INL/CON-24-82519-Revision-1

Job Scheduler-Driven Power Gateway for High Performance Computing

June 2025

Matthew R Sgambati, Matthew William Anderson

Changing the World's Energy Future

Idaho National Laboratory

INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, LLC

INL/CON-24-82519-Revision-1

Job Scheduler-Driven Power Gateway for High Performance Computing

Matthew R Sgambati, Matthew William Anderson

June 2025

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Job Scheduler-Driven Power Gateway for High Performance Computing

Matthew Sgambati and Matthew Anderson
Idaho National Laboratory
Idaho Falls, Idaho, USA
matthew.sgambati@inl.gov, matthew.anderson2@inl.gov

***Abstract*—Power gateways in the form of a microgrid can incorporate multiple distributed energy resources (DER) in either grid forming or grid following mode and support high performance computing (HPC) power profiles including the large load-follow requirements observed in multi-user HPC systems. The microgrid's flexibility to operate in either grid forming or grid following mode and to actively switch between these modes enables baseline power from multiple non-baseline DER while maintaining high power quality metrics for the HPC system. But this enormous flexibility in demand response and time of use shifting is generally programmed independently of any integration with an HPC job scheduler, which can better inform the load shaping by the microgrid. While there are many existing approaches where the HPC job scheduler takes in information from the grid to make queue scheduling decisions, this work takes the opposite view and explores three strategies where the forecast of jobs in the queue can change the settings of the grid. Three strategies are tested where the forecast of jobs in the queue adjusts the settings of a microgrid designed for a datacenter, in this case with three classes of HPC architectures. The strategies are demonstrated using a microgrid with 64 kW of solar capacity and 320 kWh of battery over a period of 21 days operating with significant load-follow swings, a throttled grid, cloudy conditions, switching between grid following and grid forming modes, and a wide range of battery states-of-charge, all while maintaining high quality power metrics. The strategies presented provide a mechanism for the forecast of jobs in the queue to influence and adjust the settings of a microgrid and to improve HPC power outcomes such as maximizing renewable energy usage.**

***Index Terms*—Microgrids, HPC Schedulers, Power**

I. Introduction

The integration of distributed energy resources (DER) that consist principally of renewable energy sources with inconsistent baseline power supply and widely varying load-follow limitations presents a challenge for high performance computing (HPC) operations. These challenges are exacerbated by the significant load-follow needs present in HPC systems where power swings can exceed 30% of load per minute even in steady-state operation. As an example, the power for a 100,000 core system serving 1600 users is shown in Figure 1 for the period of one year, and a scatter plot of load-follow jumps in percent of load per minute is shown in Figure 2. The load-follow jumps vary significantly during normal operation in large part due to the wide variety of job sizes submitted: jobs submitted that require tens of thousands of cores mean that the system must reserve cores and wait to clear out smaller jobs in order to meet the large core-count resource request while forcing other jobs in the queue that cannot be backfilled to wait until the larger resource request job has run. This behavior causes significant periodic jumps in load-follow even while the HPC system is in steady-state operation.

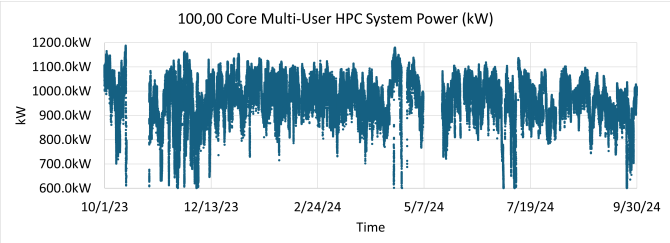

Fig. 1. The power usage of a 100,000 core x86 multi-user HPC system is shown here in 1 minute increments for a one year period. The system supports 1600 users running a wide variety of applications from Monte Carlo to thermal hydraulics and shows load-follow requirements typical of multi-user HPC systems.

DERs such as wind, solar, geothermal, hydrogen, biomass, and nuclear microreactors, in contrast, may not support such large load-follow requirements except via overprovisioning of power or via load shaping techniques such as peak shaving [1] in order to match the DER capabilities to the load demand. One way to add flexibility to integrating DERs with HPC operations is using a microgrid as a power gateway. The microgrid can integrate multiple DERs simultaneously, thereby reducing the load-follow constraints of an individual DER, while also addressing power quality issues such as a high total harmonic distortion (THD), high power harmonics, voltage spikes, and voltage dips or swells to which HPC systems are sensitive [11].

Microgrids typically consists of batteries, converters, inverters, programming mechanisms, monitoring, HVAC, and DER integration mechanisms and have played prominent roles in providing emergency power after natural disasters including the Tohoku earthquake [15] and hurricane Ian [18]. The success of microgrids operating in emergency conditions inspired the creation of the "microgrid in a box" (MIB) class of microgrid [14, 10, 7] built out of commercial off the shelf (COTS) components. The MIB class of microgrid was also designed with unique power conditioning and fast response capabilities that makes it well suited for HPC operation. As an example, in Figure 3 the MIB class microgrid implementation used in this work is shown conditioning 480 volt input power via program

settings for four days and then turning off the conditioning for four days to demonstrate the capability to reduce voltage variance. While HPC server power supplies are frequently provisioned with some power conditioning capability for short voltage sags or swells, this comes at additional expense per server and does not cover significant variances nor all types of power quality issues. In contrast, an MIB class microgrid conditions the power against a wide range of power quality issues, including transients for the entire system at once, while providing flexibility in load shaping with input from multiple DERs.

In addition to conditioning power quality, integrating multiple DERs, and buffering load-follow constraints for any single DER input, the MIB class microgrid enables a crucial capability not typically available to existing HPC datacenters: switching from grid following mode to grid forming mode and vice versa while maintaining high quality power for HPC usage. In grid following mode, the microgrid is connected to the grid but the power taken from the utility grid can be throttled to a certain threshold that is programmable. In this mode, HPC operators can control when and how much power they choose to take from the grid independent of the current HPC load and thereby implement time of use shifting. In grid forming mode, the microgrid can operate entirely independent of the utility grid where the microgrid acts as an uninterruptible power supply (UPS). Alternatively, the microgrid can operate in a type of hybrid mode where it is grid-tied but where the microgrid takes no power from the utility grid but actively sells power back to the utility. This level of flexibility in programming the grid settings is not typical for HPC systems and presents multiple opportunities where the scheduler, using the job queue or other parameters, could alter and optimize the programming choices and behavior of the microgrid settings for improved power outcomes beyond just improving HPC system resiliency.

This work explores an MIB class microgrid implementation made from COTS components for use with HPC systems to explore three use cases:

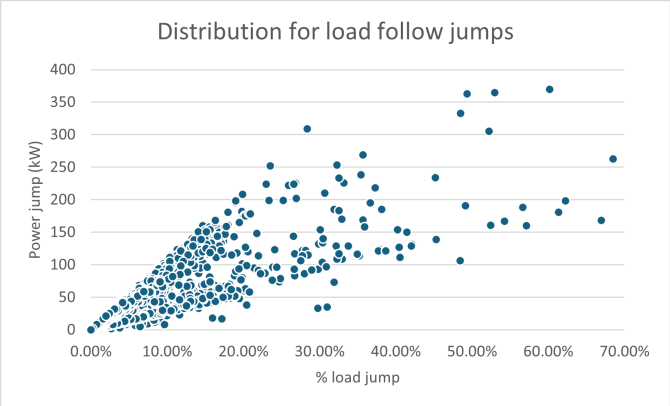

Fig. 2. The power load-follow jumps in percent of load per minute of the 100,000 core x86 multi-user HPC system in Figure 1 is shown here in 1 minute increments over a one year period.

