Front. Comput. Sci., 2022, 16(5): 165107
https://doi.org/10.1007/s11704-022-0625-8

REVIEW ARTICLE

Prediction of job characteristics for intelligent resource allocation in HPC systems: a survey and future directions

Zhengxiong HOU (✉)[1], Hong SHEN[2], Xingshe ZHOU[1], Jianhua GU[1], Yunlan WANG[1], Tianhai ZHAO[1]

1 Center for High Performance Computing, School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
2 School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 510275, China

Abstract Nowadays, high-performance computing (HPC) clusters are increasingly popular. Large volumes of job logs recording many years of operation traces have been accumulated. In the same time, the HPC cloud makes it possible to access HPC services remotely. For executing applications, both HPC end-users and cloud users need to request specific resources for different workloads by themselves. As users are usually not familiar with the hardware details and software layers, as well as the performance behavior of the underlying HPC systems. It is hard for them to select optimal resource configurations in terms of performance, cost, and energy efficiency. Hence, how to provide on-demand services with intelligent resource allocation is a critical issue in the HPC community. Prediction of job characteristics plays a key role for intelligent resource allocation. This paper presents a survey of the existing work and future directions for prediction of job characteristics for intelligent resource allocation in HPC systems. We first review the existing techniques in obtaining performance and energy consumption data of jobs. Then we survey the techniques for single-objective oriented predictions on runtime, queue time, power and energy consumption, cost and optimal resource configuration for input jobs, as well as multi-objective oriented predictions. We conclude after discussing future trends, research challenges and possible solutions towards intelligent resource allocation in HPC systems.

Keywords high-performance computing, performance prediction, job characteristics, intelligent resource allocation, cloud computing, machine learning

1 Introduction

Since the dawn of high-performance distributed computing, such as cluster computing, grid computing and cloud computing, resource allocation has been an essential part of the high-performance computing (HPC) systems. In most HPC clusters and supercomputers, local resource management and job scheduling software usually deal with rigid jobs. HPC end users specify the number of computing nodes and processors they need for rigid jobs, and run for a certain time using the required computing resources. Many production systems collect and archive their workload traces after dozens of years of operation [1−3]. In fact, in a common cluster computing system at universities, there may be millions of completed jobs within a year. In many cases, end users have to estimate a deadline for running a job, after which the job will be killed even if it has not been finished. So end users usually specify a much longer time limit for the job. The requested resources and estimated runtime by end users are utilized for job scheduling, such as the popular backfilling algorithms [4]. This mechanism usually leads a low productivity and low efficiency of HPC systems.

In the same time, cloud computing has been popular for enterprise services. Currently, HPC cloud [5] is emerging as an alternative to on-premise clusters for executing traditional scientific and engineering applications. Cloud computing users also need to request specific resources for different workloads by themselves. However, because they are usually not familiar with hardware details and software layers as well as performance behavior of the underlying cloud computing systems. The resource configurations they select are usually poor in performance, cost and energy efficiency [6].

Thus, it is important to study intelligent resource allocation that can improve the productivity and efficiency in HPC systems, such as clusters and HPC clouds. The metrics of productivity and efficiency are not only performance (such as resource utilization, job runtime, speedup and efficiency), but also the operation **cost** and energy efficiency [7] of the HPC systems.

Prediction of job characteristics plays a key role for intelligent resource allocation for providing on-demand services on HPC systems, especially on the emerging Exascale supercomputers and HPC clouds [5]. With the advent and popularization of big data processing technology and artificial intelligence (such as machine learning and deep learning) [8], it is interesting to utilize the cumulated large amount of data of

Received December 31, 2020; accepted January 5, 2022

E-mail: houzhx@nwpu.edu.cn

completed jobs and AI technologies for performance and energy consumption predictions of jobs. Thus, they can be helpful to optimize resource allocation and realize on-demand services for end-users and high efficiency for service providers.

Therefore, in this paper, we present a survey and future directions on prediction of job characteristics for intelligent resource allocation in HPC systems. There are a number of challenges for the prediction of job characteristics, such as how to obtain fine-grain performance and energy consumption data, how to predict the optimal resource configuration of jobs in terms of runtime, energy consumption and cost. This paper aims at helping service providers and users to understand the efforts in data driven prediction of job characteristics for intelligent resource allocation in HPC systems, including multi-core and many-core machines, clusters and supercomputers, grids and HPC clouds. We also discuss future research directions of intelligent resource allocation in HPC systems.

The rest of this paper is organized as follows. In Section 2, we introduce the background, including classification of main HPC workloads and HPC systems, and the optimization objectives of intelligent resource allocation. In Section 3, we present a classification and review of the methods for prediction of job characteristics for intelligent resource allocation, including collection of workload related data, prediction of job characteristics for optimizing resource allocation. In Section 4, we discuss future research directions. Section 5 concludes the paper.

2 Background

Resource allocation is a middleware service connecting the upper layer of workloads with the underlying HPC systems. In this section, we introduce a classification of the main HPC workloads and HPC systems, and the optimization objectives of intelligent resource allocation.

2.1 Classification of HPC workloads and HPC systems

From the perspective of general application domains, HPC workloads can be classified into traditional scientific and engineering computing applications, big data processing applications, and AI/machine learning/deep learning applications, etc. Traditional HPC applications can be further classified into some more specific applications domains, such as CFD (Computational Fluid Dynamics) and Materials Science. From the perspective of performance characteristics, traditional HPC applications can be divided into four categories: compute-intensive, memory-intensive, data-intensive, and communication-intensive. Some applications may have more than one character [9].

From the perspective of resource management and job scheduling, HPC workloads can be classified as rigid, moldable, malleable, and evolving jobs [10,11]. From the perspective of job types, the workloads can be classified into parallel applications, bag of independent tasks (BoT), and workflows, all represented by DAGs (Directed Acyclic Graphs). To implement parallel applications, typical parallel programming models include Message Passing Interface (MPI), Pthreads, OpenMP, OpenACC, athreads, OpenCL and Compute Unified Device Architecture (CUDA) for GPU (Graphics Processing Unit) programming, as well as hybrid MPI and OpenMP/CUDA/Pthreads, etc.

HPC systems mainly contain traditional HPC systems and HPC clouds. Currently, traditional HPC systems mainly refer to cluster computing systems and MPP (Massively Parallel Processing) systems, which may be further classified into multi-core homogeneous and heterogeneous systems (hybrid CPU and GPU, etc.). Cloud computing systems are usually composed of virtualized server clusters (clusters of virtual machines and containers), including public clouds and private clouds. Hybrid clouds [12] comprise on-premise clusters and remote cloud computing resources. What's more, clouds can be combined forming a cloud federation [13]. Especially, an HPC cloud refers to a cloud computing system for HPC workloads. Appropriate HPC cloud applications are mid-range tightly coupled and independent tasks, e.g., bag-of-tasks. Netto et al. [5] argue that large-scale tightly coupled MPI parallel applications are not suitable for virtualized HPC cloud.

2.2 Optimization objectives of intelligent resource allocation

In HPC systems, the main computing resources contain CPU/GPU, memory, storage, networking, and software resources. As described in Fig. 1, in HPC clusters and supercomputers, resource allocation usually refers to how many physical computing nodes are provided for a job, including mapping and execution of jobs based on selected resources. In HPC clouds, resource allocation usually refers to how many virtualized resources (virtual machines, containers, and bare metal nodes) are provided for a workload, including mapping and execution of cloud consumer workloads based on selected resources.

Table 1 lists the optimization objectives of intelligent resource allocation in HPC systems. From the perspective of users, the main objectives are to optimize performance (minimize job's runtime, turnaround time, or makespan) and minimize job execution cost. From the perspective of resource and service providers, the main objectives are to minimize energy consumption, maximize profit, job throughput, resource utilization and load balance. Both of them may pursue fairness, reliability, resilience and fault-tolerance [14].

Multi-objective optimization is also a popular research

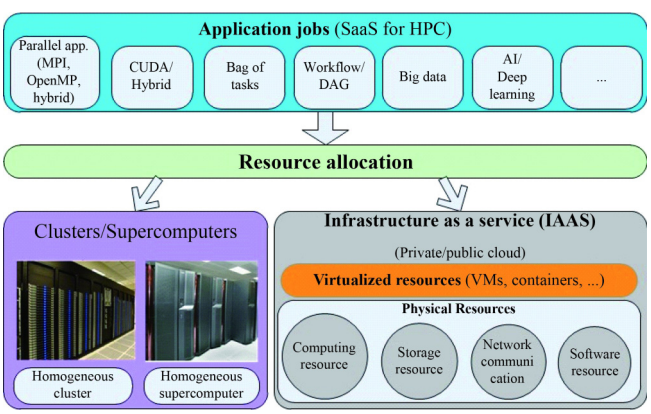

Fig. 1 Resource allocation for typical jobs in HPC systems

Table 1 Optimization objectives of intelligent resource allocation in HPC systems

Objectives	User	Provider
Performance/makespan/job throughput	Yes	Yes
Fairness	Yes	Yes
Resilience/fault-tolerance	Yes	Yes
Cost	Yes	
Power/energy/thermal/carbon emission		Yes
Resource utilization/load balance		Yes
Profit		Yes

topic. Multi-objective [14,15] may comprise makespan, economic cost, energy consumption, and resilience, etc. The ideal aim is to optimize multi-objective simultaneously. However, in many cases, we must tradeoff between conflicting objectives, such as performance and cost, performance and energy consumption. So the aim is to find pareto-optimal solutions in the two-objective space. In many cases, we set one objective as a constraint and optimize the other objective. For instance, when taking performance loss is a constraint, we can minimize energy consumption. Likewise, when energy consumption is a constraint, we optimize performance.

Service-level agreement (SLA) is defined for cloud providers [16]. In an on-demand HPC service approach on federated clouds, end users can customize cloud service selection strategies and SLAs [17]. On the basis of satisfying SLA, how to save energy and improve the profit and productivity for resource providers is a practical challenge for multi-objective optimization.

3 Prediction of job characteristics for intelligent resource allocation in HPC systems

Data driven prediction of job characteristics for intelligent resource allocation in HPC systems proceeds in two stages: (1) collecting workload related data, (2) prediction of job characteristics for optimizing resource allocation. Figure 2 shows its framework.

The main task of the first stage is to collect workload related resource, performance, power and energy consumption data. Economic cost of a job can be calculated based on its runtime and consumed resources. Cumulated workload and resource monitoring data are especially useful for machine learning based predictions. What's more, they can be used for the evaluation of prediction based resource allocation methods.

For the second stage, prediction of job characteristics mainly include job runtime and queue time prediction, power and energy consumption prediction, cost prediction, as well as optimal resource configuration prediction for performance, energy efficiency and cost. Here, the queue time of the jobs is determined by the runtime of running jobs, energy consumption is the product of power consumption and runtime, and cost is related with the price and runtime on allocated computing resources. So the runtime prediction can be used for the predictions of queue time, energy consumption and cost. And these predictions can be further used for the prediction of optimal resource configuration for corresponding goals, such as minimum runtime and cost, maximum energy efficiency.

