Towards Efficient Resource Allocation for Distributed Workflows Under Demand Uncertainties

Ryan D. Friese[1], Mahantesh Halappanavar[1], Arun V. Sathanur[1], Malachi Schram[1], Darren J. Kerbyson[1], and Luis de la Torre[2]

[1] Pacific Northwest National Laboratory
Richland, WA
{firstName.lastName}@pnnl.gov
[2] Universidad Metropolitana
San Juan, Puerto Rico
delatorrel1@suagm.edu

Abstract. Scheduling of complex scientific workflows on geographically distributed resources is a challenging problem. Selection and scheduling of a subset of available resources to meet a given demand in a cost efficient manner is the first step of this complex process. In this paper, we develop a method to compute cost-efficient selection and scheduling of resources under demand uncertainties. Building on the techniques of Sample Average Approximation and Genetic Algorithms, we demonstrate that our method can lead up to 24% improvement in costs when demand uncertainties are explicitly considered. We present the results from our preliminary work in the context of a high energy physics application, the Belle II experiments, and believe that the work will equally benefit other scientific workflows executed on distributed resources with demand uncertainties. The proposed method can also be extended to include uncertainties related to resource availability and network performance.

Keywords: Cost-efficient scheduling; uncertainty quantification; large scale workflows; sample average approximation

1 Introduction

Efficient utilization of computing resources is an important goal for the design and execution of complex scientific workflows. However, scheduling of these workflows on distributed computing resources is fraught with several uncertainties that lead to poor utilization of resources. In this work, we introduce the notion of uncertainties in forecasted demand and develop strategies for cost-efficient utilization of distributed resources. The three main components of our work are: (*i*) a prototypical scientific workflow from the Belle II experiments containing aspects of data generation through experiments and simulations, and analysis of this data; (*ii*) a methodology based on Sampled Average Approximation (SAA)

to generate scenarios and select efficient strategies; and (*iii*) a Genetic Algorithm (GA) based method to compute efficient strategies to meet forecasted demand given a set of resources and their usage costs.

The Belle II experiments probe the interactions of fundamental constituents of our universe. The experiments will generate about 25 peta bytes (a peta byte is 10^{15} bytes) of raw data per year with an anticipated stored data of over 350 peta bytes at the end of the experiment (2022) [1, 2]. Data is generated not only from the physical experiments conducted through the Belle II detector, but also from Monte Carlo simulations and user analysis. Similar to many large-scale experiments, the users, data, storage and computational resources related to the experiment are geographically distributed across the globe. Therefore the Belle II experiment is an ideal case study for our work.

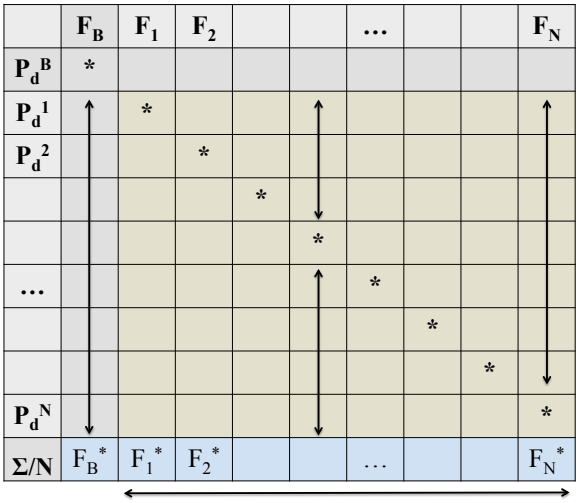

Fig. 1: An illustration of our approach using the Sample Average Approximation method. Different scenarios (represented along the rows: $P_d^1 - P_d^N$) are created by drawing random samples from the distribution of a given base demand (P_d^B). Solutions (represented along the columns: $F_B, F_1 - F_N$) are generated using a Genetic Algorithm based optimizer for each scenario. The asterisks on the diagonal correspond to the scenario used to create a solution. Each solution is used to solve every other scenario (non-diagonal entries). The bottom row represents the mean value for each solution over every scenario. The minimum mean value provides an optimal solution.

Intuitively, the Sample Average Approximation (SAA) provides a mechanism to optimize functions with variables that are subject to uncertainties. Two key ideas in SAA are the use of sampling and optimization under certainty [3]. Given a particular demand, in terms of the number of compute units, a cost-efficient mix of resources can be chosen to meet this demand, where the cost of using each type of resource is different. We provide a rigorous formulation of this optimization problem in Section 2. However, this deterministic optimization

problem becomes hard when the demand is subject to uncertainties. Note that the supply (availability of resources) is also subject to uncertainties that we will ignore in this work and note that our work can be extended to include such uncertainties. Assuming the probability distribution functions for demands are known apriori from the application domain, we first generate several scenarios by drawing random samples from these distributions. We then solve each scenario as a deterministic case using a Genetic Algorithm (GA) based approach. We implement a modified version of the popular Nondominated Sorted Genetic Algorithm II (NSGAII) [4]. We detail this method in Section 3.

The optimal strategy (the mix of resources to meet the demand) for a given scenario is used to compute the cost of meeting the demand from all the other scenarios. A mean cost for each strategy is computed, and the minimum of these means is chosen as the optimal strategy that is robust to different scenarios. A base case is also computed and compared against the optimal solution (in terms of solving different scenarios). The proposed SAA based method is illustrated in in Figure 1. We discuss the experimental setup and results in Sections 4 and 5, respectively.

Contributions

We make the following contributions in this preliminary work paper:

- Present a scheduling framework for Belle II workflows under demand uncertainties.
- Present a novel Genetic Algorithm based approach for computing cost-efficient selection of resources based on the analogy of unit commitment problem in electric power grids.
- Present a Sample Average Approximation based method to develop scenarios and compute solutions that are optimal across different scenarios. We demonstrate the effectiveness of this approach using a prototypical workflow from high energy physics application.
- We demonstrate a cost benefit of up to 24% relative to the optimal base case, and thus, make a case for the utility of considering demand uncertainties in scheduling of complex scientific workflows executed on distributed resources.