- Significant load-follow time variation on an HPC system resulting from a scheduler driven workload in a heterogeneous system consisting of ARM, GPU, and x86 nodes;
- Grid following power operation where the MIB class microgrid throttles the utility grid usage to programmable levels determined by the job forecast on an HPC system;
- Grid-tied operation where there is no utility grid usage and the microgrid sends power back to the utility grid all while supporting the entire HPC system.

Three strategies are tested where a job queue forecast is used to adjust the microgrid settings. The key metrics explored in these experiments are load shaping and scheduler operation on time scales of three weeks during which the total harmonic distortion, HPC load, battery state-of-charge, solar generation, and utility grid usage are measured. This work is structured as follows: related work on microgrids and schedulers is covered in section II; the MIB class implementation is detailed in section III along with key programming settings; the HPC architectures, workloads, and mobile datacenter used in the experiments are described in section IV; the scheduler strategies tested in this work are described in section V; results from operating the HPC datacenter for three weeks using the microgrid with the scheduler implementation are presented in section VI; a discussion of the results and implications along with conclusions are given in section VII.

II. Related Work

Power-aware scheduling in the absence of a microgrid has been explored extensively in HPC literature. In one of the early studies of power-aware scheduling, Verma et al. [24] quantified a broad range of HPC application power profiles and explored the dynamic placement of HPC applications to implement a power-savings scheduler. Much later, Yang et al. [28] introduced a power-aware scheduling mechanism capable of adjusting for the variability in electricity costs and

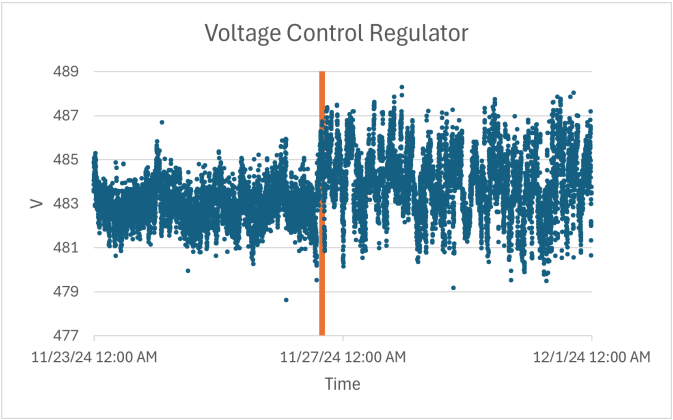

Fig. 3. The voltage regulation provided by the MIB class microgrid used for powering the HPC systems in this study is shown here. In this figure, the voltage regulation control was initially turned on for four days and then turned off at the orange line for four days for comparison. The voltage regulation control significantly improves the power quality provided by the microgrid to the HPC systems.

tested this strategy using HPC job traces. Similarly, Wallace et al. [25] introduced a scheduling approach to limit the power consumption of the entire HPC system to a user defined value by leveraging a job power profile estimator. Peng et al. [19] introduced a renewable-energy-aware job scheduler where jobs are placed in a scheduling window if resources permit and the window size is changed based on availability of renewable energy. With the exception of Verma et al., these former approaches all used HPC job traces to evaluate the effectiveness of the proposed strategies. In contrast, Chadha et al. [3] extended the SLURM scheduler to support power-aware reconfiguration options for so-called malleable jobs where the job resource size can be adjusted to improve power outcomes after execution has begun and tested this on a production HPC system rather than using job traces. Concurrently, Dupont et al. [6] explored power-aware scheduling strategies for such malleable jobs from a theoretical and simulation perspective. Efforts towards power-aware HPC schedulers have remained disjoint from microgrid deployments with the exception of Ding et al. [5] where a scheduler was integrated with a microgrid in a theoretical study to reduce HPC datacenter carbon emissions.

Microgrids are often deployed during natural disasters [15, 18] while more recently being mentioned more in the context of HPC for integration with nuclear microreactors to manage reactor load-follow constraints [2, 16] or for integrating small modular reactors for a university campus [27, 9]. Microgrid technology is also frequently explored as a mechanism for integrating nuclear technologies with renewable energy sources [17]. Sharma et al. [20] provided a comparative analysis of microgrid control strategies and cover the cognitive decision agent architecture [23] for island and grid connected modalities that is relevant to HPC scheduler integration. A general review of microgrid technology and trends was conducted by Uddin et al. [22].

The work presented here complements the previously mentioned studies by providing a reduction to practice of an HPC datacenter integrated with a programmable microgrid built from COTS components capable of grid following and grid forming operation without creating power quality issues that would cause damage to the HPC systems.

III. The MIB class Microgrid

A microgrid is an integrated power system with a rich history in providing an emergency power grid during natural disasters. However, in 2018 Ding et al. [5] discussed the potential of a microgrid as a mechanism to reduce carbon emissions from an HPC datacenter in day-to-day operations rather than simply as an emergency power operation mechanism. The approach followed by Ding et al. was simulated and showed a theoretical route whereby an HPC datacenter driven by a microgrid could provide baseline power by leveraging multiple non-baseline DERs and thereby reduce emissions from the HPC datacenter and maximize renewable energy usage. However, the approach could not explore the power quality impacts of the strategy, and the batch job core count resource request was not considered, as would be typically done for HPC operations. To complement the work of Ding et al., the MIB class microgrid demonstrated in this work provides an experimental apparatus for testing HPC integration with a real microgrid that includes HVAC and where the HPC systems have nonconstant Power Usage Effectiveness (PUE), incorporate several different HPC architectures, host a wide range of job core counts and walltime requests, and demonstrate significant load-follow jumps. The HPC integration with the microgrid enables testing scheduler strategies where a job queue forecast can alter the microgrid settings during production across multiple modes of grid operation.

The MIB class of microgrid is a specific class of a portable and programmable microgrid that can meet the high power quality standards for the day-to-day operation of an HPC datacenter. The settings of the microgrid are set according to the IEEE 1547-2018 standard [13] with inverter switch settings operating at the millisecond timescale. A picture of the MIB class microgrid used in this study is provided in Figure 4. The MIB class microgrid used in these experiments was designed for an operational window of up to 250 kW, but the components can be scaled to larger power loads. The MIB microgrid components are listed in Table I.

Fig. 4. An image of the MIB class microgrid used in the experiments in this work is shown here. Some of the components visible outside the container are breaker panels, inverters (white), transformers (gray), and HVAC for the microgrid, while the rest of the components reside inside the container. The component list for the microgrid is shown in Table I.

The microgrid can operate in grid forming or grid following modes and supports fast transition between the two modes. For HPC users, grid following is appealing due to the ability to limit the amount of power used from the grid at different times of the day or night based on factors such as the job queue, grid power cost, instantaneous carbon footprint of grid power, or amount of local DER power production including wind, solar, geothermal, micronuclear, biomass, or hydrogen. Some grid energy providers enable an interface for customers to indicate the power carbon footprint for power production in real time

which can be used to automatically alter the amount of power used from the grid by the microgrid when operating in grid following mode.