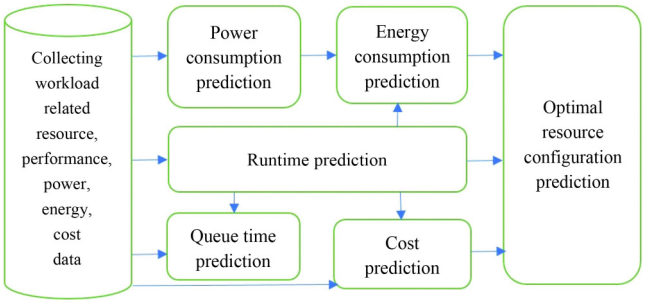

Fig. 2 Prediction of job characteristics for intelligent resource allocation in HPC systems

In traditional HPC clusters, supercomputers, HPC clouds, hybrid on-premise clusters and clouds, and federated clouds, the predictions can help implement intelligent resource allocation and resource selection for parallel workloads.

In the following subsections, we present the details of the representative works in each stage.

3.1 Collecting workload related data

In this subsection, we overview major methods and techniques for collecting workload related performance, hardware counters and energy consumption data.

3.1.1 Collecting resource related performance data

Feitelson et al. [1] propose Standard Workload Format (SWF). There are 18 properties to describe a job, such as runtime, wait time, and number of allocated processors. In an HPC cluster and supercomputer, there are already resource monitoring functions and job logs using the main local resource management systems, such as PBS, LSF, Condor, SLURM [18]. An overview of them is shown in Table 2. Resource monitoring may include static information of computing nodes, dynamic resource usage information about CPU, memory, disk I/O and network, etc. There are also some other special resource monitoring tools, for instance, Ganglia [19]. The job logs usually record performance data for each job. Large volumes of job logs on HPC production systems have been accumulated [1]. For example, Allcock et al. [20] collect the workload traces in 2013−2017 from the petascale supercomputer Mira. Yoon et al. [21] collect the job execution logs of two years from a Tachyon2 supercomputer at Korea. However, collecting and storing energy consumption data of jobs are still at a preliminary stage.

Cost of a job is usually counted on the basis of job runtime and consumed computing resources. In fact, most of service providers can account the billing of every user according to the completed jobs.

For a cloud data center, there are also some related work for collecting resource and workload related data. For instance, Islam et al. [22] collect the aggregated percentage of CPU usage of all the EC2 instances every minute. Cortez et al. [23] collect average CPU utilization, virtual CPU utilizations, lifetime of virtual machines, as well as deployment size in number of Virtual Machines (VMs) and CPU cores.

3.1.2 Collecting hardware event performance metrics

Other than resource related performance data, there are also

Table 2 Overview of resource monitoring and job logs in popular systems

Typical systems	Resource monitoring	Job logs (difference with SWF [1])	Open source and commercial
High throughput condor	Hostname, o.s., architecture, state, activity, average load, memory, activity time	Resource name and ID, keyword	Open source
LSF (IBM)	Hostname, status, r15s, r1m, r15m, ut, pg, ls, it, tmp, swp, mem, io	Project, command, work directory, submit host, output file, error file, execute host, job name	Commercial
SLURM	Partition, status, time limit, # of nodes, node list	Job name, number of used nodes, nodelist	Open source
PBS/Torque	Hostname, state, # of CPU cores, type, running jobs, load, physical memory, available memory, idle time, # of users, # of sessions	Job name, start time, end time, submit host, execute host	Commercial for PBS PRO; Open source for OpenPBS and Torque
Grid engine (Sun/Oracle grid engine)	Hostname, architecture, number of CPU/socket/cores/thread, load, total memory, used memory, total swap, used swap	Job name, CPU usage, io	Open source
HPC Pack (Microsoft)	Cores/memory/disk, processors, Cores, sockets, cores in use, CPU usage, affinity, status, workload, network	Job name, job template, project, priority, requested nodes, cores per node, licenses, environment variables, depends on jobs	Commercial

some tools collecting hardware event performance metrics in hardware performance counters, such as Intel Vtune [24], Performance Application Programming Interface (PAPI) [25], Perf [26], PerfMon2 [27], Likwid [28], and Hardware Performance Monitor (HPM) [29]. We introduce three commonly used tools. Intel Vtune [24] is a powerful commercial performance analysis software tool for Intel processors. PAPI [25] is an open source tool for various platforms. Perf [26] has been included in the Linux kernel, which can make a statistical analysis for specific applications and Linux kernel.

3.1.3 Collecting energy consumption data

There has been increasing attention to the power consumption and energy efficiency in HPC systems [30,31]. Some researchers [2] have collected both job logs and the corresponding power data for the experiments. There are already various sensors in HPC systems to monitor power consumption of their components, and some software libraries have been developed to access the power consumption data [32−35]. Some typical standard interfaces have been provided to monitor power consumption, such as Intelligent Platform Management Interface (IPMI) [36], Intel Running Average Power Limit (RAPL), and PAPI [37]. Baseboard Management Controller (BMC) is used by IPMI to monitor power consumption. RAPL estimates fine-grain energy usage by using hardware performance counters and I/O models [38]. PAPI is also based on RAPL interfaces. What's more, external power meters can be used to monitor a whole computing node, as well as other equipments, such as switches and routers. Leng et al. study how to measure and optimize the GPU power consumption [39].

To measure accurate power data of a parallel job, exclusive policy is necessarily used so that one computing node will not be shared with other jobs. For example, to obtain power data of a single job in a cluster using Cobalt scheduler [40], there is no sharing of hardware resources on the cluster. If a job shares a node with other jobs, it will be difficult to measure accurate power consumption of the job. Some researchers [41,42] use both RAPL and IPMI to monitor energy consumption by extending the resource utilization collection module of Simple Linux Utility for Resource Management (SLURM). With these extensions for energy accounting and control, SLURM can profile power consumption for each job. The overhead and error rate of energy monitoring in SLURM is acceptable, even on large-scale systems. On Cray systems, they also provide a Cray Advanced Platform Monitoring and Control (CAPMC) [43] module to monitor and control power consumption. Users can collect energy consumption and control power usage for each job in both command-line interface and HTTP APIs.

In a cloud data centre, they usually measure power consumption through PDU (Power Distribution Unit) and power meters. In this way, power usage effectiveness (PUE) can be calculated as the ratio of total power consumption to power consumption of IT devices.

3.1.4 Summary

In HPC clusters and supercomputers, there are usually job logs recording performance and resource data for each job with local resource management systems, such as PBS, LSF, [18] Condor [44], and SLURM [45]. /proc information and Linux commands are mainly used for collecting resource information. Some tools have been provided to measure performance counters, such as Intel Vtune [24], PAPI [25], and perf [26]. In cloud data centers, people usually collect performance data about CPU utilizations and lifetime of virtual machines, as well as deployment size in number of VMs and CPU cores. [22,23]

IPMI, RAPL and power meters are the main approaches to obtain power and energy consumption data for computing resources, such as CPU, memory, one computing node, one rack and even the whole HPC system. Based on IPMI and RAPL, SLURM and Cray systems have been extended to profile and control power usage for each job. The job logs and the power data should be linked [41−43]. The correlations among the collected data are analyzed based on the information per node gathered by a particular plugin.

However, in HPC clusters and supercomputers, it still needs further study about fine-grained power and energy consumption data. For example, it still needs to further gather and store power and energy consumption data for each job using SLURM and Cray systems. And it is still a challenge for other local resource management systems (PBS, LSF, etc.) to profile, gather and store power and energy consumption data for each job. The collected job logs should be cleaned. Correlated hardware events and energy consumption data may be used as extended parameters for better predictions. In HPC

cloud data centers, fine-grained power consumptions of virtual machines (VM) and containers also need further study for intelligent power management.

3.2 Runtime prediction

In this subsection, we overview the related work about runtime prediction in HPC clusters, supercomputers, HPC clouds, and hybrid environments.

3.2.1 Runtime prediction in clusters and supercomputers

In practice, to ensure that a job will not be killed before normal termination, end users usually specify a much longer time limit for the job. Thus, more accurate prediction of job runtime is needed to improve productivity and efficiency of resource allocation in HPC systems. The main existing approaches for job runtime prediction include category and statistics based prediction, application and system model based prediction, and machine learning based prediction.

Early in 1997, categorizing the jobs based on a set of rules was used for job runtime prediction [46]. The jobs were categorized according to their executable name, degree of parallelism, and user name. Then, statistics based prediction was studied. They predict the runtime of jobs based on the templates provided by the resource management system and system administrator. A genetic algorithm was proposed to evolve template attributes [47]. In these works, the used job information can be obtained in job logs.

Another method of runtime prediction is modeling of the jobs. For instance, Schopf et al. predict the runtime of applications based on their functional structure [48]. They study different workloads on a contended network of workstations. Mendes et al. perform static analysis of the applications [49]. These approaches do not consider dependencies between job submissions. Nissimov propose a stochastic model [50] for predicting job runtime distributions. This method only relies on previous runtime information, without needing job descriptions. Successive runtimes of a given user are treated as the observations of a Hidden Markov Model [51]. Thus, dependencies between job submissions are considered. Since users tend to estimate a much longer runtime, Tsafrir et al. [52] use system-generated predictions to improve backfilling policy. They use a simple method for the runtime prediction. The average runtime of the last two completed jobs by the same user is used as an estimate of the runtime of a new job. However, the experimental results indicate that the prediction is more accurate and efficient by using recent data rather than historical data of similar jobs.

Machine learning is an important method for job runtime prediction [53]. Matsunaga et al. [54] use linear regression based on the information about both applications and machines. A hybrid Bayesian-neural network approach is proposed by Duan et al. [55] to predict the runtime of parallel applications. It also uses information about historical jobs and machines. Gaussier et al. improve the popular backfilling policy using machine learning to predict job runtime [56]. They propose new cost functions based on a polynomial model. The proposed methods are evaluated through simulations using several job logs on production parallel computers. An online linear regression and k-Nearest Neighbors (kNN) based method [57] is proposed for efficient runtime prediction. McGough et al. [58] use random forest model for the prediction of job runtime and memory footprint in high-throughput computing systems. Ensemble learning [59,60] integrates several machine learning algorithms to predict job runtime in HPC clusters. Some machine learning algorithms [4,61] are proposed to improve runtime prediction accuracy while trying to reduce underestimates.

3.2.2 Performance prediction in HPC cloud and hybrid environments

Performance prediction is very important for optimizing resource allocation in cloud computing systems [5]. Almost all of the optimizations of resource allocation for HPC cloud and hybrid environment [62] benefit from performance and resource usage predictions.