The paper is organized as follows. We provide a rigorous formulation of the cost-efficient selection of resources to meet a given demand in Section 2. We present the proposed methodology in Section 3 and explain the details of Sample Average Approximation and Genetic Algorithm based methods. We provide our experimental setup in Section 4 and experimental results in Section 5. We present related work in Section 6, and our conclusions in Section 7.

2 Problem Formulation

Given that a variety of distributed resources are available to meet a certain demand, the question we address is: *What is the most cost-effective allocation of resources to meet the given demand?* We address this question using the analogy

of Unit Commitment problem in the context of electric power grids [5]. We first introduced this idea in our previous work [6] using linear programming approaches to develop solutions. In this paper, we develop a genetic algorithm based approach and introduce the notion of *uncertain demand*. An important assumption we make is that only a subset of available resources is sufficient to meet the demand for a given time period. We note that this is a fair assumption when cloud computing resources are being utilized and when multiple dedicated resources are used to support several scientific applications.

For Belle II experiments, several types of distributed resources including dedicated and opportunistic on-demand resources are available. The cost structures (both fixed (start up) and operating) vary significantly for different kinds of resources. As an example, while dedicated resources have a fixed operating cost, cloud computing resources such as Amazon EC2 have a variety of resources with different fixed and usage based costs [7]. A similar problem of resource utilization also arises in the context of electric power grids that is popularly known as the Unit Commitment problem. Consider a power grid operator with access to several power generators with different start-up and operating costs, and demand (load) that varies significantly over a period of time including seasonal fluctuations. For a given period of time, generally from several hours to a few days, the objective of unit commitment is to determine a subset of power generators that will used and the amount of power they will generate to meet the demand at a *minimum cost*. Using the notation introduced by Wright [8], we can formally express the objective function as:

$$\min F = \sum_{t=1}^{T} \sum_{j=1}^{N} C_j(P_j(t)) + S_j(x_j(t), u_j(t)), \tag{1}$$

subject to constraints:

$$\sum_{j=1}^{N} P_j(t) = P_d(t) \tag{2}$$

$$\sum_{j=1}^{N} r_j(x_j(t), P_j(t)) \geq P_r(t), \tag{3}$$

where F is the total system cost for N power generators with operating (fuel) cost C_j and start-up cost S_j to generate P_j units of power. The variable x_j represents the number of time units (for example, hours) that a given generator is on (positive) or off (negative). Similarly, the variable u_j represents the state of a generator at a given time unit $t + 1$. It is positive (+1) if the state is up and negative (-1) if the state is down. The constraints enforce that the demand, $P_d(t)$ at time t, is satisfied. Further, the system is also required to meet a certain additional (reserve) unanticipated demand P_r, and r_j is the reserve available from generator j for time period t. The total time period under consideration is T.

For the purposes of this paper, we define $P_j(t)$ as the computing power of a resource v_j for time unit t. A metric for expressing power in the context of

Belle II is HEP SPEC – a metric derived from SPEC CPU 2006 standard [3]. Different costs for resources available for Belle II experiments can be potentially modeled using machine specifications (energy and power consumption) and operation policies, in a method described in Singer *et al.* [7]. Demand for computing resources arise from several tasks including Monte Carlo simulation campaigns with given number of events (P_d) that are simulated over a given period of time. The user analysis jobs are generally chaotic and lead to uncertainties in demand that need additional resources to satisfy. This situation is equivalent to the spinning reserve (P_r) in a power grid. A key problem addressed in this paper is that we consider demand uncertainties explicitly and solve the optimization problem. We will discuss our proposed methodology to solve the unit commitment based formulation using genetic algorithm in Section 3. An advantage of using the analogy of unit commitment is that a large body of work is available to solve the problem efficiently. We discussed a linear programming based approach in [6], and note that several other approaches are available in literature [5].

3 Proposed Methodology

We formulated the cost-efficient resource selection problem using the analogy of unit commitment problem in Section 2. In this section, we develop a method to handle demand uncertainties in this formulation using Sample Average Approximation (SAA), and then develop a genetic algorithm based method to compute efficient solutions for the optimization problem.

3.1 Sample Average Approximation (SAA)

The optimization problem described in Equation 1 cannot be computed exactly when demand is uncertain. Therefore, we employ the SAA technique, which is based on the ideas of sampling and deterministic optimization, to solve this problem. Formally, the problem can be expressed as:

$$\min f(x) = f(x, \xi), \tag{4}$$

where x represents the scheduling strategy for resources (which resources to commit and at what level to run) and ξ represents demand that is random and independent of x. $f(x, \xi)$ is therefore the total cost. Given a probability distribution function for demand, we randomly sample different values for demand $\xi_1, \xi_2, \ldots, \xi_n$, and set

$$\min f_n(x) = \frac{1}{n} \sum_{i=1}^{n} f(x, \xi_i). \tag{5}$$

For a given value of demand, the optimization problem becomes deterministic, which we solve using a genetic algorithm based approach (described next). We

[3] HEP SPEC is based on SPEC CPU 2006. Further information is available from http://w3.hepix.org/benchmarks/doku.php/

then pick the minimum value from Equation 4 ($\min f(x) = \min f_n(x)$), as illustrated in Figure 1. The SAA method converges to an optimal solution as the number of samples n increases [3]. We conduct our experiments with 5000 samples that we detail in Section 4. We note that in this particular formulation of the problem, our solution space is limited to the $(n+1)$ solutions corresponding to the n scenarios and the base case. The eventual solution (commitment strategy) is chosen to be the minimum mean solution among $(n+1)$ solutions. We plan to modify our current method to account for the objective function at each step as the expectation over $(n+1)$ scenarios to develop a globally optimal solution in our future work. We also plan to explore mathematical programming based methods to compute optimal solutions.