In the context of power for HPC systems, the MIB class microgrid not only supports the load-follow demands of the HPC systems but also the associated cooling needs for the datacenter. HPC power in the Top500 list [21] is typically reported in terms of the power used by the HPC servers, switches, and storage but not the power required for cooling the systems. The PUE metric captures the power that goes into cooling the HPC systems; the average PUE for a datacenter was 1.56 in 2024 [4]. This means that for every kW used to power the HPC systems, 560 Watts are needed to cool the systems. The power for cooling is not static and itself shows changes in load due to atmospheric conditions as well as HPC system usage. As an example, in Figure 5, the PUE for the datacenter hosting the system in Figure 1 is shown over the past year. Even though the system in Figure 1 is a direct liquid cooled system and the average PUE is much lower than the industry standard for 2024, the PUE still shows considerable variance throughout the year that will need to be supported by a microgrid power gateway. For air cooled datacenters, step-function like power behavior for cooling systems suggests the need for supporting peak shaving capability which is supported by the MIB class microgrid in this work.

The MIB class microgrid has multiple programmable settings and set points that could be altered by the HPC job scheduler, which has a complete view of current and pending jobs and available nodes. The most relevant settings of the microgrid for an HPC scheduler are the grid throttle, the DER source, the minimum battery state-of-charge, and the battery charge set points. These settings overall create both a time of use shifting capability and a point of common coupling control for all power inputs that can be driven by the HPC scheduler.

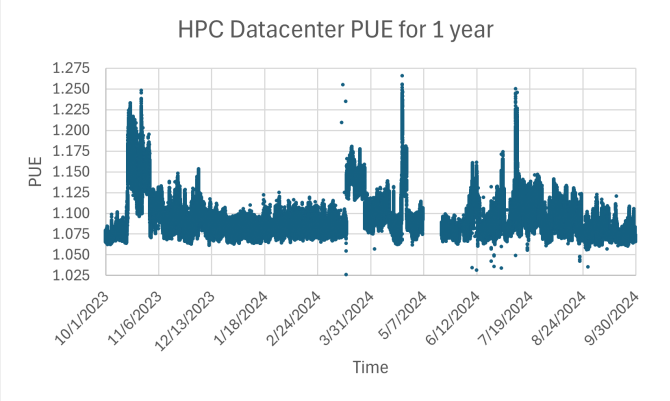

Fig. 5. The Power Usage Effectiveness (PUE) over a one year time period of the HPC datacenter hosting the HPC system shown in Figure 1. The PUE for an HPC system is not constant and itself shows significant load-follow jumps which would also need to be supported by the microgrid.

IV. Experimental Apparatus

The HPC system integrated with the MIB class microgrid consists of a wide mixture of commonly used HPC system architectures including ARM, GPU, and x86 based systems. The HPC systems are outlined in Table II. The ARM cluster is based off of the HPE Apollo 80 chassis holding four blades with each blade comprising two nodes with 48 cores per node. The GPU cluster is based off of the HPE Apollo 6500 chassis containing two nodes with each node having 32 cores and four NVIDIA A100 40GB GPUs. The x86 cluster is based off of a standard 2U 112 core node. Additionally, there are two Cisco Business 1 Gb Ethernet management switches, two Mellanox EDR switches, and two standard 1U 32 core head nodes (one for the x86/GPU clusters and one for the ARM cluster). Each of these systems have substantially different power profiles and power quality requirements. The power profiles of a single node of each of the architectures individually under idle load and full computational load are as follows:

- ARM - idle 105W load - 132W
- GPU - idle 669W load - 2277W
- x86 - idle 153W load - 562W

Because each system has different power quality limitations, using a variety of systems casts a wide net for capturing system specific failures due to power quality.

The HPC datacenter is a self-contained mobile datacenter (MDC) from World Wide Technology shown in Figure 6 that contains 3 racks and a cooling system that are all single source, main panel fed. The MDC uses 208V three-phase power with a total current capacity of 100A. The Fluke 1775 meter was attached at the primary 100A breaker in the MDC's main panel. The cooling system was a MovinCool Climate Pro K60. The PUE for the mobile datacenter at full load varied between 1.21 to 1.31 and averaged at 1.24 which is consistent with industry standard [26].

Fig. 6. The HPC mobile datacenter provided by World Wide Technology that was integrated with the MIB class microgrid.

Component	Quantity
DynaPower Converters - 150kW	2
Friedrich AC Units	3
ANG Converter Electronic Unit - 25kW	1
ANG Input Transformer SPCT - 25kW	1
ANG Output Transformer - 25kW	1
BluePlanet Energy Batteries - 32kWh	10
Square D HCP 21513 Panel 600V 3 x 400A Breakers [Generator, Load(s), Wye Side of Transformer] 1 x 200A Solar PV Breaker	1
Larson Panel 400A	1
10kV DeltaMaddox Industrial Transformer (Part Number: MIT-DRY-182)	1
Easter Owens High Voltage Cabinet	1
SEL 751	1
Power Supply	1
SEL/Ubiquiti Network and Controls Integration	1
Ageto Microgrid Controller with Mona Industrial PC containing Schneider Electric Software	1

TABLE I
DETAILED COMPONENT LIST OF THE MIB CLASS OF MICROGRID USED IN THE EXPERIMENTS HERE.

Fig. 7. The 64 kW solar farm that served as a DER input to the microgrid.

The experimental apparatus included a 64 kW solar farm shown in Figure 7 as well as 320 kWh in battery capacity. Power and power quality were measured at four locations via Fluke meters. Two different series of Fluke meters were used, three Fluke 430 Series II and one Fluke 1775. The Fluke meters measure frequency, voltage, THD, and power harmonics and provided an independent measure of power quality. Failures in the HPC systems themselves provide the most important power quality metric in the experiments. A detailed overview of the experimental apparatus and locations of power quality meters is provided in Figure 8.

The scheduling system for running jobs on the HPC systems is SLURM [29], and the mix of workloads originated from either the MOOSE framework [8] or MNIST digits training using Horovod [12].

V. Scheduler Integration with Microgrid

HPC schedulers provide a crucial link between the HPC system and the microgrid. For example, in Ding et al. [5] where a theoretical approach was presented integrating a

Type	ARM	GPU	x86
Node Count	32	6	5
CPU	1 x FUJITSU A64FX	1 x AMD EPYC 7543P	2 x AMD EPYC 7663
Cores	48	32	112
Memory	32 GB HBM	1 TB	1 TB
Network	EDR InfiniBand	Ethernet	EDR InfiniBand
GPU	N/A	4 x A100 40 GB	N/A

TABLE II
DETAILS FOR THE THREE CLUSTERS USED IN ALL COMPUTATIONAL EXPERIMENTS.