HPC-as-a-Service Toolkit (HPCaaST) is proposed by Li et al. [63] to provide HPC services in the cloud. Time stamp of simulation time, current time, and the specification of a job are used for the estimation of job completion time. Shi et al. [64] use Amdahl's Law for instrumentation assisted program scalability analysis and performance prediction in different HPC systems, including a private HPC cloud environment. They separate communication time from computing to generate more trustworthy prediction results, and try to predict the optimal degree of parallelism. Saad et al. [65] propose an analytical model for predicting the running time of MPI based parallel jobs on an HPC cloud.

Cunha et al. [12] use kNN algorithm to predict job runtime and waiting time. They consider the inaccuracy of the performance predictions when making job placement decisions in hybrid HPC cloud environments. Due to different resource attributes of various cloud computing systems, it is more difficult to predict a job's runtime in federated multiple cloud platforms. Fan et al. exploit Rough Set Theory (RST) [66] to predict job runtime on the basis of historical data. The prediction accuracy is higher with more job records.

3.2.3 Summary of runtime prediction

Job runtime and performance prediction is important for resource allocation and job scheduling in clusters, supercomputers, HPC clouds and federated environments. In Table 3, we summarize and compare the performance prediction methods. Regarding job runtime prediction in traditional clusters and supercomputers, some typical methods include categorization and statistics based prediction, model based prediction, and machine learning based prediction. The first two methods are easy to implement, but the prediction accuracy is relatively low comparing with that of machine learning based methods. The specific machine learning algorithms mainly include supervised learning algorithms based on the information of history jobs, current jobs and machines, such as linear regression, random forests, and ensemble learning. They usually focus on the runtime prediction for a new job. In HPC clouds, Amdahl's law, simulations, and kNN algorithm are utilized for scalability and performance prediction. Rough Set Theory is also used to predict job runtime in hybrid clouds and federated cloud platforms.

Table 3 Overview of job runtime prediction methods for clusters, supercomputers and HPC clouds

Prediction methods	Refs.	Key techniques and inputs	Experimental data & platform	Prediction accuracy	Prediction stability	Difficulty & cost
Categorizing &statistics	[46]	Categorizing similar jobs template according to the executable name, degree of parallelism, and user name	Log files from 3 parallel computers	Coarse	Normal	Low
	[47]	A genetic algorithm evolving template attributes	4 workloads recorded from parallel computers	41%−71%	Normal	Normal
Application modeling	[48]	Stochastic values are used to parameterize performance models	Different workloads on a contended network of workstations	70%−95%	Normal	Normal
	[50]	Hidden markov model based on historical running time	Traces from several parallel computers	66.4%−99%	Normal	High
System modeling	[52]	Average runtime of the last two jobs by the same user is used for runtime estimation of a new job	4 traces from 4 parallel computers; an event-based simulation of scheduling	31%−62%	Normal	Low
	[63]	Based on the time stamp of simulation time, current time, and the specification of the job	Fire dynamics simulator; cloud	−90%	High	Normal
	[64]	Analysis based on Amdahl's law	NPB; traditional HPC cluster and a private HPC cloud	low	Normal	Normal
	[65]	Modeling the execution of an MPI job on a cluster using a queueing network	5 SPEC-MPI, 5 NPB; a cluster of bare-metal servers and virtual machines	88%	High	High
ML: HBNN	[55]	Running information of history jobs, job input parameters	The parallel workload archive; simulation with pyss	High	Normal	Normal
LR	[54]	Input parameters and machine information (e.g., disk speed)	BLAST and RAxML; 4 multi-core clusters	51.8%−89.2%	Normal	Normal
	[57]	Attributes of the current job and history jobs	4 real job logs; simulations	62.1%−82.3%	High	Low
Polynomial model	[56]	Runtime and requested resources of history jobs by the user	6 real job logs; simulations	Normal	Normal	Normal
RF	[58]	Task submission information	Trace-driven simulation; clusters using Condor	Normal	High	High
EL	[59]	Ensemble learning, LightGBM algorithm; RF, SVR, BRR, Bayesian model	3 job logs in HPC systems; VASP jobs on an HPC cluster	Normal	Normal	Normal
Tobit model	[4]	Tobit regression based TRIP algorithm	Workload traces from two IBM Blue Gene supercomputers	75%−80%	Normal	Normal
SVM	[60, 61]	Categorization and instance learning	2 real job logs (HPC2N04, ANL09)	70%−80%	Normal	High
kNN	[12]	Job submission information and free processors (it considers the uncertainty of the predictions)	Parallel job traces from real Supercomputing centers; hybrid on-premise cluster and HPC cloud	High	Normal	High
RST	[66]	Rough set theory	two computational jobs; multiple cloud platforms	High	Normal	High

ML: Machine learning, HBNN: Hybrid Bayesian-neural network, RF: Random forests, EL: Ensemble learning, kNN: k-nearest neighbors, LR: Linear regression, SVM: Support vector machine, BRR: Bayesian ridge regression, RST: Rough set theory.

Most methods has a prediction accuracy of job runtime less than 95%, and just predict the category of runtime [60]. The accuracy still needs to be improved, especially for selecting the optimal resource configuration to satisfy a given SLA. Online performance counters monitoring data may be adopted as the input features. In the same time, hardware and time cost of runtime predictions for multiple jobs need to be reduced to support intelligent resource allocation in real product HPC systems.

3.3 Queue-time prediction in clusters or supercomputers

Most parallel computers in HPC and supercomputing centers are shared by many users. Because computing resources are limited, in many cases, the submitted jobs have to wait in queues for some time before execution. Batch-queuing systems are usually used for job scheduling and resource allocation.

Smith et al. [67] predict wait-time by estimating runtime of running jobs. Job runtime predictions can also be used to optimize the performance of scheduling algorithms. Nurmi et al. [68,69] propose a statistical approach and a binomial method to predict queue wait times of large supercomputing centers. They can predict delay bounds for jobs in different queues, and for jobs requesting different number of processors. Nurmi et al. [70] can predict both runtime of the application and wait time of individual workflow tasks in batch queues. They predict quantiles directly based on wait time of historical jobs.

Utilizing on-premise clusters frequently entails long waits in queues. The queue analyzer [71] predicts the wait times for different jobs. They estimate queue bounds based on time series [68,72]. A major advantage of using cloud systems is resource availability. In theory, workloads do not need to wait for the available resources in queues. In the future, job queue systems may be introduced in HPC cloud systems due to the limitation of cloud computing resources.

To predict queue waiting times on parallel systems, Murali et al. [73,74] propose an algorithm based on spatial clustering using information of history jobs. The system state is represented by distributions or summary of features. A set of strategies are used for adaptively choosing the features and varying the related weights for each job prediction. A particular prediction algorithm is dynamically selected based on history jobs and the specific target.

We summarize common methods for predictions of job queue time in clusters and supercomputers in Table 4. The

Table 4 Overview of job queue time prediction methods for clusters and supercomputers

Prediction methods	Refs.	Key techniques/general comments	Experimental data & platform	Prediction accuracy	Prediction stability	Difficulty & cost
Runtime estimates	[67]	Predict wait time by runtime predictions based on history jobs	Workload traces from 3 supercomputer centers	Low	Normal	Normal
Statistical approach	[68]	Fit a statistical distribution to history jobs and use the distribution quantile of interest as the predictor for the next job	Batch jobs from 11 clusters over a 9-year period	Normal	High	Normal
Binomial method	[69]	Estimate an upper bound for the queuing delay with a quantified confidence level	7 archival job logs covering a 9-year period from large HPC centers	Normal	Normal	High
	[70]	Predict quantiles directly based on wait time of history jobs	Workflow on 5 supercomputers	Normal	Normal	High
	[68,71,72]	Estimate queue bounds from time series, predict quantile based on nonparametric inference	History jobs from 11 clusters over a 9-year period	Normal	High	Normal
Qespera	[73,74]	Spatial clustering using information of history jobs	Feitelson's parallel workloads archive from parallel computers	Most errors are less than 1 hour	Normal	High

existing approaches include job run-time based prediction, queue analysis based statistical approach, non-parametric inference method, spatial clustering based predictions using information of history jobs. However, the accuracy of job queue-time predictions can also be improved. Thus with more accurate prediction of job runtime and queue-time, we can improve the effectiveness of resource allocation.

3.4 Energy consumption prediction

In this subsection, we overview the related work about energy consumption predictions in both homogeneous and heterogeneous clusters and supercomputers.

3.4.1 Power and energy consumption prediction in homogeneous clusters and supercomputers

To improve energy efficiency, intelligent resource allocation relies on not only accurate performance prediction but also accurate power and energy prediction. Jin et al. [41] present a survey on software methods to improve the energy efficiency of parallel computing. They investigate different power-aware job scheduling algorithms [75−81]. Most of them extend the typical backfilling policy working online. Accurate prediction models of workload, power and energy are very important for these scheduling algorithms.

There are two main methods for predictions of power and energy consumption in homogeneous clusters and supercomputers: power modeling based prediction and power profiling.

(1) Power modeling based prediction

Dhiman et al. [82] use Gaussian Mixture Models for power modeling and predicting power consumption on a virtualized multi-core machine. The prediction model relies on not only CPU utilization, but also some other run-time architectural metrics, such as the number of Instructions Per Cycle (IPC) and the number of Memory accesses Per Cycle (MPC). The proposed model performs better than common regression methods.

Energy consumption of CPU and other components can also be estimated through different models [83,84]. Multi-variable regression was used to model the performance and power of the high-performance LINPACK (HPL) benchmark [85]. People can use performance counters to monitor specific performance metrics [86]. Performance counter information and power consumption models are used for the prediction of energy consumption in a lot of research work.

Kelechi et al. [8] review the artificial intelligence (AI) approaches for improving energy efficiency of HPC. They mentioned some AI-based energy prediction approaches, such as linear regression, support vector machine (SVM), and artificial neural network (ANN). Saillant et al. [40] propose an instance-based regression model using the submission data of resource and job management system to predict the power consumption of a job in HPC systems.

(2) Power profiling

Basically, HPC jobs have distinct power profiles. To control the power consumption of the entire system under a given budget, Wallace et al. [2] propose a data driven power-aware scheduler. They estimate job power profiles based on empirical analysis.

Patki et al. [87] propose hardware overprovisioning to work efficiently under the power constraint. Using overprovisioning, if all of the computing resources in a supercomputer are fully powered, power consumption of the supercomputer will exceed the power constraint. So only some system components can run at the peak power. System administrators need to dynamically reconfigure the system components using DVFS and power limit policies. The goal is to optimize the performance and energy efficiency of workloads under the power constraint. Patki et al. [88] utilize application profiles to predict the performance and power consumption of jobs. Job traces of different applications on clusters with hardware overprovisioning are used for experiments. On an overprovisioned supercomputer, Sarood et al. [89] use RAPL to constrain the power consumption of each node. They propose a power-aware scheduler for a supercomputer to maximize the job throughput under the power constraint. Mathematical regression model and application's profile data are used to provide a constant power consumption as the estimation. Ellsworth et al. [90] also propose a power-aware scheduler under the power constraint. They use a consistent value as the power consumption estimation of a job within a short time period.