3.2 Genetic Algorithm

Genetic algorithms (GAs) are common evolutionary optimization techniques useful in solving problems that contain large and complex search spaces (e.g., [9–13]). GAs try to emulate the process of natural selection; i.e., producing better (fitter) solutions as time progresses. Typical GAs maintain a population of individuals called chromosomes. Each chromosome is a solution to the problem being solved. Chromosomes are compared with one another by evaluating their fitness. Fitness functions are often, but not always, the objective function to optimized (this paper presents two fitness functions, one is the direct objective (i.e. Equation 1), while the other is a relaxed version). Chromosomes are further composed of genes, the base component of a solution, their representation is highly dependent on the problem being solved. Our paper implements a popular multi-objective GA, the NSGAII [4].

Better solutions in a GA are produced as the population evolves through time. Evolution occurs due to three genetic-operators: selection, crossover, and mutation. During selection, chromosomes are chosen as parents to "mate" and produce offspring chromosomes. Typically, selection operators are biased towards selecting more fit chromosomes. The crossover operation takes the chromosomes chosen during selections and swaps a portion of the genes of each parent into one another, resulting in offspring chromosomes that contain genetic information from both parents. Finally, mutation operates on chromosomes individually, with individual genes in a chromosome being randomly mutated to introduce new genetic information. Selection, crossover, and mutation are applied to the population until some stopping criteria is met, e.g., the population converges, or a given number of iterations have been performed.

Numerous techniques can be used to speed up the process of finding fit chromosomes by taking advantage of parallel systems. One approach is an island model, which has numerous populations evolving simultaneously with occasional migration of chromosomes from population to population [14, 15]. Our GA implements a variation of the island model.

Chromosome Structure for Unit Commitment: As stated previously in Section 2, our goal is to select a combination of machines that meet a given demand and minimize cost over an N-month time period. Every month has an associated demand that must be met.

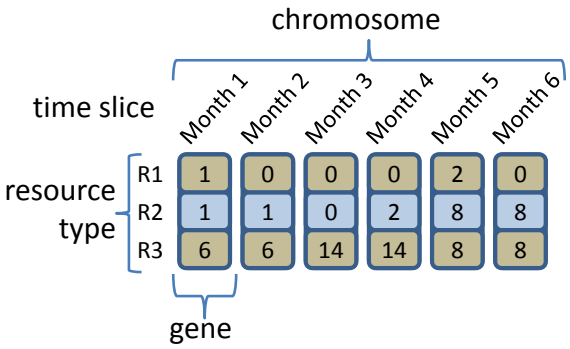

Fig. 2: Unit Commitment chromosome structure. Each gene represents the number of resources of each type used in a given month.

Due to the month-by-month decomposition of the problem, we can easily construct chromosomes where each gene will represent a specific month within the time period. Furthermore, for each month we indicate the number of resources of each type to be used. As a result, each chromosome can be represented as a $M \times R$ matrix, where M is the number of months, and R is the number of resource types available. For this paper, we assume machines can only be used in integer quantities, but accounting for "partial" machines is a trivial change. Figure 2 presents the chromosome and gene structure used in our GA. In this example we are purchasing resources for a 6 month period, and have 3 resource types to choose from.

In the case where the computing resources specified by a given chromosome do not meet the required demand (Equation 2), additional on-demand resources are "purchased" to make up the difference and produce a valid solution. Typically on-demand resources are more expensive per unit of compute performance, thus it is desirable to not have to purchase additional resources.

Genetic Operators: The selection operator used in our GA is a binary tournament selection [16] with replacement. Two randomly selected chromosomes are compared against one another and the fitter chromosome is selected as a parent for crossover.

Crossover is performed using a two-point crossover scheme [9]. In this scheme, the indices of two genes are randomly selected, and the genes between these two points are swapped between parent chromosomes to produce two new offspring chromosomes.

Finally, the mutation operator we implemented operates on indivdual genes. During this operation, each gene within a chromosome may mutate with a given probability. When a gene is selected for mutation, each resource type has a $\frac{1}{\#\text{ Resource Types}}$ probability of having the number of allocated resources changed. When a resource type mutates, a normal distribution with the mean equal to the current number of resources and a standard deviation of .5 is used to create a new value for the allocated resources of that type. We set negative allocations to zero.

Island Model Implementation: In island model GAs there exist P multiple populations that evolve concurrently. After a given number of iterations, the

populations will migrate chromosomes from one population to another, introducing new genetic material into each population. The idea is that the individual populations will explore and optimize different areas of the search space. During migration events, new genetic material is introduced that has been optimized for a different sub-spaces. By combining optimized genetic material for two different sub-spaces, the hope is that new solutions will be created that span the sub-spaces and result in solutions that are more fit overall.

The rate of migration can significantly impact the performance of the GA. When a migration event occurs after every iteration, the individual populations essentially form a single larger population. This is because the constant exchange of genetic material from neighboring populations prevents diversity and exploration within the individual populations. When migration events never occur, no genetic material is exchanged between the populations, and the result is the same as if P regular populations were executed and the best solution was chosen from among them. Selecting an appropriate migration rate is generally related to the convergence rates of the individual populations, and is thus highly dependent on the problem being solved. We used an empirically determined static migration rate.

We implement a modified island model GA, where we have a central population, and several satellite populations. The satellite populations migrate chromosomes with one another using a ring pattern and also perform an one-way many-to-one communication with the central population (Figure 3). The satellite (blue) populations both send and receive chromosomes (blue arrows) while the central (orange) population only receives chromosomes (orange arrows). During a migration event, each population (excluding the central population) will send copies of its *most fit* chromosome. When a new chromosome is received by a population, it will replace its *least fit* chromosome with the new one.

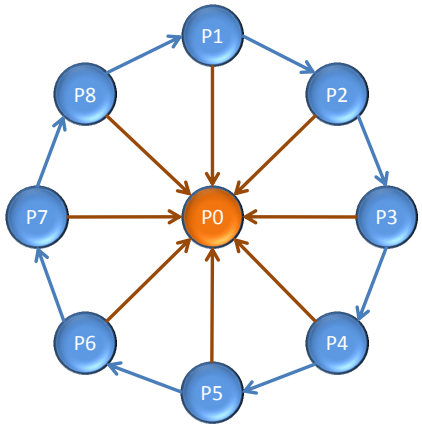

Fig. 3: Illustration of GA Island Model. The ring communication pattern is represented as the blue arrows connecting the satellite populations. The many-to-one communication between the central population and satellite populations is represented as the orange arrows.