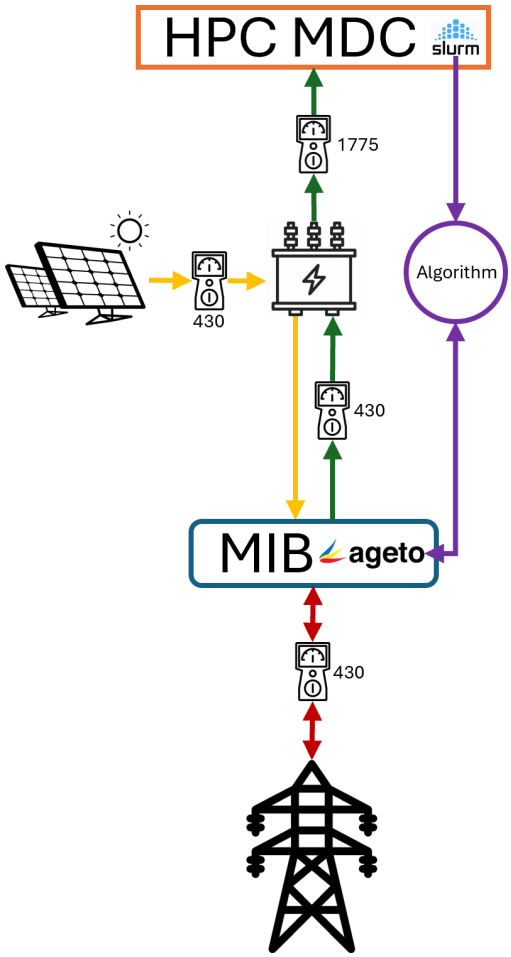

Fig. 8. Experimental layout with meter locations, mobile datacenter (MDC), power flow, and integration path of the job queue forecasting, and the microgrid controller using the Ageto software. Yellow arrows indicate the power flow for the solar, purple arrows indicate the communication for the job queue forecasting, red arrows indicate the path from the grid to the microgrid, and green arrows indicate the path of power from the microgrid to the MDC.

microgrid with a datacenter, a batch scheduler was also incorporated into the analysis. Similarly, in the reduction to practice presented here a power forecast originating from the job queue was integrated with the microgrid via one of three strategies to enable altering the settings of the microgrid based on the existing job queue. This strategy complements other existing research efforts where the job order and resource availability are altered by the scheduler based on available power input [25] but do not attempt to alter the settings of the grid providing power to the HPC system.

There have been many efforts to make HPC schedulers power aware so that the priority of a job is adjusted to optimize power consumption or cost [3, 24, 6]. Often times that approach requires knowing the power profile of the HPC application a priori, which complicates such efforts for large multi-user systems running hundreds of applications each with widely varying power profile signatures. The strategies explored here leverage a predicted load for the HPC system as a whole rather than the power profile for each HPC application individually. This information is then used to inform and alter the microgrid settings, made possible by the arrival of programmable microgrids for HPC with enormous flexibility in load shaping, peak shaving, mode switching, and power conditioning. Because the utility grid for most HPC system users is not configurable, this capability would have been very difficult to test in the past.

The most relevant settings of a microgrid which would benefit from power forecast information is the level of grid throttle and the trade-off between using excess DER generation to increase the battery state-of-charge or to sell the excess DER back to the grid. Another potential capability would be to allow the user to specify a carbon emission threshold for a submitted simulation in the scheduler and then adjust the microgrid load shaping settings to meet that threshold. The goal of the algorithm in this work is inform the microgrid settings in order to maximize renewable energy usage while avoiding outages. While not a focus of the research, this approach could also be easily adjusted in order to reduce utility grid charges for HPC operations.

Three strategies are explored along with a control case where the job queue did not impact the microgrid settings. The first strategy performs a 4 hour look-ahead resource usage prediction based on estimated start times. The grid throttle is then incrementally increased if the battery state-of-charge would drop below a specified set point. This strategy is detailed in Algorithm 1.

The second strategy performs a 12 hour look-ahead resource usage prediction and if the battery state-of-charge would drop below a specified set point, the grid throttle is significantly increased at a time when the renewable is not available and when utility grid power is the least expensive. This strategy is detailed in Algorithm 2.

The third strategy can be used in conjunction with the second strategy and is where any excess DER input is sold back to the utility grid when DER production exceeds a 2 hour look-ahead resource usage and the battery state-of-charge is above a provided threshold. A corollary to this strategy is where the excess power is used to improve battery state-of-charge rather than selling it back to the grid. This strategy and its corollary are detailed in Algorithm 3. The forecasting algorithm used in Algorithms 1–3 for the experiments was a time window-based power ratio estimation which reasonably predicts the power usage for the entire HPC system in the experiments here but would need refinement for experiments with more complex job mixes. The power is estimated based on the ratio of resources (cores/GPUs) that will be used at the end of the forecast time window to the power measured on those resources from a full system HPL or MLPerf run depending on the resource type and given in Section IV. The power measurement used for the ratio is the difference between the measured idle state and power consumed while at full HPL or MLPerf load. This way if no resources are used at the end of the time window, the power estimate function will still provide the idle power consumed by the resources. Individual HPC applications can exhibit power variability much larger than the MOOSE and distributed Horovod training job mix used in these experiments. In such cases, incorporating individual power profiles in the forecast from the job queue would be desirable for optimal microgrid settings.

Algorithm 1: Integrated microgrid - Strategy 1.

G	Current grid input (kW)
C	Current battery state-of-charge (%)
C_{min}	Battery state-of-charge minimum threshold (%)
P	Current solar array input (kW)
L	Current load (kW)
S	Predicted load from scheduler (kW)
T	Look ahead interval length (hours)

Data: $G, C, C_{min}, P, L, S, T$
Result: G

Function `PredictedStateOfCharge`$(C, \Delta L, T)$**:**
  B /* Max Battery Capacity (kWh)*/
  return $((B * C) - (\Delta L * T))/B$
Function `SetGridInput`$(G, C, C_{min}, P, L, S, T)$**:**
  if $L \geq P + G$ **then**
    $\Delta L \leftarrow L - (G + P)$
    if `PredictedStateOfCharge`$(C, \Delta L, T)$ $< C_{min}$ **then**
      ϵ /* Small buffer to allow for variance in power load (kW) */
      if $L - \epsilon < S < L + \epsilon$ **then**
        $G \leftarrow L \times 1.50$
      else
        $G \leftarrow S$
      end
    end
  end
  return G

VI. Results

The HPC system operated using the MIB class microgrid for 21 continuous days as shown in Figure 9 during which the integration strategies explored in section V were tested. The THD and power harmonics were monitored the entire time and no power quality issues were observed. As an example, in Figure 10 a significant power transition event is shown. In this event, the utility grid was throttled at 15 kW which was below the HPC system load even with peak shaving. The solar production gradually increased as mid-day approached