3.4.2 Power and energy consumption prediction in heterogeneous clusters

Heterogeneous clusters usually use accelerators, such as hybrid CPU and general purpose GPU or MIC (Many Integrated Core) systems. Chiesi et al. [91] propose a power-aware scheduler on a hybrid CPU and GPU system. They use power profiles of characterized applications for predicting the power consumption of jobs. Using the predictions of power consumption, the jobs can be assigned to the available

computing nodes under the power constraint of each node.

Alina et al. [92] use regression methods with SVR (Support Vector Regression) for predicting power consumption of jobs on a hybrid CPU-GPU-MIC system. One regression analysis method is performed for each main resource component, i.e., CPU, GPU, MIC, Storage and Memory. Then power consumption of a job is estimated as the sum of the predicted power consumptions of the components. Specifically, SVR with Radial Basis Function kernels is used.

Ciznicki et al. [93] predict energy consumption of a stencil workload by approximating the energy usage of stencil operations, i.e., arithmetic operations and memory operations. Based on the predicted energy consumption, they use an integer linear programming method to minimize the energy usage of jobs on a hybrid CPU and GPU node.

In Table 5, we summarize the energy consumption prediction methods for jobs in homogeneous and heterogeneous clusters and supercomputers. The main approaches include power modeling based prediction, power profiling, and machine learning. Power modeling can be based on the power consumption of computing components, such as CPU, memory, disk, GPU and MIC. It can also utilize CPU utilization, IPC, MPC and performance counters and regression models. Most of them are used for the estimation of power and energy consumption at runtime. Before the execution of jobs, power profiling and empirical values can be used for the prediction of power and energy consumption. Some work need to be further studied, for example, how to predict the resource configuration for optimal energy efficiency.

There are also some related work of energy prediction in data centers. For instance, Dayarathna et al. [94] present a survey about the techniques used for energy consumption modeling and prediction in data centers. Lee et al. [95] propose a proactive thermal-aware virtual machine allocation method in HPC cloud datacenters. On the basis of monitoring the thermal at different regions, they can predict future temperature in a cloud datacenter. However, it still lacks research for energy consumption prediction of jobs in HPC clouds.

3.5 Cost prediction on HPC clouds

Cost is one of the main concerns in HPC clouds. Because cost prediction can be conducted based on performance prediction, there are relatively less literatures about cost prediction. And cost prediction is usually combined with performance prediction. A typical HPC user cares about the performance/cost ratio for running her jobs. Only high-end users care more about performance, even regardless of cost [96]. Machine learning models are utilized to predict performance/cost ratio of different cloud I/O configurations for HPC applications [97]. Using simulations and benchmarks during the application development phase, Rak et al. [98] can predict performance and costs of running bag-of-tasks scientific applications in the HPC cloud. Mariani et al. [6] propose a machine-learning approach to predict execution cost and performance of different cloud configurations.

3.6 Optimal resource configuration prediction

Other than the job runtime and energy consumption predictions based on given configurations by users, there are also studies on optimal resource configuration prediction for performance and energy efficiency. So users can select the predicted optimal resource configuration for a job, such as number of computing nodes/processes/threads, and CPU frequencies. Geist et al. [99] study energy constrained scheduling and adaptive parallelism in a survey of HPC scaling challenges. The optimal resource configuration for energy

Table 5 Overview of power/energy consumption prediction methods for jobs in clusters and supercomputers

Prediction methods	Refs.	Key techniques and inputs	Experimental workload and platform	Prediction accuracy	Prediction stability	Difficulty & cost
Power modeling	[40]	Instance-based regression model using submitted data	Logs of SLURM submission data of 12476 jobs on a COBALT supercomputer	Normal	Normal	Normal
	[82]	Regression models based on CPU utilization, IPC, MPC, and performance counters	Four benchmarks from the SPEC2000 suite; virtualized multi-core server	>90%	normal	High
	[83–86]	Multi-variable regression based on power of CPU, memory, disk, etc.	Wave2D, Jacobi2D, HPL; multi-core cluster	High	High	Normal
	[92]	Regression with SVR	Trace data from a prototype HPC system; heterogeneous clusters (CPU+GPU+MIC)	>80% Good	Normal	High
	[93]	Approximate the energy usage arithmetic operations and memory operations	Two simulation grids related to weather simulations; a CPU+GPU node	>90%	High	High
Power profiling	[2]	Estimate job power profiles based on monitoring power consumption of jobs	Trace-based simulations with one year of logs from an IBM Blue Gene/Q system	94%	High	Normal
	[88]	Using power profiles	job traces of different applications; clusters with hardware overprovisioning	87%	High	High
	[89]	Constant power	Wave2D, Jacobi2D, LeanMD, Lulesh, AMR; a 38-node Dell cluster	Normal	High	Low
	[90]	Consistent value	Simulation and data from Blue Gene/Q machine Mira	Normal	High	Low
	[91]	Using power profiles of characterized applications	10 workloads of 1000 jobs in the field of linear algebra; hybrid CPU and GPU	Normal	Normal	Normal
RF	[108]	Performance events, such as CPI, power of dram and processors.	NPBs and two co-design mini-applications; a dual-processor node	Processor 97.7%; DRAM 92%	High	High

RF: Random forests, kNN: k-Nearest Neighbors

efficiency may be predicted based on the collected data of history jobs. In the following subsections, we survey optimal resource configuration predictions for performance, energy efficiency and cost in multi-core and many-core machines, HPC clusters and supercomputers, and HPC clouds.

3.6.1 Multi-core and many-core machines

Wang et al. [100] predict the scalability of a program and the optimal number of threads for performance on multi-core machines using Artificial Neural Network (ANN). Based on the analysis of workload, Cochran et al. [101] propose polynomial logistic regression models to predict the optimal number of threads and DVFS settings for energy efficiency on a multi-core system. Gomatheeshwari et al. [102] propose a lightweight deep neural network (DNN) model for predicting the appropriate cores to enhance the energy efficiency on asymmetric multicore architectures.

On Intel Xeon Phi many-core coprocessor, Bai et al. [103], predict the optimal number of threads for performance based on "bytes per instruction". Ju et al. [104] predict the optimal number of threads for energy efficiency using Amdahl's law and regression analysis method. Both of them used least squares approach for the predictions. Lawson et al. [105] propose a general model that predicts the optimal resource configuration for performance and energy consumption, including number of threads, DVFS settings, number of MIC, and number of nodes. Offload mode of an accelerator is used for running the applications.

3.6.2 HPC clusters and supercomputers

Ge and Cameron [11,41] predict the optimal number of processors and frequencies for performance and energy-delay products for a given parallel application. The prediction method is based on Amdahl's law, accounting for parallel overhead.

Raghu et al. [31] predict the optimal frequency and voltage (DVFS settings) for energy consumption using decision tree based on knowledge and experience. A power aware scheduling algorithm is implemented to reduce energy consumption and enable load balance of HPC systems. Ozer, et al. [106] use random forest regression to predict CPU power and select the optimal CPU frequency for a job.

3.6.3 HPC Clouds

Based on machine-learning methods, Mariani et al. [6] propose a prediction model for the users to select the optimal cloud configuration when running their applications. The goal is to predict the optimal number of nodes, number of cores per node, and the amount of memory per node for performance and cost. They couple a performance prediction model on the cloud-provider side with a hardware independent profile-prediction model on the user-side. Niu et al. [107] dynamically control (expand or shrink) the size of the virtual cluster to optimize the cost. They make dynamic decisions weekly based on the prediction of future workloads using updated history logs. They extend the Exponential Smoothing method for time series data to implement the workload prediction.

In Table 6, we overview the optimal resource configuration prediction methods for jobs on multi-core and many-core processors, HPC clusters and supercomputers, and HPC clouds. There are some approaches to predict the optimal resource configurations for performance, energy efficiency and cost, such as number of threads and processes, DVFS. Existing prediction approaches commonly include a power-aware scalability model on the basis of Amdahl's law, multivariate regression techniques, knowledge base and decision tree, and machine learning methods. Most of them use NASA parallel benchmarks (NPB) and other parallel workloads for the experiments. Better prediction accuracy and integration of the prediction with real productive systems still need further studies.

3.7 Multi-objective optimizations oriented predictions

There are some approaches focusing on multi-objective optimizations. Most of them study performance and power/energy efficiency, such as predicting performance and energy at the same time, through the aggregated data (e.g., both performance counters and power measurements).

AutoMOMML [108] adopts a machine-learning based method to build performance and power prediction models. It uses the random forests algorithm for processor and DRAM power prediction.

Maury et al. [109] predict the optimal concurrency for performance when running multithreaded scientific codes on a multi-core server. Multivariate linear regression method and hardware counters are utilized for the prediction. It can also save energy using concurrency throttling. Within one node, multiple linear regression model [110] is also applied for the prediction of execution time and power consumption of multi-threaded parallel applications. The configurations mainly include the number of cores used by the application and the frequency of these cores.

Endrei et al. [111] predict the Pareto optimal performance and energy efficiency trade-off configurations when running hybrid MPI/OpenMP programs on a multi-core cluster. They adopt a B-spline piecewise polynomial prediction model using ordinary least squares regression. The configurations include the number of computing nodes, number of threads and CPU frequency.

Manumachu [112] et al. broadly classify the bi-objective optimization problem for performance and energy as system-level and application-level methods. Predicting the performance and energy consumption of applications is very important for them. Based on the problem size, they propose highly non-linear non-convex functions to represent the performance and dynamic energy consumption.

A multi-objective machine learning algorithm [113] is proposed to predict the Pareto-optimal powercap configurations for performance and energy, including processor and memory powercap settings.