The number of chromosomes are the same for every population and should be at least equal to the number of populations themselves (this guarantees the central population has enough space to hold a chromosome from every satellite population). By using same sized populations, the execution time to perform n iterations between migration events will be very similar for each population. Thus, we have implemented the communication between populations using synchronized MPI send/recvs. For our problem, idle time due to load unbalance (from some populations finishing before others) and communication overhead is negligible compared to the compute cost between migration events.

Objective Functions (minimize total and base cost): We optimized for two objectives simultaneously. The first objective is the total cost to purchase the machines required to meet a given demand, while the second cost is what we call the "base" cost. The total cost includes both the cost due to purchasing the machines specified by a chromosome, and any cost incurred when extra on-demand machines are required to be purchased. The base cost does not include the cost for additional machines. Recall, chromosomes may potentially create solutions that do not meet the required demand, thus extra machines need to be purchased to make these chromosomes valid solutions. Minimizing total cost is the exact objective we are trying to minimize (Equations 1 and 2), while minimizing base cost is a relaxation.

We use the base cost as the second objective instead of only optimizing for total cost for multiple reasons. First, minimizing the base cost (in addition to total cost) allows the GA to more directly explore areas of the search space that contain (potentially high performing) invalid solutions (i.e. compute resources that do not meet required demand). Typically, the base cost should have a higher contribution towards the total cost (for good solutions), by explicitly optimizing the base cost we more directly optimize the total cost as well. Finally, any solution that requires extra resources to be purchased can actually be represented by a number of valid solutions, thus minimizing the base cost covers larger areas of the search space than minimizing the total cost alone.

Algorithm 1 shows the method for calculating the costs (base and extra) of a given chromosome, To calculate the base cost of a chromosome we iterate over each month, summing the on-demand cost of any used resource (applies to on-demand and hybrid cost resources) as well as the start-up cost for any newly purchased subscription based resources. To accurately account for hybrid resources that incur an on-demand cost when used, we keep track of previously purchased resources for the length of their contract (line 8). Note, for purely on-demand machines, their contract length is one month.

Calculating extra cost also iterates over each month, but the sum of compute power supplied by each purchased resource is compared to the demand required for that month. If the supplied compute power is greater than or equal to the demand, no additional resources are purchased. Otherwise the supplied compute power does not meet the required demand, and additional extra (on-demand) resources must be purchased. Typically, this extra resource has a higher cost than other resources. The total cost of a chromosome would simply be the sum of the base and extra costs.

Algorithm 1 Calculating the costs (base and extra) of a chromosome

Input: machinesToPurchase[months][numResourceTypes], demand[months], extraResource
Output: cost,extraCost

$baseCost = 0$
$extraCost = 0$
$purchased[months][numResourceTypes]$
for m in $months$ **do**
  $availCompute = 0$
  for r in $resourceTypes$ **do**
    for $t \leftarrow m..(m + r.contractLength - 1)$ **do**
      $purchased[t][r.type] += machinesToPurchase[m][r.type]$
    $baseCost += purchased[m][r.type] * r.onDemandCost$ ▷ 0 for up-front resources
    $baseCost += machinesToPurchase[m][r.type] * r.upFrontCost$ ▷ 0 for on-demand resources
    $availCompute += purchased[m][r.type] * r.compute$
  $neededCompute = demand[m] - availCompute$
  if $neededCompute > 0$ **then**
    $extraCost += ceil(neededCompute/extraResource.compute) * extraResource.onDemandCost$
return $baseCost, extraCost$

4 Experimental Setup

Computation in Belle II experiments arise from three kinds of activities: (*i*) processing of raw data from the Belle II detector, (*ii*) Monte Carlo simulations of physical phenomena, and (*iii*) physics analysis of experimental and simulation data. Both data storage and computation span a geographically distributed set of resources covering several continents. While the computational demand for Monte Carlo campaigns is fairly stable, the demand for user analysis tends to be chaotic leading to uncertainties in demand. Inspired from this setting, we use a representative setup for demand and supply in our experiments that are detailed in this section.

We present two sets of simulations, the first is a small illustrative example while the second is a larger simulation inspired from the Belle II experiment. For the large simulations, we use models of cloud computing resources based on Amazon EC2, as shown in Table 1 (note that the prices in the table may not reflect current Amazon EC2 prices). The resource costs for the illustrative example are presented in Table 2. Each compute resource has an associated ECU (compute power), contract length, and up-front (S_j) and on-demand costs (C_j).

Multiple demand curves are constructed as follows. For our illustrative example, we simulate a 6-month workflow using 10 sampled scenarios. The base demand curve for this example is the black line in Figure 4. To create a new scenario, for each month, we used the demand d_m from the base curve as the seed value and sample a new value from a uniform distribution $U(d_m - 7.5, d_m + 10)$. In Figure 4, the sampled demand curves are the blue lines while the area of