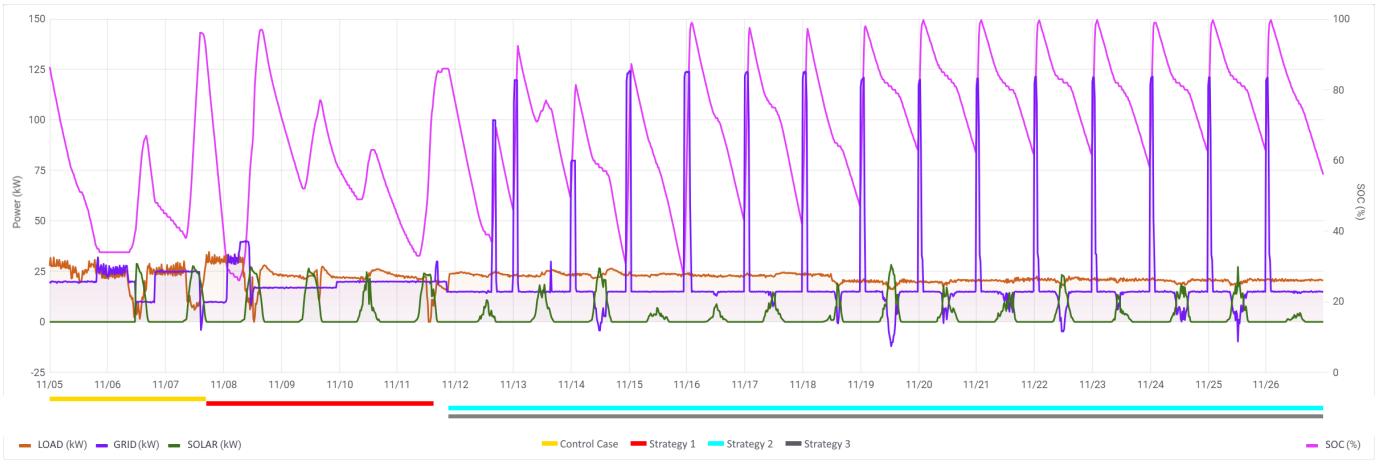

Fig. 9. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time for 21 days. All three algorithms were explored during this time as well as a control case where the microgrid and HPC job scheduler were uncoupled. The time windows during which the control case and algorithms were operational are shown in the color bars beneath the time axis. The control case was tested first followed by the three algorithms.

Algorithm 2: Integrated microgrid - Strategy 2.

G Current grid input (kW)
C Current battery state-of-charge (%)
C_{min} Battery state-of-charge minimum threshold (%)
P Current solar array input (kW)
L Current load (kW)
S Predicted load from scheduler (kW)
T Look ahead interval length (hours)

Data: $G, C, C_{min}, P, L, S, T$
Result: G
Function `PredictedStateOfCharge` $(C, \Delta L, T)$ **:**
  B /* Max Battery Capacity (kWh)*/
  return $((B * C) - (\Delta L * T))/B$
Function `SetGridInput` $(G, C, C_{min}, P, L, S, T)$ **:**
  if $L \geq P + G$ **then**
    $\Delta L \leftarrow S - (G + P)$
    if `PredictedStateOfCharge` $(C, \Delta L, T)$ $< C_{min}$ **and** $P = 0$ **and** $11:30p.m. \leq TimeOfDay \leq 2:00a.m.$ **then**
      $G \leftarrow \Delta G$ /* High Grid Threshold (kW) */
    end
  end
  return G

Algorithm 3: Integrated microgrid - Strategy 3.

G Current grid input (kW)
C Current battery state-of-charge (%)
C_{min} Battery state-of-charge minimum threshold (%)
P Current solar array input (kW)
L Current load (kW)
S Predicted load from scheduler (kW)
T Look ahead interval length (hours)

Data: $G, C, C_{min}, P, L, S, T$
Result: G
Function `PredictedStateOfCharge` $(C, \Delta L, T)$ **:**
  B /* Max Battery Capacity (kWh)*/
  return $((B * C) - (\Delta L * T))/B$
Function `SetGridInput` $(G, C, C_{min}, P, L, S, T)$ **:**
  if $L \leq P + G$ **then**
    $\Delta L \leftarrow S - (G + P)$
    if `PredictedStateOfCharge` $(C, \Delta L, T)$ $\geq C_{min}$ **then**
      $G \leftarrow -1$ /* Sell back power */
    else
      $G \leftarrow -2$ /* Charge batteries */
    end
  end
  return G

and quickly exceeded the HPC load; consequently, the utility grid usage dropped and the microgrid began selling power back to the grid since the battery state-of-charge was above the specified set point. As solar production dropped later in the day, the utility grid returned to the throttle set by the strategy which was operating under Algorithm 3 at the time. The voltage THD measured at the HPC system during this significant transition is shown in Figure 11 and remained below 3.5%.

The first two days of the 21 day run in Figure 9 show the operation of the HPC system and microgrid entirely decoupled from each other except for the microgrid reading the instantaneous load from the HPC system as a control case for comparison. The control case utilized the Ageto microgrid software for controlling its settings. The control case period is shown in Figure 12. The grid throttle is kept at 20 kW

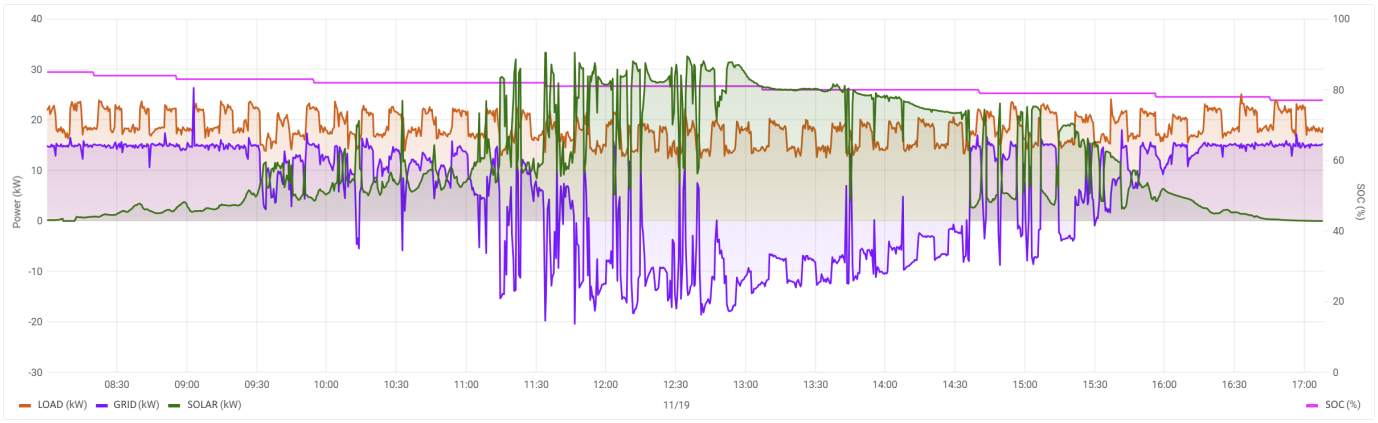

Fig. 10. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time where the microgrid feeds power back into the grid due to high solar production. The battery state-of-charge axis is shown on the right; the power load axis is shown on the left. In this portion of the run, the utility grid was throttled to 15 kW. As solar production increased during mid-day, power production exceeded the HPC load and the excess power was sold back to the utility grid since the battery state-of-charge was already at nearly 80%. The voltage THD for this segment of the run is shown in Figure 11, and power quality remained high throughout.