4 Future research directions

With the development of computing capability and evolving applications, it is promising to further utilize artificial intelligence technologies and accumulated logs data for the prediction based intelligent resource allocation in HPC

Table 6 Overview of optimal resource configuration prediction methods in HPC systems

Prediction methods	Refs.	Optimization goal	Key techniques and inputs	Experimental application & platform	Prediction accuracy	Prediction stability	Difficulty & cost
ANN	[100]	O#TP	Code, data, and runtime features extracted from profiling executions	20 programs from UTDSP, NPB, Mibench; multi-core	Normal (<96-97%)	Normal	High
LR	[109]	O#TP	Collected data from hardware counters	10 NPB benchmarks using OpenMP, MM5; multi-core	median (87.4%)	High	Normal
PLR	[101]	O#TE, DVFS	Workload characteristics and resource information (ave. core temperature, frequency, stalls, etc.)	parallel workloads from the PARSEC suite; quad-core processor	>87.4%	Normal	Normal
Lightweight DNN	[102]	O#TE	Performance events, execution time and average power	MiBench, IoMT, Core-Mark workloads; ARM multicore	Up to 97%	High	High
BPI model	[103]	O#TP	Bytes per instruction model and least squares approach	NPB; MIC (Many-core)	Average 93.2%	High	High
Amdahl's law and regression analysis	[104]	O#TE	Regression analysis and the least squares method on the basis of the Amdahl's law	ten programs from the PARSEC suite; MIC (Many-core)	Normal	Normal	Normal
General model	[105]	O#TP, O#TE DVFS, #of MIC	Modeling runtime in the offload mode, power and energy consumption of all devices	CoMD proxy application; one and multiple nodes using MIC (Many-core)	>90%	Normal	Normal
Amdahl's law	[11], [41]	O#TP, O#TE DVFS	Power aware speedup model accounting for parallel overhead	NPB; a 16-node DVS-enabled cluster	Normal	Normal	Normal
Knowledge and experience	[31]	DVFS	Knowledge (decision tree) and experience based prediction	Linpack, SFM, NPB; a HP cluster with 5 nodes	Normal	Normal	Normal
RF	[106]	DVFS	Global extensible open power manager and dataleft database	Coral-2 suit; CooLMUC-3 cluster system	94%	High	High
Least squares regression	[111]	O#TP, O#TE, DVFS	Number of computer nodes, number of cores and threads, CPU frequency and DVFS settings	Stencil, Transpose, AMG and LAMMPS; a Cray system with 86 44-core compute nodes	>90%	Normal	Normal
Amdahl's and Gustafson's law	[64]	O#TP, O#TE	Asymptotic complexity models, timing models and separate communication from computation	NPB; a traditional HPC cluster and a private HPC cloud	Low	Normal	Normal
Exponential smoothing	[114]	O#VMC	Predict the future workload weekly by extending exponential smoothing method for time series data	parallel workloads archive and social network population traces; SLURM, Amazon EC2 and Google Cloud Engine	Good	Normal	High
RF; LR, ANN	[6]	O#TP, O#TC, OMP/OMC	About 200 (out of 935) features of operation mix, instruction level Parallelism, reuse distance, library calls, communication requirements	NASA parallel benchmarks (NPB); Openstack	>88% (CP); >70% (PP)	High	High

ANN: Artificial neural network, LR: multivariate linear regression, PLR: Polynomial logistic regression, RF: Random forests, O#TP/O#TE/O#TC: optimal number of threads/cores/nodes for performance/energy efficiency/cost, DVFS: optimal frequency and voltage for energy, O#VMC: controlling the size of the virtual cluster to optimize the cost, OMP/OMC: optimal amount of memory per node for performance/cost

systems. In Fig. 3, we present future research directions with related challenges for intelligent resource allocation in HPC systems.

From the perspective of HPC resource and service providers in the underlying hardware, cluster computing has been developed from tera-scale (10^{12} Flops), peta-sacle (10^{15} Flops) towards exa-scale (10^{18} Flops) and zeta-scale (10^{21} Flops) supercomputers. Cloud computing may be combined with on-premise clusters and supercomputers as hybrid environments. Grid computing [114−116] is evolving as cloud federation. Consequently, we need to study intelligent resource allocation in these new environments.

In the system middleware layer, prediction based intelligent approaches for multi-objectives (such as performance, energy efficiency, cost and resilience) [20] are promising comparing against traditional manual approaches and single objective (usually performance) optimization. End users specify the number of computing nodes and number of processors for the rigid jobs on traditional clusters. While moldable jobs can be executed using different numbers of processors. The allocated number of processors for each job can be modified before running the job. Malleable and evolving jobs can even change resource allocations (expand or shrink the size of computing nodes or processors) at runtime. Other than traditional rigid jobs, future HPC systems will also support simultaneous execution of moldable, malleable and evolving jobs. System logs will include not only a large volume of performance data but also power/energy consumption and cost data of workloads. There are some new challenges, such as how to improve multi-dimensional data quality and prediction accuracy with less cost for large volume of data driven predictions, as well as implementation of prediction based intelligent resource allocation in product HPC systems.

From the perspective resource consumers (HPC applications) in the upper layer, traditional parallel applications for science and engineering computing will still be popular, such as OpenMP, MPI, hybrid MPI and OpenMP/CUDA applications. There are also emerging new workloads, especially those coming from big data applications, Artificial Intelligence (AI) and deep learning applications, as well as sensor-based stream computing for Internet of Things. In some cases, there are hybrid workloads, such as hybrid HPC and big data applications, hybrid HPC and deep learning applications. Intelligent resource allocation for the new and hybrid HPC workloads is also a new research challenge.

The shaded regions in Fig. 3 are corresponding research

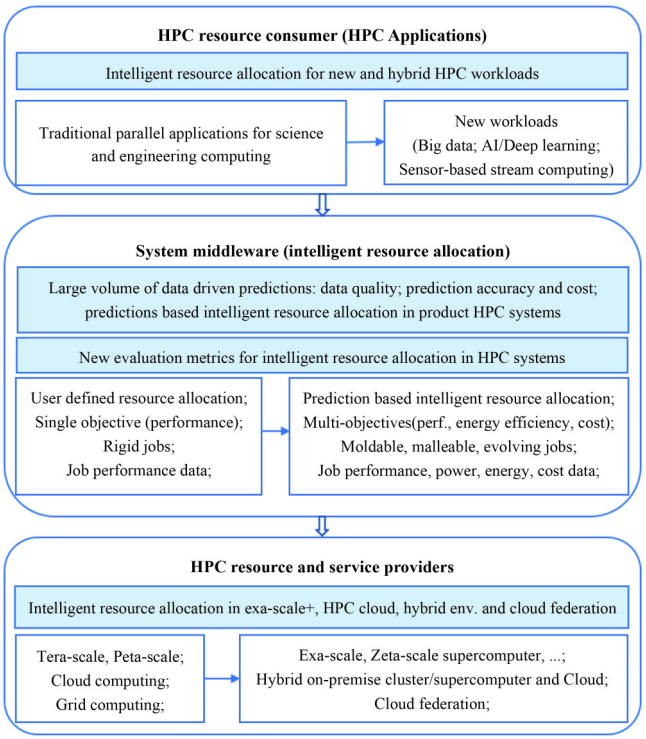

Fig. 3 Future research directions for intelligent resource allocation in HPC systems

challenges. In the following subsections, we discuss the specific research challenges.

4.1 Intelligent resource allocation in exa-scale and post-exa-scale supercomputers

In Exa-scale and post-Exa-scale supercomputers, large scale computing resources are basically heterogeneous and shared by different kinds of applications. They usually lead to massive energy consumption and error-prone parallel jobs. In many cases, parallel applications do not need a full-scale supercomputer. It is a challenge on how to provide on-demand HPC service by intelligent resource allocation for various applications.

Automatic resource allocation may be implemented based on the predictions. Or the requested resource can be intelligently modified before the execution of a moldable job. The optimization goals mainly include maximizing performance or energy efficiency, and multiple objectives according to the specific demands of end users.

System logs can be utilized for the proactive failure prediction on exa-scale supercomputers. Then, error-avoidance resource mapping may be implemented after filtering the error-prone nodes. Other specific techniques include prediction based heterogeneous resource allocation, power capping based job scheduling.

4.2 Intelligent resource allocation in HPC clouds, hybrid environments and federated clouds

Hybrid on-premise clusters (or in-house clusters) and clouds is a promising platform for HPC applications. Industry trends [5] show that hybrid environments are promising to combine the advantages of both on-premise and cloud resources. The users can run acceptable jobs on on-premise resources and overloaded jobs on cloud resources. However, there are several challenges:

(1) How to intelligently select on-premise resources and cloud resources according to specific scenarios?

(2) On cloud data centers, how to automatically and intelligently select the optimal resource configuration for a specific application from various configurations of virtual machines, containers, and bare-metal servers?

(3) On in-house clusters and cloud data centers, how to automatically allocate resources to the specific workloads according to SLA for different users?

(4) The resources are autonomous and the job logs may be sensitive for various cloud providers. How to use machine learning techniques for intelligent resource allocation on federated clouds?

Collecting operation data of clouds will be helpful to address the challenges. Data driven performance and energy prediction are also very important for intelligent resource allocation in this scenario. The predictions of job run time and wait time, as well as costs may help make a decision whether to select on-premise resources and cloud resources. The predictions can also help select the optimal configuration and allocate appropriate resources. Due to the data privacy of different cloud vendors, federated learning algorithms [117] may be applied for intelligent resource allocation on federated clouds.

4.3 Intelligent resource allocation for new and hybrid HPC workloads

In the HPC systems, other than traditional parallel applications for science and engineering computing, some new and hybrid applications are emerging.

Typical high-performance data analysis includes precision medicine, business intelligence, etc. MapReduce is a well-known framework for big data processing. Job placement is a basic problem in both MapReduce framework and HPC cluster [15]. Multi-objective optimization of performance and power consumption are studied for job placement in the MapReduce framework.

Machine learning and deep learning frameworks, such as Caffe and Tensorflow, may use CPU resources and GPU resources. Many classification and regression problems are continually improved [118]. AI applications have been emerging as one of the main HPC workloads (HPC for AI).

Cloud-Edge collaboration is promising to deal with massive data generated by Internet of Things [119]. Sensor-based streaming computing is another new workload in cloud-edge environment.

HPC systems can provide the fundamental computing capability for the new and hybrid workloads. However, the characteristics of them are quite different. It is also a challenging research direction how to provide on-demand services with intelligent resource allocation for the new HPC workloads, especially on the uniformed platform for hybrid HPC, big data processing, and AI applications.

One possible approach is static partition based resource allocation for the traditional and new workloads. Another

approach is dynamically created dedicated clusters (virtual machines, containers, physical or bare-metal nodes) for various workloads on a unified cloud computing platform. Multi-queue based workloads scheduling may be introduced. Each queue is built with customized resources for the specific application. For example, GPU resources are provided for AI and deep learning applications, and CPU resources are used for scientific and engineering float-point operations. Predictions of run time, energy consumption, and costs can also help allocate resources intelligently for the specific applications.

4.4 Multi-dimensional data driven predictions for intelligent resource allocation

For the intelligent resource allocation in different scenarios, it is obvious that predictions play a key role. On modern HPC clusters and supercomputers, as well as cloud data centers, there are more and more logs of resources and jobs, as well as system logs. Although it is promising to utilize these large volume of cumulated multi-dimensional data and machine learning technologies for performance, energy consumption and system failure predictions, there are also many new challenges in large volume data driven predictions in HPC systems (AI for HPC).

4.4.1 Multi-dimensional data quality

On in-house clusters and clouds, there are performance and power variability even for the same workload [31]. And there may be some errors in the job logs of local resource management systems. Some data may be missing, abnormal, or illogical in the job logs.