Machine	**Cost type**	**ECU**	**Time (month)**	**S_j** ($)	**C_j** ($/month)
1	on-demand	0.2	1	0	13.14
2	hybrid	0.2	12	102	4.38
3	subscription	0.2	12	151	0
4	hybrid	0.2	36	218	2.92
5	subscription	0.2	36	303	0
6	on-demand	0.8	1	0	26.28
7	hybrid	0.8	12	204	8.76
8	subscription	0.8	12	302	0
9	hybrid	0.8	36	436	5.84
10	subscription	0.8	36	607	0
11	on-demand	6.5	1	0	63.51
12	hybrid	6.5	12	324	27.01
13	subscription	6.5	12	635	0
14	hybrid	6.5	36	657	18.25
15	subscription	6.5	36	1235	0
16	on-demand	13	1	0	126.29
17	hybrid	13	12	648	54.02
18	subscription	13	12	1271	0
19	hybrid	13	36	1314	36.5
20	subscription	13	36	2470	0
21	on-demand	26	1	0	252.58
22	hybrid	26	12	1296	108.04
23	subscription	26	12	2541	0
24	hybrid	26	36	2628	73
25	subscription	26	36	4941	0
26	on-demand	53.5	1	0	505.89
27	hybrid	53.5	12	2593	216.08
28	subscription	53.5	12	5082	0
29	hybrid	53.5	36	5256	146
30	subscription	53.5	36	9881	0
31	on-demand	124.5	1	0	1264.36
32	hybrid	124.5	12	6482	540.2
33	subscription	124.5	12	12706	0
34	hybrid	124.5	36	13150	365
35	subscription	124.5	36	24703	0

Table 1: Representative subscription costs for Amazon EC2 resources. The third column EC2 Compute Unit (ECU) provides the relative measure of the processing power of an Amazon EC2 machine instance. The fourth column represents the subscription time for a specific machine subscription plan. The fifth column S_j represents the fixed (set up) cost (in dollars) for the period specified in the fourth column. The sixth column represents the monthly usage cost C_j (in dollars per month).

Machine	Cost typ	ECU	Time (month)	S_j ($)	C_j ($/month)
1	on-demand	1.3	1	0	10.0
2	hybrid	1.3	4	11.5	2.9
3	subscription	1.3	4	22.8	0

Table 2: Resource costs for illustrative example.

possible demands is shown as the shaded area. For the larger experiments, we simulated a 24-month workflow and executed two 5000-scenario campaigns. The base demand curve, upon which all the other scenarios drew their demand curves from, is presented as the solid line in Figure 7. For the first campaign, the range for the uniform distribution is $a = d_m - 15, b = dm + 20$. The range for the second campaign is half that of the first, i.e., $a = d_m - 7.5, b = d_m + 10$. Sample demand curves and possible demand values for the higher (blue line and shaded area) and lower variation (green line and shaded area) campaigns are presented in Figure 7. In future work, we plan to examine base curves for additional workflows, and different sampling distributions.

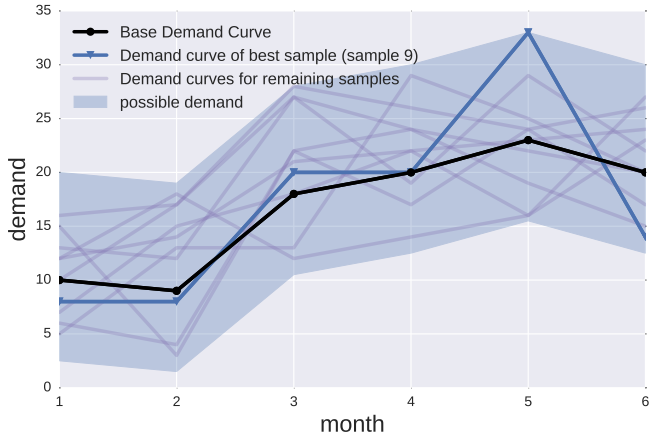

Fig. 4: Demand curves for the illustrative example. The base demand curve is shown as the black line. The demand curve for best sample is the dark blue line, while the light blue lines show the demand curves for all other samples. The shaded area represents the possible demand values $(d_m - 7.5, d_m + 10)$.

All simulations were run on a 20-node cluster with an InfiniBand interconnect. Each node is equipped with dual 10-core Intel IvyBridge CPUs (Intel E5-2680 v2, 2.8 GHz) and 128 GB RAM (DDR3-1866). Each scenario fully occupied one node (one population per core), and multiple scenarios were executed across the cluster concurrently.

The following parameters were used in the genetic algorithm to produce solutions for each scenario. The number of populations in the island model was set to 20 (19 satellite populations and a single central population). Each population contained 25 chromosomes. Each chromosome starts with a random initial state. Migration events were performed every 200 generations. The probability that an individual gene would be selected for mutation was set to 1%. If two chromosomes were selected for crossover, they would always mate and produce two offspring. Finally, we terminated the algorithm and took the best solution found after 60 seconds. Given these parameters and our cluster configuration, each population performed roughly 30,000 iterations, resulting in 300 migration events. The compute-to-communication ratio is between 10-12.

5 Experimental Results and Discussion

We now present the experimental results for the two simulations, a smaller illustrative example and larger simulations, as detailed in Section 4. We will start with a detailed presentation of the illustrative example and then present the key details of the larger simulations. Our goal in this section is to highlight the substantial gains that can be obtained by carefully considering demand uncertainties relative to a solution obtained without uncertainties (base case).

As an illustration, we present the cost of computing all the scenarios (represented as rows) with the most optimal strategy (represented as columns) for a given scenario (diagonal entries) in Table 5 for the small illustrative example. Note that the diagonal entry will be the best solution for a given scenario (P_d^i) among solutions computed by different strategies (F_i). The final strategy is selected by picking a strategy that provides the minimum mean value across different scenarios, represented in the bottom-most row of the matrix. Further details of the optimal strategy (F_9) are provided in Figure 6. We show the difference between the mean values of base case (F_B) and the optimal strategy as the expected benefit of the proposed method.

We consider two variants of the larger simulation by varying the probability distribution functions for demand relative to the base case. While one variation is small, the other is relatively large. We capture the key results in Figure 7. First, we show the base demand curve (black line) and the regions of possible demand for the high (blue shaded area) and low (green shaded area) variations. The demand curves for the best performing solutions for the high (blue line) and low (green line) variants show that both solutions tend to have higher demands (per month) than the base curve. In general, the monthly demand is met by a higher contribution of subscription and hybrid machines, using few (costly) on-demand machines. It is cheaper to buy a subscription or hybrid resource and let it sit idle for a couple of months, than to replace that resource with an equivalent on-demand resource. Thus, the best performing solutions are those that are over-provisioned when applied to other demand scenarios.