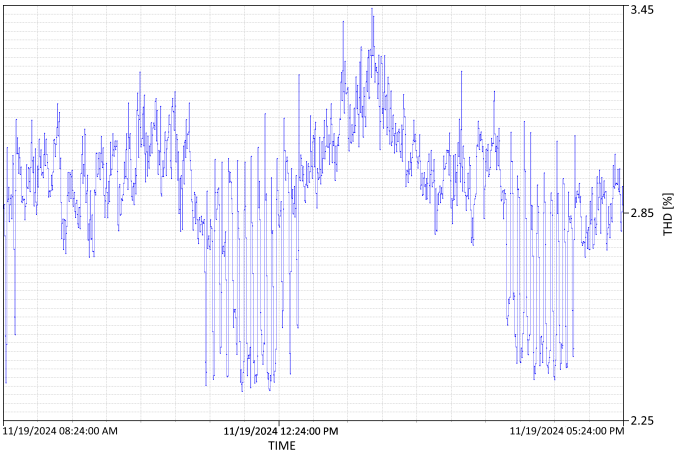

Fig. 11. Results showing the voltage THD as a function of time for the time period shown in Figure 10. THD was the highest during the point of time where the excess power was sold back to the utility grid but remained below 3.5%.

which is lower than the HPC instantaneous load resulting in a precipitous drop in battery state-of-charge. When the queue begins to clear out jobs to make space for larger runs shortly before an increase in solar power production, the grid throttle drops to 15 kW as the microgrid is not aware of the imminent significant increase in HPC load. This same behavior happens twice and would have resulted in the battery state-of-charge dropping below 25% during the second swing where damage might have occurred had the control case continued operating. For a brief moment the microgrid even sells some power back to the utility even while battery state-of-charge was low and a large HPC power swing was about to take place. Consequently, the control case was stopped at this point; if it had continued operating, an HPC outage would have resulted within three days. The control case with default microgrid settings would not have been able to operate the experimental apparatus for timescales of a week or longer without an outage.

Figure 13 shows four days over which Algorithm 1 controlled the microgrid settings. When the battery state-of-charge hit the lower threshold, the grid throttle was first increased to match the HPC load and then further incrementally increased in anticipation of an upcoming increase in HPC load. The throttle was adjusted on multiple occasions thereafter using the look-ahead load forecast of the HPC scheduler. This strategy was much more effective than the control case in maximizing renewable energy usage and maintaining a satisfactory battery state-of-charge in spite of four major abrupt changes in HPC load due to job scheduling.

Figure 14 shows nearly two days over which Algorithm 2 controlled the microgrid. When the HPC load forecast would lead to a battery state-of-charge below the set point threshold, the grid throttle was significantly increased for a short period at a time when renewable energy resources were not available. This strategy maximizes the battery state-of-charge as well as renewable energy usage. While Figure 14 shows a zoomed in window of the behavior, Algorithm 2 was used from 12 Nov mid-day until the end of the run in Figure 9 and reached steady-state operation within two days in spite of several HPC load changes. For the grid utility in this experiment, the least expensive times for power were expected between 11:30 P.M. and 2:00 A.M.

Figure 15 demonstrates Algorithm 3 using the HPC load to guide the decision when to recharge batteries from excess solar production or sell power back to the utility. In this time window, both modalities are present. This algorithm supports Algorithm 2 by reducing the amount of grid throttle needed to recharge batteries while also maximizing solar production.

While time of use shifting capability was present in both the control case and the three algorithm cases, the ability to consistently operate at a grid throttle lower than the HPC load was much better when using the algorithms. When examining kW difference between the HPC load and grid throttle during

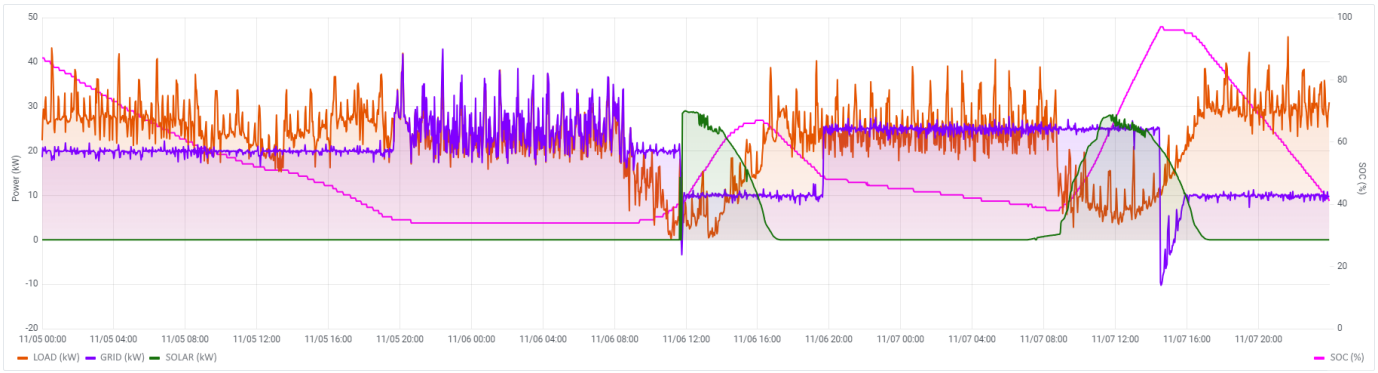

Fig. 12. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time for over two days during which the HPC job scheduler did not impact the microgrid settings. This serves as a control case and shows several problems when cores on the HPC system are reserved to support pending larger core count jobs and the microgrid prematurely reduces the grid throttle twice and even sells power back to the utility instead of improving the battery state-of-charge in spite of the large pending HPC load swing.

Fig. 13. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time for over four days during which the microgrid followed Algorithm 1 which created incremental changes to the grid throttle. When the battery state-of-charge hit the lower threshold, the grid throttle was first increased to match the HPC load and then further incrementally increased in anticipation of an upcoming increase in HPC load. The throttle was adjusted on multiple occasions using the look-ahead load forecast of the HPC scheduler. This strategy was much more effective than the control case in maximizing renewable energy usage and maintaining a satisfactory battery state-of-charge in spite of four major abrupt changes in HPC load due to job scheduling.

those times when the HPC load exceeds the grid throttle, the control case had difficulty operating the grid throttle at 25% below the HPC load whereas Algorithm 1 ran the grid throttle at 30% below HPC load and Algorithms 2-3 ran the grid throttle consistently at 40% below the HPC load.

VII. Discussion and Conclusions

This work has explored an HPC system integrated with a microgrid functioning as a power gateway. The microgrid was an MIB class microgrid composed of COTS components and the HPC system consisted of three common HPC architectures running workloads with a SLURM scheduler managing the allocation of resources. The microgrid included batteries as well as a solar farm and operated principally in a grid following mode for 21 continuous days with short occasions where power was sold back to the utility in a grid-tied mode. The power quality throughout the experiments met or exceeded the requirements of the HPC systems even during significant transitions of the microgrid away from grid following operation, and no hardware failures were observed. Three different strategies were tested during this experiment that controlled various settings of the microgrid. A control case where the HPC scheduler did not impact the microgrid settings was also included. The control case utilized the Ageto microgrid software (Figure 8 and Table I) for controlling its settings and did not maximize renewable energy usage for the HPC system. The control case had significant difficulty maintaining a healthy battery state-of-charge and would not have been able to operate for timescales of a week or longer without an outage.