Performance, energy consumption and cost predictions rely on multi-dimensional data of history workloads. Data quality [1] of history workloads will impact the prediction accuracy. Thus data cleaning of history workloads is important for improving prediction accuracy. Other than removing the abnormal data, erroneous data, illogical data, incomplete data, and inconsistent data, we shall make sure that the data cleaning is effective. We may also use more techniques to improve data quality, such as reconstructing missing data and deep analysis of the correlations among the multi-dimensional data.

4.4.2 How to improve prediction accuracy with less cost and deal with the prediction errors?

Prediction accuracy is important for selecting the optimal resource configurations, such as the optimal number of computing nodes, optimal parallelism to obtain the best performance and energy efficiency for a specific application. However, the prediction accuracy still needs to be improved [12]. As described in Table 7, in the future, it needs more accurate performance and energy consumption predictions of queuing jobs for the intelligent resource allocation system to minimize hardware cost and time overhead.

In addition to cleaned job logs, more data from on-line hardware performance counters and characteristics of various applications should be considered to improve the prediction accuracy.

Adopting deep learning [58], reinforcement learning [120] and broad learning [121] for performance and energy consumption prediction is a promising approach to improve prediction accuracy. Deep learning algorithms have been very popular for image processing and voice recognition applications. It is also interesting to apply deep learning algorithms for processing large volume of performance and energy consumption logs, as well as system logs in HPC systems. Broad learning is a promising technique for reducing the hardware cost and time overheads of predictions.

Table 7 Some main concerns between current and future prediction of job characteristics in HPC systems

Concerns	Current	Future
Job logs	Performance and resource	+energy and cost
Features	Application and resource characteristics of history jobs	+online hardware counters
# of jobs	One job	+multiple jobs
Objective	Runtime	+energy and cost
Accuracy	Accurate	More accurate
Overhead	It depends	Less time overhead

"+" means something more than current status

Even we could use advanced algorithms based on good logs of history workloads, it is very hard to avoid prediction errors. Thus, how to deal with the prediction errors is one of the main challenges.

One approach is to prepare corresponding actions for different prediction errors. Naghshnejad and Singhal [122] proposed a hybrid backfilling and plan based scheduling platform, which not only predicts job runtimes but also estimates the prediction reliability. Jobs with high prediction reliability will be executed using a plan-based scheduling strategy, other jobs will be executed using the backfilling policy. This problem may also be solved to some extent by setting a certain amount of resource reservation space by some algorithms so as to ensure that the system maintains a stable state when there is a sudden increase in resource usage.

4.4.3 Predictions based intelligent resource allocation in product HPC systems

Currently, most of the prediction of job characteristics are utilized for resource allocation by simulations. In the future, to implement predictions based intelligent resource allocation in real product HPC systems, one approach is automatic resource allocation for moldable jobs [123]. The end users do not have to specify the resource requirements by themselves, such as the number of computing nodes, the number of cores per node, and memory requirements. The resource management system will automatically and intelligently allocate appropriate resources for each job according to the predictions so that user-requested computing resources can be intelligently optimized by the system.

Another approach is runtime intelligent adjustment of resource allocation for malleable jobs. In this case, after resource allocation based on performance and energy prediction before running a job, resource allocation can still be dynamically adjusted when running the job. So, if the optimization objectives are not achieved using the predefined

and predicted resources, we can still dynamically adjust resource allocation during runtime of jobs.

4.5 New evaluation metrics for intelligent resource allocation in HPC systems

Linpack, HPCG are used for the evaluation of Top 500 supercomputers [124] in the world. Performance per watt is used for the ranking of energy efficiency of supercomputers in the Green500 list [125]. They are benchmarks and metrics for evaluating the entire HPC system before productive operation with real applications. In productive HPC systems, there are usually some jobs of real applications sharing the HPC resources. It is necessary to study how to evaluate the runtime HPC systems with prediction based intelligent resource allocation. Previously, high productivity computing [126] has been studied. Most existing high-productivity computing models were studied between 2002 and 2006 in the DARPA HPCS program [127]. New comprehensive evaluation metrics for runtime HPC systems need to be further studied so that we can evaluate performance, energy efficiency [7], resilience and productivity for the whole runtime HPC systems with real applications.

5 Conclusion

In this paper, we present a survey on prediction of job characteristics for intelligent resource allocation in HPC systems, including clusters and supercomputers, HPC clouds and hybrid environments. We first discuss how to collect workload related performance and energy consumption data. Then, we overview the existing prediction methods for performance (runtime or queue wait time), power/energy consumption, cost, optimal resource configurations and multi-objectives. Finally, we highlight research challenges and future directions.

Although extensive research work has been carried out about prediction of job characteristics for optimizing resource allocation and scheduling in HPC systems, there are still a lot of work to be done, such as how to implement and evaluate intelligent resource allocation on Exa-scale supercomputers, hybrid on-premise cluster and clouds. It is also challenging on how to provide on-demand resources and services for new and hybrid HPC workloads. With the accumulated large volume of performance and energy consumption data, we need to study how to improve data quality and prediction accuracy.

Acknowledgements We would like to thank the anonymous reviewers for their valuable comments and suggestions. We also thank Dr./Prof. Feng Zhang from Renmin University of China and Dr./Prof. Jidong Zhai from Tsinghua University, China for their helpful suggestions and discussions. This work was partly supported by the National Key R&D Program of China (2018YFB0204100), the Science & Technology Innovation Project of Shaanxi Province (2019ZDLGY17-02), and the Fundamental Research Funds for the Central Universities.

References

1. Feitelson D G, Tsafrir D, Krakov D. Experience with using the parallel workloads archive. Journal of Parallel and Distributed Computing, 2014, 74(10): 2967–2982
2. Wallace S, Yang X, Vishwanath V, Allcock W E, Coghlan S, Papka M E, Lan Z. A data driven scheduling approach for power management on HPC systems. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2016, 56
3. Tsujita Y, Uno A, Sekizaw R, Yamamoto K, Sueyasu F. Job classification through long-term log analysis towards power-aware HPC system operation. In: Proceedings of the 29th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). 2021, 26–34
4. Fan Y, Rich P, Allcock W E, Papka M E, Lan Z. Trade-off between prediction accuracy and underestimation rate in job runtime estimates. In: Proceedings of the 2017 IEEE International Conference on Cluster Computing (CLUSTER). 2017, 530–540
5. Netto M A S, Calheiros R N, Rodrigues E R, Cunha R L F, Buyya R. HPC cloud for scientific and business applications: taxonomy, vision, and research challenges. ACM Computing Surveys, 2019, 51(1): 8
6. Mariani G, Anghel A, Jongerius R, Dittmann G. Predicting cloud performance for HPC applications before deployment. Future Generation Computer Systems, 2018, 87: 618–628
7. Orgerie A C, De Assuncao M D, Lefevre L. A survey on techniques for improving the energy efficiency of large-scale distributed systems. ACM Computing Surveys, 2014, 46(4): 47
8. Kelechi A H, Alsharif M H, Bameyi O J, Ezra P J, Joseph I K, Atayero A A, Geem Z W, Hong J. Artificial intelligence: an energy efficiency tool for enhanced high performance computing. Symmetry, 2020, 12(6): 1029
9. Wang E D. High Productivity Computing System: Design and Applications. China Science Publishing & Media Ltd, 2014
10. Prabhakaran S. Dynamic resource management and job scheduling for high performance computing. Technische Universität Darmstadt, Dissertation, 2016
11. Ge R, Cameron K W. Power-aware speedup. In: Proceedings of the 2017 IEEE International Parallel and Distributed Processing Symposium. 2007, 1–10
12. Cunha R L F, Rodrigues E R, Tizzei L P, Netto M A S. Job placement advisor based on turnaround predictions for HPC hybrid clouds. Future Generation Computer Systems, 2017, 67: 35–46
13. Leite A F, Boukerche A, De Melo A C M A, Eisenbeis C, Tadonki C, Ralha C G. Power-aware server consolidation for federated clouds. Concurrency and Computation: Practice and Experience, 2016, 28(12): 3427–3444
14. Yu L, Zhou Z, Fan Y, Papka M E, Lan Z. System-wide trade-off modeling of performance, power, and resilience on petascale systems. The Journal of Supercomputing, 2018, 74(7): 3168–3192
15. Blagodurov S, Fedorova A, Vinnik E, Dwyer T, Hermenier F. Multi-objective job placement in clusters. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2015, 66
16. Toosi A N, Calheiros R N, Buyya R. Interconnected cloud computing environments: challenges, taxonomy, and survey. ACM Computing Surveys, 2014, 47(1): 7
17. Hou Z, Wang Y, Sui Y, Gu J, Zhao T, Zhou X. Managing high-performance computing applications as an on-demand service on federated clouds. Computers & Electrical Engineering, 2018, 67: 579–595
18. Hussain H, Malik S U R, Hameed A, Khan S U, Bickler G, Min-Allah N, Qureshi M B, Zhang L, Wang Y, Ghani N, Kolodziej J, Zomaya A Y, Xu C Z, Balaji P, Vishnu A, Pinel F, Pecero J E, Kliazovich D, Bouvry P, Li H, Wang L, Chen D, Rayes A. A survey on resource allocation in high performance distributed computing systems. Parallel Computing, 2013, 39(11): 709–736
19. Massie M L, Chun B N, Culler D E. The ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing, 2004, 30(7): 817–840