In order to highlight the main benefits of incorporating uncertainties, we capture the benefit (difference between the mean costs of base and optimal strategies) in Figures 8 and 9. We present the difference as a percentage relative to

	F_B	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
P_d^B	510.7	572.1	588.5	618.4	620.1	558.9	535.7	574.3	550.2	529.4	538.7
P_d^1	640.7	597.9	661.7	705.8	653.3	674.7	644.4	614.3	651.8	639.4	647.4
P_d^2	707.8	697.9	636.5	780.0	689.1	734.7	718.6	680.1	726.0	709.4	707.4
P_d^3	616.2	633.4	644.3	575.8	679.1	622.8	605.7	642.7	642.8	646.5	631.3
P_d^4	800.7	767.9	764.9	865.8	689.4	806.3	814.4	730.1	797.6	799.4	807.4
P_d^5	617.8	652.1	661.4	668.4	663.0	527.9	618.6	634.3	606.0	609.4	615.8
P_d^6	549.1	583.4	605.6	614.2	631.7	590.2	524.4	592.7	567.3	566.5	577.1
P_d^7	650.7	629.2	664.6	722.9	637.5	677.6	654.4	570.1	647.6	649.4	657.4
P_d^8	573.3	606.0	638.2	622.6	620.1	611.5	558.6	559.8	497.6	543.6	542.9
P_d^9	586.2	651.8	627.2	672.6	690.1	620.2	575.7	639.8	586.0	549.4	585.8
P_d^{10}	523.3	576.3	564.0	601.0	605.6	544.4	527.0	578.5	483.1	496.4	457.4
Σ/10	**614.0**	**633.5**	**641.5**	**677.1**	**652.6**	**633.6**	**616.1**	**619.7**	**614.2**	**612.6**	**615.3**

Fig. 5: Complete set of results for the illustrative example. Different scenarios are represented as rows, the diagonal entries represent the best strategy (solution) for the corresponding scenario. A column indicates the cost for each scenario for a given strategy. The bottom-most row represents the mean cost for a given strategy. The final strategy picked is the one that is minimum (F_9) in this row.

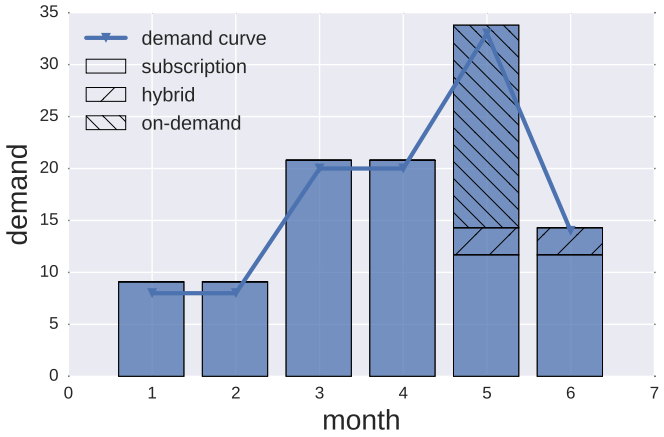

Fig. 6: Resource allocation (and demand curve) of best solution for illustrative example. The different hatches within a bar represent the amount of each resource cost type (subscription, hybrid, on-demand) used.

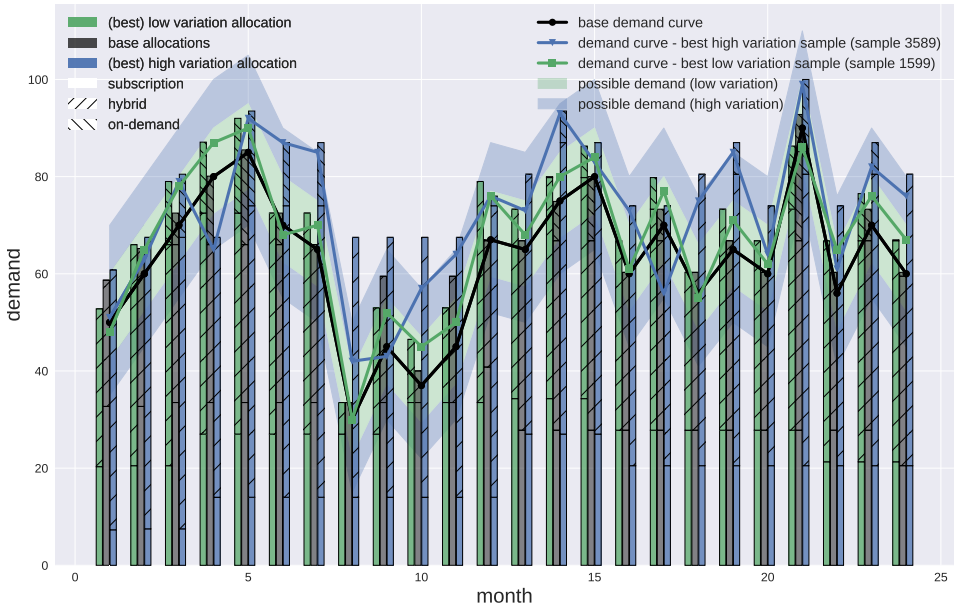

Fig. 7: Demand curves and allocations for Belle II based simulation. The base demand curve is shown as the black line. The demand curve for high variation best sample is the blue line, while the green line shows the demand curve for low variation best sample. The blue shaded area represents the possible demand values for the high variation experiment$(d_m - 15, d_m + 20)$. The green shaded area represents the possible demand values for the low variation experiment $(d_m - 7.5, d_m + 10)$. The vertical bars show resource allocations for the base (black), the best high variation (blue) and best low variation (green). The different hatches within a bar represent the amount of each resource cost type (subscription, hybrid, on-demand) used.

the base case in Figure 8. The difference is presented as a function of the number of samples and variation in demand. We present the benefits as absolute numbers (along Y-axis) in Figure 9 and provide the identity of the optimal strategy for a given number of samples plotted along X-axis.