The first strategy performed a 4 hour look-ahead of resource usage and incrementally increased the grid throttle if the battery state-of-charge dropped below a specified threshold. The results from this strategy show that it was capable of keeping the HPC systems functioning correctly and did allow for the battery state-of-charge to dip below the set threshold. However, it did come very close to the 25% hard limit for the battery state-of-charge where the microgrid Ageto software would have then overwrote the algorithm setting and imposed a grid throttle to maintain the battery state-of-charge to prevent damage to the batteries.

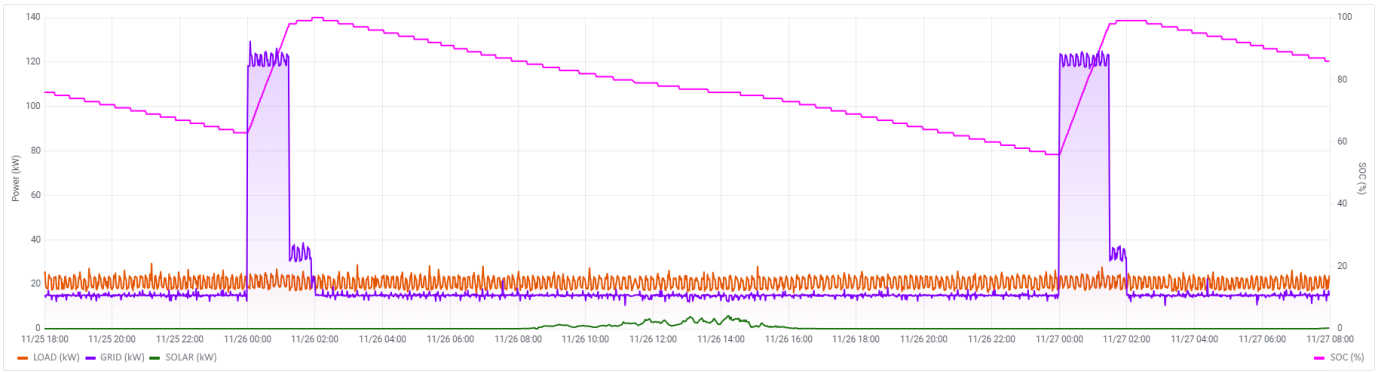

Fig. 14. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time for a day and a half period during which the microgrid followed Algorithm 2 where if the anticipated HPC job load would result in a poor battery state-of-charge forecast, the grid throttle would significantly increase for a relatively short period of time at points when renewable energy sources were not available. For the grid utility in this experiment, the least expensive times for power are expected between 11:30 P.M. and 2:00 A.M. During the time shown in this Figure, steady-state behavior had been mostly reached.

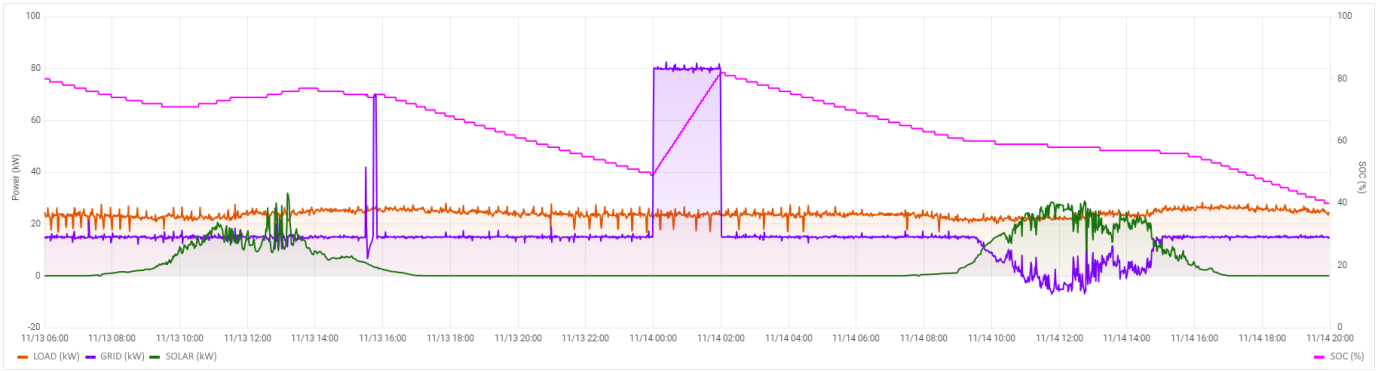

Fig. 15. Results showing the HPC and microgrid power load (LOAD), the utility grid (GRID), DER (SOLAR), and battery charge level (SOC) as a function of time for a day and a half period during which the microgrid followed Algorithms 2 and 3. Algorithm 3 uses the HPC load to guide the decision when to recharge batteries from excess solar production or sell power back to the utility. In this time window, both modalities are present. This algorithm supports Algorithm 2 by reducing the amount of grid throttle needed to recharge batteries while also maximizing solar production.

The second strategy performed a 12 hour look-ahead for resource usage and drastically increased the grid throttle if the battery state-of-charge would drop below a set threshold and if there were no DER available during the time window when power is expected to cost the least. The goal of this strategy was to charge the batteries as much as possible to maximize their utilization during the day along with DER power. This strategy reached a steady-state operation within a period of a little over two days indicating an ability to operate for long periods of time in this modality.

The third strategy performed a 2 hour resource usage forecast and was performed in conjunction with the second strategy. This strategy used the job queue to inform the microgrid whether to charge the batteries when there was excess solar production or to sell that excess production back to the utility. This strategy maximized the usage of the solar production after the second strategy achieved steady-state.

Using the microgrid as a flexible power gateway provides effective time of use shifting capability that is highly configurable. Integrating strategies which adjusted the microgrid settings into this real-time configuration nearly halved the level at which the grid throttle was able to operate compared to the control case. This maximized the renewable energy usage without any restriction on the distribution and types of jobs submitted by the user into the job queue.

For future work, the algorithms presented here would be more efficient with a better battery charge prediction function along with more advanced load forecasting including forecasting on the HVAC usage by the microgrid in addition to the HPC power cooling needs.

ACKNOWLEDGEMENTS

This research made use of Idaho National Laboratory's High Performance Computing systems located at the Collaborative Computing Center and supported by the Office of Nuclear Energy of the U.S. Department of Energy and the Nuclear Science User Facilities under Contract No. DE-AC07-05ID14517. We would like to thank Kurt Myers, Porter Hill, and Jeremiah Gilbert for providing access to the MIB class microgrid used in these experiments. We would like to thank Cody Grover for helpful conversations on power quality concerns for HPC architectures and Trad Day for technical support.

References

[1] Matthew Anderson and Matthew Sgambati. "High Performance Computing Peak Shaving for Microreactor Operation". In: *Practice and Experience in Advanced Research Computing 2024: Human Powered Computing*. PEARC '24. Providence, RI, USA: Association for Computing Machinery, 2024. ISBN: 9798400704192. DOI: 10.1145/3626203.3670517. URL: https://doi.org/10.1145/3626203.3670517.