20. Allcock W, Rich P, Fan Y, Lan Z. Experience and practice of batch scheduling on leadership supercomputers at Argonne. In: Proceedings of 21st Job Scheduling Strategies for Parallel Processing. 2017, 1−24
21. Yoon J, Hong T, Park C, Noh S Y, Yu H. Log analysis-based resource and execution time improvement in HPC: a case study. Applied Sciences, 2020, 10(7): 2634
22. Islam S, Keung J, Lee K, Liu A. Empirical prediction models for adaptive resource provisioning in the cloud. Future Generation Computer Systems, 2012, 28(1): 155–162
23. Cortez E, Bonde A, Muzio A, Russinovich M, Fontoura M, Bianchini R. Resource central: understanding and predicting workloads for improved resource management in large cloud platforms. In: Proceedings of the 26th Symposium on Operating Systems Principles. 2017, 153−167
24. Marowka A. On performance analysis of a multithreaded application parallelized by different programming models using Intel VTune. In: Proceedings of the 11th International Conference on Parallel Computing Technologies. 2011, 317−331
25. Terpstra D, Jagode H, You H, Dongarra J. Collecting performance data with PAPI-C. In: Proceedings of the 3rd International Workshop on Parallel Tools for High Performance Computing. 2009, 157−173
26. Dimakopoulou M, Eranian S, Koziris N, Bambos N. Reliable and efficient performance monitoring in Linux. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2016, 396−408
27. Weaver V M. Self-monitoring Overhead of the Linux perf_event performance counter interface. In: Proceedings of the 2015 IEEE International Symposium on Performance Analysis of Systems and Software. 2015, 102−111
28. Treibig J, Hager G, Wellein G. LIKWID: a lightweight performance-oriented tool suite for x86 multicore environments. In: Proceedings of the 39th International Conference on Parallel Processing Workshops. 2010, 207−216
29. Pospiech C. Hardware performance monitor (HPM) toolkit users guide. Advanced Computing Technology Center, IBM Research. See researcher.watson.ibm.com/researcher/files/us-hfwen/HPM_ug.pdf website, 2008
30. Georgiou Y, Glesser D, Rzadca K, Trystram D. A scheduler-level incentive mechanism for energy efficiency in HPC. In: Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. 2015, 617−626
31. Raghu H V, Saurav S K, Bapu B S. PAAS: power aware algorithm for scheduling in high performance computing. In: Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing. 2013, 327−332
32. Wallace S, Vishwanath V, Coghlan S, Tramm J, Lan Z, Papka M E. Application power profiling on IBM Blue Gene/Q. In: Proceedings of the 2013 IEEE International Conference on Cluster Computing (CLUSTER). 2013, 1−8
33. Browne S, Dongarra J, Garner N, Ho G, Mucci P. A portable programming interface for performance evaluation on modern processors. The International Journal of High Performance Computing Applications, 2000, 14(3): 189–204
34. Rashti M, Sabin G, Vansickle D, Norris B. WattProf: a flexible platform for fine-grained HPC power profiling. In: Proceedings of the 2015 IEEE International Conference on Cluster Computing. 2015, 698−705
35. Laros J H, DeBonis D, Grant R E, Kelly S M, Levenhagen M, Olivier S, Pedretti K. High performance computing-power application programming interface specification, version 1.2. See cfwebprod.sandia.gov/cfdocs/CompResearch/docs/PowerAPI_SAND_V1.1a(3).pdf website, 2016
36. Kavanagh R, Djemame K. Rapid and accurate energy models through calibration with IPMI and RAPL. Concurrency and Computation: Practice and Experience, 2019, 31(13): e5124
37. Weaver V M, Johnson M, Kasichayanula K, Ralph J, Luszczek P, Terpstra D, Moore S. Measuring energy and power with PAPI. In: Proceedings of the 41st International Conference on Parallel Processing Workshops. 2012, 262−268
38. Rotem E, Naveh A, Ananthakrishnan A, Weissmann E, Rajwan D. Power-management architecture of the Intel microarchitecture code-named Sandy Bridge. IEEE Micro, 2012, 32(2): 20–27
39. Leng J, Hetherington T, ElTantawy A, Gilani S, Kim N S, Aamodt T M, Reddi V J. GPUwattch: enabling energy optimizations in GPGPUs. In: Proceedings of the 40th Annual International Symposium on Computer Architecture. 2013, 487−498
40. Saillant T, Weill J C, Mougeot M. Predicting job power consumption based on RJMS submission data in HPC systems. In: Proceedings of the 35th International Conference on High Performance Computing. 2020, 63−82
41. Jin C, De Supinski B R, Abramson D, Poxon H, DeRose L, Dinh M N, Endrei M, Jessup E R. A survey on software methods to improve the energy efficiency of parallel computing. The International Journal of High Performance Computing Applications, 2017, 31(6): 517–549
42. Georgiou Y, Cadeau T, Glesser D, Auble D, Jette M, Hautreux M. Energy accounting and control with SLURM resource and job management system. In: Proceedings of the 15th International Conference on Distributed Computing and Networking. 2014, 96−118
43. Martin S J, Rush D, Kappel M. Cray advanced platform monitoring and control. In: Proceedings of the Cray User Group Meeting, Chicago, IL. See cug.org/proceedings/cug2015_proceedings/includes/files/pap132-file2.pdf website, 2015, 26−30
44. Thain D, Tannenbaum T, Livny M. Distributed computing in practice: the Condor experience. Concurrency and Computation: Practice and Experience, 2005, 17(2-4): 323–356
45. Yoo A B, Jette M A, Grondona M. SLURM: simple Linux utility for resource management. In: Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Processing. 2003, 44−60
46. Gibbons R. A historical application profiler for use by parallel schedulers. In: Proceedings of Workshop on Job Scheduling Strategies for Parallel Processing. 1997, 58−77
47. Smith W, Foster I, Taylor V. Predicting application run times with historical information. Journal of Parallel and Distributed Computing, 2004, 64(9): 1007–1016
48. Schopf J M, Berman F. Using stochastic intervals to predict application behavior on contended resources. In: Proceedings of the Fourth International Symposium on Parallel Architectures, Algorithms, and Networks. 1999, 344−349
49. Mendes C L, Reed D A. Integrated compilation and scalability analysis for parallel systems. In: Proceedings of the 1998 International Conference on Parallel Architectures and Compilation Techniques. 1998, 385−392
50. Nissimov A. Locality and its usage in parallel job runtime distribution modeling using HMM. Hebrew University, Dissertation, 2006
51. Rabiner L R. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 1989, 77(2): 257–286
52. Tsafrir D, Etsion Y, Feitelson D G. Backfilling using system-generated predictions rather than user runtime estimates. IEEE Transactions on Parallel and Distributed Systems, 2007, 18(6): 789–803
53. Hou Z, Zhao S, Yin C, Wang Y, Gu J, Zhou X. Machine learning based performance analysis and prediction of jobs on a HPC cluster. In: Proceedings of the 20th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT). 2019, 247−252
54. Matsunaga A, Fortes J A B. On the use of machine learning to predict

the time and resources consumed by applications. In: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. 2010, 495−504

55. Duan R, Nadeem F, Wang J, Zhang Y, Prodan R, Fahringer T. A hybrid intelligent method for performance modeling and prediction of workflow activities in grids. In: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. 2009, 339−347
56. Gaussier E, Glesser D, Reis V, Trystram D. Improving backfilling by using machine learning to predict running times. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2015, 1−10
57. Li J, Zhang X, Han L, Ji Z, Dong X, Hu C. OKCM: improving parallel task scheduling in high-performance computing systems using online learning. The Journal of Supercomputing, 2021, 77(6): 5960–5983
58. McGough A S, Moubayed N A, Forshaw M. Using machine learning in trace-driven energy-aware simulations of high-throughput computing systems. In: Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion. 2017, 55−60
59. Chen X, Zhang H, Bai H, Yang C, Zhao X, Li B. Runtime prediction of high-performance computing jobs based on ensemble learning. In: Proceedings of the 4th International Conference on High Performance Compilation, Computing and Communications. 2020, 56−62
60. Wu G B, Shen Y, Zhang W S, Liao S S, Wang Q Q, Li J. Runtime prediction of jobs for backfilling optimization. Journal of Chinese Computer Systems (in Chinese), 2019, 40(1): 6−12
61. Xiao Y H, Xu L F, Xiong M. GA-Sim: a job running time prediction algorithm based on categorization and instance learning. Computer Engineering & Science (in Chinese), 2019, 41(6): 987−992
62. Parashar M, AbdelBaky M, Rodero I, Devarakonda A. Cloud paradigms and practices for computational and data-enabled science and engineering. Computing in Science & Engineering, 2013, 15(4): 10–18
63. Li X, Palit H, Foo Y S, Hung T. Building an HPC-as-a-service toolkit for user-interactive HPC services in the cloud. In: Proceedings of the 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications. 2011, 369−374
64. Shi J Y, Taifi M, Pradeep A, Khreishah A, Antony V. Program scalability analysis for HPC cloud: applying Amdahl's law to NAS benchmarks. In: Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis. 2012, 1215−1225
65. Saad A, El-Mahdy A. HPCCloud seer: a performance model based predictor for parallel applications on the cloud. IEEE Access, 2020, 8: 87978–87993
66. Fan C T, Chang Y S, Wang W J, Yuan S M. Execution time prediction using rough set theory in hybrid cloud. In: Proceedings of the 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing. 2012, 729−734
67. Smith W, Taylor V E, Foster I T. Using run-time predictions to estimate queue wait times and improve scheduler performance. In: Proceedings of the Job Scheduling Strategies for Parallel Processing. 1999, 202−219
68. Nurmi D, Brevik J, Wolski R. QBETS: queue bounds estimation from time series. In: Proceedings of the 13th Workshop on Job Scheduling Strategies for Parallel Processing. 2007, 76−101
69. Brevik J, Nurmi D, Wolski R. Predicting bounds on queuing delay for batch-scheduled parallel machines. In: Proceedings of the 11th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 2006, 110−118
70. Nurmi D, Mandal A, Brevik J, Koelbel C, Wolski R, Kennedy K. Evaluation of a workflow scheduler using integrated performance modelling and batch queue wait time prediction. In: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing. 2006, 29
71. Netto M A S, Cunha R L F, Sultanum N. Deciding when and how to move HPC jobs to the cloud. Computer, 2015, 48(11): 86–89
72. Smith W. A service for queue prediction and job statistics. In: Proceedings of the 2010 Gateway Computing Environments Workshop (GCE). 2010, 1−8
73. Murali P, Vadhiyar S. Qespera: an adaptive framework for prediction of queue waiting times in supercomputer systems. Concurrency and Computation: Practice and Experience, 2016, 28(9): 2685–2710
74. Murali P, Vadhiyar S. Metascheduling of HPC jobs in day-ahead electricity markets. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(3): 614–627
75. Elnozahy E N, Kistler M, Rajamony R. Energy-efficient server clusters. In: Proceedings of the 2nd International Workshop on Power-aware Computer Systems. 2002, 179−197
76. Lawson B, Smirni E. Power-aware resource allocation in high-end systems via online simulation. In: Proceedings of the 19th Annual International Conference on Supercomputing. 2005, 229−238
77. Etinski M, Corbalan J, Labarta J, Valero M. Optimizing job performance under a given power constraint in HPC centers. In: Proceedings of the International Conference on Green Computing. 2010, 257−267
78. Etinski M, Corbalan J, Labarta J, Valero M. Parallel job scheduling for power constrained HPC systems. Parallel Computing, 2012, 38(12): 615–630
79. Mämmelä O, Majanen M, Basmadjian R, De Meer H, Giesler A, Homberg W. Energy-aware job scheduler for high-performance computing. Computer Science - Research and Development, 2012, 27(4): 265–275
80. Zhou Z, Lan Z, Tang W, Desai N. Reducing energy costs for IBM Blue Gene/P via power-aware job scheduling. In: Proceedings of the 17th Workshop on Job Scheduling Strategies for Parallel Processing. 2014, 96−115
81. Marathe A, Bailey P E, Lowenthal D K, Rountree B, Schulz M, De Supinski B R. A run-time system for power-constrained HPC applications. In: Proceedings of the 30th International Conference on High Performance Computing. 2015, 394−408
82. Dhiman G, Mihic K, Rosing T. A system for online power prediction in virtualized environments using gaussian mixture models. In: Proceedings of the 47th Design Automation Conference. 2010, 807−812
83. Basmadjian R, De Meer H. Evaluating and modeling power consumption of multi-core processors. In: Proceedings of the 3rd International Conference on Future Systems: Where Energy, Computing and Communication Meet (e-Energy). 2012, 1−10
84. Basmadjian R, Costa G D, Chetsa G L T, Lefevre L, Oleksiak A, Pierson J M. Energy-aware approaches for HPC systems. In: Jeannot E, Žilinskas J, eds. High-Performance Computing on Complex Environments. Hoboken: John Wiley & Sons, Inc, 2014
85. Subramaniam B, Feng W C. Statistical power and performance modeling for optimizing the energy efficiency of scientific computing. In: Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing. 2010, 139−146
86. John L K, Eeckhout L. Performance Evaluation and Benchmarking. New York: CRC Press, 2005
87. Patki T, Lowenthal D K, Rountree B, Schulz M, De Supinski B R. Exploring hardware overprovisioning in power-constrained, high performance computing. In: Proceedings of the 27th International ACM Conference on International Conference on Supercomputing. 2013, 173−182