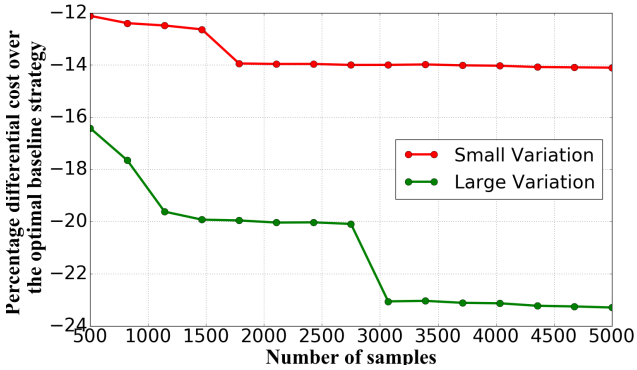

Fig. 8: Percentage differential costs between optimal base case and optimal strategy. Different number of samples are used along X-axis for two different scenarios – small variation and large variation.

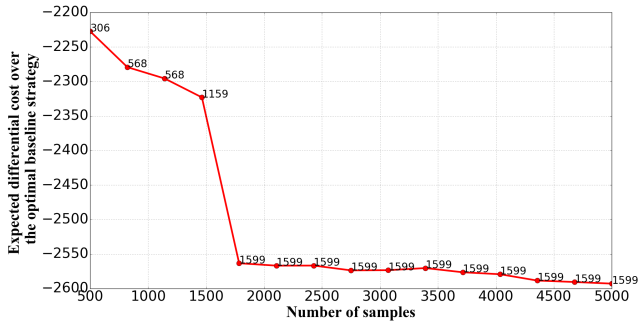

Fig. 9: Expected differential costs between optimal base case and optimal strategy for the small variation scenario with different number of samples. The numbers indicate the identity of the strategy that was chosen as an optimal strategy.

We observe that the accuracy of the proposed method increases as the number of samples increase (plotted along X-axis), and the strategies change providing different (nonlinear) amounts of benefit relative to the base case. Further, we note that for the small variation simulation, the optimal strategy remains

constant (Strategy 1599) after a certain number of samples (about 1700). The difference between the base case and the optimal strategy also increases when the amount of variation increases. We note that we are able to clearly demonstrate the benefits of incorporating demand uncertainties by using a rather simplistic setting with uniform distributions. Since real-world scenarios vary significantly and have different kinds of probability distributions, we anticipate that the proposed method will result in significant cost benefits for such situations.

6 Related Work

Our work is inspired from the unit commitment problem in power grids in developing a strategy for cost-efficient resource allocation for complex scientific workflows on distributed computing resources [6]. In this work, we introduced the notion of demand uncertainties that arise due to several different reasons including uncertainties in estimations and chaotic use of resources. Given the roots of our work in unit commitment problem, a natural area of related work lies in a large body of work on modeling and quantification of uncertainties for the unit commitment problem [17–19]. We note that the underlying physics of a power grid makes modeling uncertainties in that area a relatively challenging problem. However, we can derive from the existing work to model uncertainties in scheduling and resource allocation. Furthermore, the use of genetic algorithms to solve the unit commitment problem has been presented in [20, 21].

The notion of demand uncertainty has also been explored in the area of grid computing by Batista *et al.* [22], and in resource allocation by Johansson and Sternad [23]. Unlike the previous work, we develop cost-efficient strategies for resource selection for distributed workflows that have access to a large set of resources with diverse cost structures for usage. Our work becomes especially relevant in the context of cloud computing. The notion of uncertain demand has also been explored for cloud based resources. Johannes *et al.* explored the use of fuzzy optimization for resource allocation with uncertain demand [24]. Related work on cloud resource allocation work has been explored in [25–27]. Genetic algorithms have been used extensively for resource allocation [28–31].

To the best of our knowledge, we believe this work is the first in exploring cost-efficient resource allocation as the first step in an integrated approach to schedule and manage HEP workflows, such as Belle II, under demand uncertainties.

7 Conclusions

We presented our preliminary work on incorporating demand uncertainties in computing cost-efficient resource selection as the first step of an integrated approach for efficient scheduling of complex workflows on distributed computing platforms. Using the technique of Sample Average Approximation, we demonstrate upto 24% improvement in costs relative to an optimal base case without uncertainties. We also presented a Genetic Algorithm based technique to compute efficient solutions for optimization problems when demands are certain. We

developed our experiments within the context of a high energy physics workflow, the Belle II experiments, that execute on geographically distributed resources.

In our future work, we plan to extend our work through rigorous modeling of uncertainties and develop computationally efficient techniques based on stochastic programming. We believe that the benefits of our work will not only prepare us for the forthcoming Belle II experiments but also lead to significant reduction in costs in the utilization of available resources with varying cost structures. We also believe that our work will benefit a large number of complex workflows that utilize distributed computing resources in general and cloud computing resources in particular.

Acknowledgment

This work was supported by the Integrated End-to-end Performance Prediction and Diagnosis for Extreme Scientific Workflows (IPPD) Project. IPPD is funded by the U. S. Department of Energy Awards FWP-66406 and DE-SC0012630 at the Pacific Northwest National Laboratory. The work of Luis de la Torre was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Visiting Faculty Program (VFP).