[2] Matthew Anderson and Matthew Sgambati. "Microgrid Integration with High Performance Computing Systems for Microreactor Operation". In: *2024 IEEE International Conference on Cluster Computing Workshops (CLUSTER Workshops)*. 2024, pp. 44–54. DOI: 10.1109/CLUSTERWorkshops61563.2024.00017.

[3] Mohak Chadha, Jophin John, and Michael Gerndt. "Extending SLURM for Dynamic Resource-Aware Adaptive Batch Scheduling". In: Dec. 2020, pp. 223–232. DOI: 10.1109/HiPC50609.2020.00036.

[4] *Data center average Power Usage Effectiveness worldwide from 2007 to 2024*. https://www.statista.com/statistics/1229367/data-center-average-annual-pue-worldwide/. Oct. 2024.

[5] Zhaohao Ding et al. "Emission-Aware Stochastic Resource Planning Scheme for Data Center Microgrid Considering Batch Workload Scheduling and Risk Management". In: *IEEE Transactions on Industry Applications* 54.6 (2018), pp. 5599–5608. DOI: 10.1109/TIA.2018.2851516.

[6] Briag Dupont, Nesryne Mejri, and Georges Da Costa. "Energy-aware scheduling of malleable HPC applications using a Particle Swarm optimised greedy algorithm". In: *Sustainable Computing: Informatics and Systems* 28 (2020), p. 100447. ISSN: 2210-5379. DOI: https://doi.org/10.1016/j.suscom.2020.100447. URL: https://www.sciencedirect.com/science/article/pii/S2210537920301712.

[7] Jeremiah Gilbert et al. "High-Penetration Microgrids Providing Grid Stability Using Frequency-Watt Control". In: *2024 IEEE Green Technologies Conference (GreenTech)*. 2024, pp. 22–25. DOI: 10.1109/GreenTech58819.2024.10520610.

[8] Guillaume Giudicelli et al. "3.0 - MOOSE: Enabling massively parallel multiphysics simulations". In: *SoftwareX* 26 (2024), p. 101690. ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2024.101690. URL: https://www.sciencedirect.com/science/article/pii/S235271102400061X.

[9] T. et al. Grunloh. *Microgrid Modeling with Small Modular Reactors: Decarbonizing University Campus Microgrids through Optimal Deployment of Nuclear Power Reactors*. https://ws.engr.illinois.edu/sitemanager/getfile.asp?id=7057. July 2024.

[10] Porter Hill et al. "Inverter-Based Layered Volt-VAR Control Scheme for High-Penetration Microgrids". In: July 2024, pp. 1–5. DOI: 10.1109/PESGM51994.2024.10689208.

[11] Andres Honrubia-Escribano et al. "Influence of voltage dips on industrial equipment: Analysis and assessment". In: *International Journal of Electrical Power Energy Systems* 41 (Oct. 2012), pp. 87–95. DOI: 10.1016/j.ijepes.2012.03.018.

[12] *Horovod distributed training MNIST*. https://github.com/horovod/horovod/tree/master/examples/tensorflow2.

[13] "IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces". In: *IEEE Std 1547-2018 (Revision of IEEE Std 1547-2003)* (2018), pp. 1–138. DOI: 10.1109/IEEESTD.2018.8332112.

[14] *INL demonstrates microgrid in a box*. https://www.world-nuclear-news.org/Articles/INL-demonstrates-microgrid-in-a-box. July 2023.

[15] Hiroshi Irie et al. "The Sendai Microgrid Operational Experience in the Aftermath of the Tohoku Earthquake: A Case Study". In: *New Energy and Industrial Technology Development Organization* 308 (Feb. 2013), pp. 1–6.

[16] Alvin J. H. Lee et al. "Modeling Microreactor Requirements for High-Performance Computing". In: *Nuclear Technology* 210.6 (2024), pp. 1027–1041. DOI: 10.1080/00295450.2023.2276999.

[17] D. Michaelson and J. Jiang. "Review of integration of small modular reactors in renewable energy microgrids". In: *Renewable and Sustainable Energy Reviews* 152 (2021), p. 111638. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2021.111638. URL: https://www.sciencedirect.com/science/article/pii/S1364032121009138.

[18] *Microgrids created electric sanctuaries amidst the devastation of Hurricane Ian*. https://www.microgridknowledge.com/distributed-energy/article/11436860/microgrids-created-electric-sanctuaries-amidst-the-devastation-of-hurricane-ian. Oct. 2022.

[19] Xiaopu Peng et al. "Energy-efficient Management of Data Centers using a Renewable-aware Scheduler". In: *2022 IEEE International Conference on Networking, Architecture and Storage (NAS)*. 2022, pp. 1–8. DOI: 10.1109/NAS55553.2022.9925479.

[20] Arvind Sharma et al. "Comparative Analysis of Different Types of Micro-grid Architectures and Controls". In: *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*. 2018, pp. 1200–1208. DOI: 10.1109/ICACCCN.2018.8748491.

[21] *Top500: The List*. https://top500.org. Nov. 2023.

[22] Moslem Uddin et al. "Microgrids: A review, outstanding issues and future trends". In: *Energy Strategy Reviews* 49 (2023), p. 101127. ISSN: 2211-467X. DOI: https://doi.org/10.1016/j.esr.2023.101127. URL:

https://www.sciencedirect.com/science/article/pii/S2211467X23000779.

[23] Rosemarie Velik and Pascal Nicolay. "A cognitive decision agent architecture for optimal energy management of microgrids". In: *Energy Conversion and Management* 86 (2014), pp. 831–847. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2014.06.047. URL: https://www.sciencedirect.com/science/article/pii/S019689041400572X.

[24] Akshat Verma, Puneet Ahuja, and Anindya Neogi. "Power-aware dynamic placement of HPC applications". In: *Proceedings of the 22nd Annual International Conference on Supercomputing*. ICS '08. Island of Kos, Greece: Association for Computing Machinery, 2008, pp. 175–184. ISBN: 9781605581583. DOI: 10.1145/1375527.1375555. URL: https://doi.org/10.1145/1375527.1375555.

[25] Sean Wallace et al. "A Data Driven Scheduling Approach for Power Management on HPC Systems". In: *SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 2016, pp. 656–666. DOI: 10.1109/SC.2016.55.

[26] *What Is Data Center PUE? Defining Power Usage Effectiveness*. https://www.datacenterknowledge.com/sustainability/what-is-data-center-pue-defining-power-usage-effectiveness. Dec. 2024.

[27] Tae Ho Woo and Yun Il Kim. "Microgrid control incorporated with Small Modular Reactor (SMR)-based power productions in the university campus". In: *Progress in Nuclear Energy* 183 (2025), p. 105678. ISSN: 0149-1970. DOI: https://doi.org/10.1016/j.pnucene.2025.105678. URL: https://www.sciencedirect.com/science/article/pii/S0149197025000769.

[28] Xu Yang et al. "Integrating dynamic pricing of electricity into energy aware scheduling for HPC systems". In: *SC '13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. 2013, pp. 1–11. DOI: 10.1145/2503210.2503264.

[29] Andy B. Yoo, Morris A. Jette, and Mark Grondona. "SLURM: Simple Linux Utility for Resource Management". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Dror Feitelson, Larry Rudolph, and Uwe Schwiegelshohn. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 44–60. ISBN: 978-3-540-39727-4.