88. Patki T, Lowenthal D K, Sasidharan A, Maiterth M, Rountree B L, Schulz M, De Supinski B R. Practical resource management in power-constrained, high performance computing. In: Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing. 2015, 121−132
89. Sarood O, Langer A, Gupta A, Kale L. Maximizing throughput of overprovisioned HPC data centers under a strict power budget. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2014, 807−818
90. Ellsworth D A, Malony A D, Rountree B, Schulz M. Dynamic power sharing for higher job throughput. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2015, 80
91. Chiesi M, Vanzolini L, Mucci C, Scarselli E F, Guerrieri R. Power-aware job scheduling on heterogeneous multicore architectures. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(3): 868–877
92. Sîrbu A, Babaoglu O. Power consumption modeling and prediction in a hybrid CPU-GPU-MIC supercomputer. In: Proceedings of the 22nd European Conference on Parallel Processing. 2016, 117−130
93. Ciznicki M, Kurowski K, Weglarz J. Energy aware scheduling model and online heuristics for stencil codes on heterogeneous computing architectures. Cluster Computing, 2017, 20(3): 2535–2549
94. Dayarathna M, Wen Y, Fan R. Data center energy consumption modeling: a survey. IEEE Communications Surveys & Tutorials, 2016, 18(1): 732–794
95. Lee E K, Viswanathan H, Pompili D. VMAP: proactive thermal-aware virtual machine allocation in HPC cloud datacenters. In: Proceedings of the 19th International Conference on High Performance Computing. 2012, 1−10
96. Aversa R, Di Martino B, Rak M, Venticinque S, Villano U. Performance prediction for HPC on clouds. In: Buyya R, Broberg J, Goscinski A, eds. Cloud Computing: Principles and Paradigms. Hoboken: John Wiley & Sons, Inc, 2011
97. Liu M, Jin Y, Zhai J, Zha Y, Shi Q, Ma X, Chen W. ACIC: automatic cloud I/O configurator for HPC applications. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. 2013, 1−12
98. Rak M, Turtur M, Villano U. Early prediction of the cost of cloud usage for HPC applications. Scalable Computing: Practice and Experience, 2015, 16(3): 303–320
99. Geist A, Reed D A. A survey of high-performance computing scaling challenges. The International Journal of High Performance Computing Applications, 2017, 31(1): 104–113
100. Wang Z, O'Boyle M F P. Mapping parallelism to multi-cores: a machine learning based approach. In: Proceedings of the 14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 2009, 75−84
101. Cochran R, Hankendi C, Coskun A, Reda S. Identifying the optimal energy-efficient operating points of parallel workloads. In: Proceedings of the 2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). 2011, 608−615
102. Gomatheeshwari B, Selvakumar J. Appropriate allocation of workloads on performance asymmetric multicore architectures via deep learning algorithms. Microprocessors and Microsystems, 2020, 73: 102996
103. Bai X, Wang E, Dong X, Zhang X. A scalability prediction approach for multi-threaded applications on manycore processors. The Journal of Supercomputing, 2015, 71(11): 4072–4094
104. Ju T, Wu W, Chen H, Zhu Z, Dong X. Thread count prediction model: dynamically adjusting threads for heterogeneous many-core systems. In: Proceedings of the 21st IEEE International Conference on Parallel and Distributed Systems. 2015, 456−464
105. Lawson G, Sundriyal V, Sosonkina M, Shen Y. Modeling performance and energy for applications offloaded to Intel Xeon Phi. In: Proceedings of the 2nd International Workshop on Hardware-Software Co-Design for High Performance Computing. 2015, 7
106. Ozer G, Garg S, Davoudi N, Poerwawinata G, Maiterth M, Netti A, Tafani D. Towards a predictive energy model for HPC runtime systems using supervised learning. In: Proceedings of the European Conference on Parallel Processing. 2019, 626−638
107. Niu S, Zhai J, Ma X, Tang X, Chen W, Zheng W. Building semi-elastic virtual clusters for cost-effective HPC cloud resource provisioning. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(7): 1915–1928
108. Balaprakash P, Tiwari A, Wild S M, Carrington L, Hovland P D. AutoMOMML: automatic multi-objective modeling with machine learning. In: Proceedings of the 31st International Conference on High Performance Computing. 2016, 219−239
109. Curtis-Maury M, Blagojevic F, Antonopoulos C D, Nikolopoulos D S. Prediction-based power-performance adaptation of multithreaded scientific codes. IEEE Transactions on Parallel and Distributed Systems, 2008, 19(10): 1396–1410
110. De Sensi D. Predicting performance and power consumption of parallel applications. In: Proceedings of the 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP). 2016, 200−207
111. Endrei M, Jin C, Dinh M N, Abramson D, Poxon H, DeRose L, De Supinski B R. Energy efficiency modeling of parallel applications. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2018, 212−224
112. Manumachu R R, Lastovetsky A. Bi-objective optimization of data-parallel applications on homogeneous multicore clusters for performance and energy. IEEE Transactions on Computers, 2018, 67(2): 160–177
113. Hao M, Zhang W, Wang Y, Lu G, Wang F, Vasilakos A V. Fine-grained powercap allocation for power-constrained systems based on multi-objective machine learning. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(7): 1789–1801
114. Scogland T, Azose J, Rohr D, Rivoire S, Bates N, Hackenberg D. Node Variability in Large-Scale Power Measurements: perspectives from the Green500, Top500 and EEHPCWG. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2015, 1−11
115. Foster I, Zhao Y, Raicu I, Lu S. Cloud computing and grid computing 360-degree compared. In: Proceedings of the 2008 Grid Computing Environments Workshop. 2008, 1−10
116. Seneviratne S, Witharana S. A survey on methodologies for runtime prediction on grid environments. In: Proceedings of the 7th International Conference on Information and Automation for Sustainability. 2014, 1−6
117. Yang Q, Liu Y, Chen T, Tong Y. Federated machine learning: concept and applications. ACM Transactions on Intelligent Systems and Technology, 2019, 10(2): 12
118. Ben-Nun T, Hoefler T. Demystifying parallel and distributed deep learning: an in-depth concurrency analysis. ACM Computing Surveys, 2020, 52(4): 65
119. Li C, Sun H, Tang H, Luo Y. Adaptive resource allocation based on the billing granularity in edge-cloud architecture. Computer Communications, 2019, 145: 29–42
120. Orhean A I, Pop F, Raicu I. New scheduling approach using reinforcement learning for heterogeneous distributed systems. Journal of Parallel and Distributed Computing, 2018, 117: 292–302
121. Chen C L P, Liu Z. Broad learning system: an effective and efficient incremental learning system without the need for deep architecture. IEEE Transactions on Neural Networks and Learning Systems, 2018,

29(1): 10–24
122. Naghshnejad M, Singhal M. A hybrid scheduling platform: a runtime prediction reliability aware scheduling platform to improve HPC scheduling performance. The Journal of Supercomputing, 2020, 76(1): 122–149
123. Ye D, Chen D Z, Zhang G. Online scheduling of moldable parallel tasks. Journal of Scheduling, 2018, 21(6): 647–654
124. Dongarra J J, Simon H D. High performance computing in the US in 1995 - An analysis on the basis of the TOP500 list. Supercomputer, 1997, 13(1): 19–28
125. Feng W C, Cameron K W. The Green500 list: encouraging sustainable supercomputing. Computer, 2007, 40(12): 50–55
126. Wienke S, Iliev H, Mey D A, Muller M S. Modeling the productivity of HPC systems on a computing center scale. In: Proceedings of the 30th International Conference on High Performance Computing. 2015, 358−375
127. Dongarra J, Graybill R, Harrod W, Lucas R, Lusk E, Luszczek P, Mcmahon J, Snavely A, Vetter J, Yelick K, Alam S, Campbell R, Carrington L, Chen T Y, Khalili O, Meredith J, Tikir M. DARPA's HPCS program: history, models, tools, languages. Advances in Computers, 2008, 72: 1–100

Zhengxiong Hou received the PhD degree in computer science and technology from Northwestern Polytechnical University, China. He is an associate professor at the Center for High-Performance Computing, School of Computer Science, Northwestern Polytechnical University, China. His research interests include intelligent resource management and job scheduling, performance optimization in HPC clusters and clouds.

Hong Shen received the BEng degree from the Beijing University of Science and Technology, the MEng degree from the University of Science and Technology of China, the PhLic and PhD degrees from Abo Akademi University, Finland, all in computer science. He is currently a specially-appointed professor at Sun Yat-Sen University, China. He was a professor (Chair) of computer science in the University of Adelaide, Australia. With main research interests in parallel and distributed computing, algorithms, and high performance networks, he has published more than 300 papers including more than 100 papers in international journals such as a variety of IEEE and ACM transactions.

Xingshe Zhou received the BS and MS degrees in computer science from Northwestern Polytechnical University, China. He is a professor with the School of Computer Science, Northwestern Polytechnical University, China. He was the dean and director of the Center for High-Performance Computing of this university. His research interests include embedded computing and distributed computing. He has published more than 100 papers in international journals and conferences.

Jianhua Gu received the PhD degree in computer science and engineering from Northwestern Polytechnical University, China. He is a professor at the Center for High-Performance Computing, School of Computer Science, Northwestern Polytechnical University, China. His research interests include operating system and cloud computing.

Yunlan Wang received the PhD degree in computer science from Xi'an Jiaotong University, China. She is an associate professor at the Center for High-Performance Computing, School of Computer Science, Northwestern Polytechnical University, China. Her research interests include high-performance computing and data mining.

Tianhai Zhao received the PhD degree in computer science from Xi'an Jiaotong University, China. He is a lecturer at the Center for High-Performance Computing, School of Computer Science, Northwestern Polytechnical University, China. His research interests include parallel computing and cloud computing.