References

1. D. M. Asner, E. Dart, and T. Hara, "Belle ii experiment network and computing," *arXiv preprint arXiv:1308.0672*, 2013. 2
2. T. Hara, "Belle II: Computing and network requirements," in *Proc. of the Asia-Pacific Advanced Network*, 2014, pp. 115–122. 2
3. S. Kim, R. Pasupathy, and S. G. Henderson, *A Guide to Sample Average Approximation*. New York, NY: Springer New York, 2015, pp. 207–243. 2, 6
4. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002. 3, 6
5. B. Saravanan, S. Das, S. Sikri, and D. Kothari, "A solution to the unit commitment problem – a review," *Frontiers in Energy*, vol. 7, no. 2, pp. 223–236, 2013. 4, 5
6. M. Halappanavar, M. Schram, L. de la Torre, K. Barker, N. R. Tallent, and D. J. Kerbyson, "Towards efficient scheduling of data intensive high energy physics workflows," in *Proceedings of the 10th Workshop on Workflows in Support of Large-Scale Science*, ser. WORKS '15. New York, NY, USA: ACM, 2015, pp. 3:1–3:9. 4, 5, 17
7. G. Singer, I. Livenson, M. Dumas, S. N. Srirama, and U. Norbisrath, "Towards a model for cloud computing cost estimation with reserved instances," *Proc. of 2nd Int. ICST Conf. on Cloud Computing, CloudComp 2010*, 2010. 4, 5
8. B. Wright, "A review of unit commitment," 2013. 4
9. J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, First Edition*. The University of Michigan, 1975. 6, 7
10. Z. Michalewicz and D. B. Fogel, *How to Solve It: Modern Heuristics*, 2nd ed. Springer-Verlag Berlin Heidelberg, 2004. 6

11. H. Singh and A. Youssef, "Mapping and scheduling heterogeneous task graphs using genetic algorithms," in *5th Heterogeneous Computing Workshop (HCW '96)*, Apr. 1996, pp. 86–97. 6
12. L. Wang, H. J. Siegel, V. P. Roychowdhury, and A. A. Maciejewski, "Task matching and scheduling in heterogeneous computing environments using a genetic-algorithm-based approach," *Journal of Parallel and Distributed Computing, Special Issue on Parallel Evolutionary Computing*, vol. 47, no. 1, pp. 8–22, nov 1997. 6
13. D. Whitley, "The genitor algorithm and selective pressure: Why rank based allocation of reproductive trials is best," in *3rd International Conference on Genetic Algorithms*, June 1989, pp. 116–121. 6
14. R. Tanese, "Distributed genetic algorithms," in *3rd International Conference on Genetic Algorithms*, 1989, pp. 434–439. 6
15. M. Gorges-Schleuter, "Explicit parallelism of genetic algorithms through population structures," in *1st Workshop on Parallel Problem Solving from Nature (PPSN)*, 1990, pp. 150–159. 6
16. B. Miller and D. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Systems*, vol. 9, pp. 193–212, 1995. 7
17. P. A. Ruiz, C. R. Philbrick, E. Zak, K. W. Cheung, and P. W. Sauer, "Uncertainty management in the unit commitment problem," *IEEE Transactions on Power Systems*, vol. 24, no. 2, pp. 642–651, May 2009. 17
18. A. Gholami, T. Shekari, F. Aminifar, and M. Shahidehpour, "Microgrid scheduling with uncertainty: The quest for resilience," *IEEE Transactions on Smart Grid*, vol. 7, no. 6, pp. 2849–2858, Nov 2016. 17
19. Q. Wang, J. Wang, and Y. Guan, "Stochastic unit commitment with uncertain demand response," *IEEE Transactions on Power Systems*, vol. 28, no. 1, pp. 562–563, Feb 2013. 17
20. K. S. Swarup and S. Yamashiro, "Unit commitment solution methodology using genetic algorithm," *IEEE Transactions on Power Systems*, vol. 17, no. 1, pp. 87–91, Feb. 2002. 17
21. S. A. Kazarlis, A. Bakirtzis, and V. Petridis, "A genetic algorithm solution to the unit commitment problem," *IEEE Transactions of Power Systems*, vol. 11, no. 1, pp. 42–51, Feb. 1996. 17
22. D. M. Batista, A. C. Drummond, and N. L. S. da Fonseca, "Scheduling grid tasks under uncertain demands," in *Proceedings of the 2008 ACM Symposium on Applied Computing*, ser. SAC '08. New York, NY, USA: ACM, 2008, pp. 2041–2045. 17
23. M. Johansson and M. Sternad, "Resource allocation under uncertainty using the maximum entropy principle," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4103–4117, Dec 2005. 17
24. A. Johannes, N. Borhan, C. Liu, R. Ranjan, and J. Chen, "A user demand uncertainty based approach for cloud resource management," in *2013 IEEE 16th International Conference on Computational Science and Engineering*, Dec 2013, pp. 566–571. 17
25. E. Medernach and E. Sanlaville, "Fair resource allocation for different scenarios of demands," *European Journal of Operational Research*, vol. 218, no. 2, pp. 339 – 350, 2012. 17
26. Q. Zhang, E. Gürses, R. Boutaba, and J. Xiao, "Dynamic resource allocation for spot markets in clouds," in *Proceedings of the 11th USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, ser. Hot-ICE'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 1–1. 17
27. Z. Li and M. Ierapetritou, "Process scheduling under uncertainty: Review and challenges," *Computers and Chemical Engineering*, vol. 32, no. 45, pp. 715 – 727, 2008. 17

28. M. Oxley, S. Pasricha, H. J. Siegel, A. A. Maciejewski, J. Apodaca, D. Young, L. Briceno, J. Smite, S. Bahirat, B. Khemka, A. Ramirez, and Y. Zou, "Makespan and energy robust stochastic static resource allocation of a bag-of-tasks to a heterogeneous computing system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2791–2805, Oct 2015. 17
29. M. S. Garshasbi and M. Effatparvar, "High performance scheduling in parallel heterogeneous multiprocessor systems using evolutionary algorithms," *International Journal of Intelligent Systems and Applications*, vol. 11, pp. 89–95, Oct 2013. 17
30. A. Dogan and F. Ozguner, "Genetic algorithm based scheduling of meta-tasks with stochastic execution times in heterogeneous computing systems," *Cluster Computing*, vol. 7, no. 2, pp. 177–190, Apr 2004. 17
31. R. Friese, "Efficient genetic algorithm encoding for large-scale multi-objective resource allocation," in *9th Workshop on Large-Scale Parallel Processing (LSPP'16), in the proceedings of the IPDPS 2016 Workshops & PhD Forum (IPDPSW)*, May 2016. 17