proceedings of the Conference on Design, Automation and Test in Europe*, 734–739.

Pop, F., Tudor, D., Cristea, V., & Cretu, V. (2007). Fault-tolerant scheduling framework for MedioGRID system. In *EUROCON 2007-The International Conference on "Computer as a Tool"*, 505–510.

Tang, Q., Mukherjee, T., Gupta, S. K., & Cayton, P. (2006). Sensor-based fast thermal evaluation model for energy efficient high-performance datacenters. In *2006 IEEE Fourth International Conference on Intelligent Sensing and Information Processing*, 203–208.

Vanderster, D. C., Baniasadi, A., & Dimopoulos, N. J. (2007). Exploiting task temperature profiling in temperature-aware task scheduling for computational clusters. In *Asia-Pacific Conference on Advances in Computer Systems Architecture*, 175–185.

Wang, L., Khan, S. U., & Dayal, J. (2012). Thermal aware workload placement with task-temperature profiles in a data center. *The Journal of Supercomputing*, *61*, 780–803. https://doi.org/10.1007/s11227-011-0635-z

Wang, L., & Singh, C. (2009). Multicriteria design of hybrid power generation systems based on a modified particle swarm optimization algorithm. In *IEEE Transactions on Energy Conversion*, *24*(1), 163–172. https://doi.org/10.1109/TEC.2008.2005280

Wang, L., Von Laszewski, G., Dayal, J., He, X., Younge, A. J., & Furlani, T. R. (2009). Towards thermal aware workload scheduling in a data center. In *2009 IEEE 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, 116–122. http://doi.org/10.1109/I-SPAN.2009.22

Yousif, A., Abdullah, A. H., & Ahmed, A. A. (2011). A bidding-based grid resource selection algorithm using single reservation mechanism. *International Journal of Computer and Applications*, *16*(4), 39–43. https://doi.org/10.5120/1998-2694

How to Cite: Haruna, A. A., Muhammad, L. J., & Abubakar, M. (2023). Novel Thermal-Aware Green Scheduling in Grid Environment. *Artificial Intelligence and Applications*, *1*(4), 244–251, https://doi.org/10.47852/bonviewAIA2202332