Discovering Piecewise Linear Models of Grid Workload

Tamas Elteto, Cecile Germain-Renaud, Pascal Bondon, Michèle Sebag

Discovering Piecewise Linear Models of Grid Workload

Tamás Éltető *, Cécile Germain-Renaud *, Pascal Bondon †, Michèle Sebag *
* Laboratoire de Recherche en Informatique Université Paris-Sud 11, CNRS and INRIA-Saclay
{eltetot,cecile.germain,michele.sebag}@lri.fr
†Laboratoire des Signaux et Systèmes, Supelec, CNRS
Pascal.Bondon@lss.supelec.fr

***Abstract*—Despite extensive research focused on enabling QoS for grid users through economic and intelligent resource provisioning, no consensus has emerged on the most promising strategies. On top of intrinsically challenging problems, the complexity and size of data has so far drastically limited the number of comparative experiments. An alternative to experimenting on real, large, and complex data, is to look for well-founded and parsimonious representations. This study is based on exhaustive information about the gLite-monitored jobs from the EGEE grid, representative of a significant fraction of e-science computing activity in Europe. Our main contributions are twofold. First we found that workload models for this grid can consistently be discovered from the real data, and that limiting the range of models to piecewise linear time series models is sufficiently powerful. Second, we present a bootstrapping strategy for building more robust models from the limited samples at hand.**

I. Introduction

Large-scale distributed computing systems, such as EGEE (Enabling Grid for E-sciencE) [11], allocate computing resources following the matchmaking principle pioneered by Livny [32]: the providers publish the characteristics of their resources, and these are matched with the users' requests. The fundamental motivation for the matchmaking principle is the federative nature of real-world grids. On the other hand, grid users, or grid market participants, seek for differentiated Quality of Service (QoS): in the e-science context, physicists ask for a different service for interactive analysis tasks and for long running simulations; TeraGrid users exploit the Batch Queue Predictor [3] capacities. An extensive body of research e.g. to cite only a few [21], [24], [27], [31] focuses on economic and intelligent models of resource provisioning for QoS, which sophisticate, but do not contradict, the matchmaking principle. Despite this intense activity, no consensus has emerged on the most promising strategies. For instance, the EGEE production grid adopts an agnostic approach derived from the Copernican principle [13] ("job is not special"); even research grids are quite conservative when production is concerned.

Scheduling for large-scale distributed systems explores a very complicated landscape. Any job dispatcher has to integrate a feedback loop with the resource provider; the usage involve *externalities*, decisions which affects users and resources beyond the resource consumer and producer; QoS should not result in under-utilization, thus even the more constrained models should state scheduling as a multi-objective optimization problem. On top of these intrinsic difficulties, two operational issues contribute to challenge the researcher. First real world experimentation is hardly possible. Second, significant experiments with simulators or analysis require large data sets. These data sets may be publicly available, but comparative experiments are rare in the grid community and experiments on high level concepts such as autonomic programming models [15], [22], are extremely difficult to conduct. One of the reasons is probably to be found in the well-known data-mining ratio: 80% of the effort goes to pre-processing.

An alternative to experimenting on real, large,

and complex data is to look for well-founded and parsimonious representations, with the unavoidable approximations implied. The goal of this paper is thus to explore explanatory and generative models rather than predictive ones. We answer a set of preliminary questions, which may help steering the design of those along feasible paths: is it possible to exhibit consistent models of the grid workload? If such models do exist, which classes of models are more appropriate considering both simplicity and descriptive power? How can we actually discover such models? And finally, how can we rigorously assess the quality of these models?

Our main contributions are twofold. First, we found that grid workload models can consistently be discovered from the real data, and that limiting the range of models to piecewise linear time series models is sufficiently powerful. Second, we present a bootstrapping strategy for building robust models from the limited samples at hand. This study is based on exhaustive information covering more than a year of the flagship EU grid infrastructure EGEE, and is representative of a significant fraction of e-science computing activity in Europe.

The rest of the paper is organized as follows. Section II describes the data set, its grid context, and the derivation of the times-series workload process from the empirical data. Section III defines the piecewise AR model and describes a model selection procedure. Section IV presents the validation methodology. Section V discusses the experimental results and presents the bootstrapping strategy. Section VI discusses related work, before the conclusion.

II. The Workload Process

A. EGEE and gLite

For the sake of precision and because the experimental data set come from EGEE, this section will describe its scheduling under gLite [10], its major middleware. gLite integrates the sites' computing resources through a set of middleware-level services, the Workload Management System (WMS). the WMS accepts jobs from users and dispatches them to computational resources based on the users requirements on one hand, and the characteristics (*e.g.* hardware, software, localization) and state of the resources on the other hand. The Copernican principle applies to the derivation of the Expected Response Time published by the sites' queues, named Computing Elements (CEs) in the operational version of the Grid Information Model (we skip here the fundamental issues about the semantics of a CE analyzed in [12]). As other high performance space-shared systems, most EGEE sites implement their scheduling policies through multiple FIFO queues and complex tuning of configuration files.

B. Workload Definition

In grid context, workload is the equivalent of *backlog* in queuing systems terminology. Backlog at time t has two definitions a) the amount of unfinished work in the system and b) the delay that a job arriving at time t would experience before starting execution. Our interpretation is the first one. Formally, let $T_a(j)$ be the arrival date of job j at a CE, $T_s(j)$ the date where job j starts running, and $T_e(j)$ the date where job j finishes. The cumulative running time of jobs that are accepted by the CE up to time t is

$$C^{\mathrm{RA}}(t) = \sum_{j:T_a(j)<t} T_e(j) - T_s(j).$$

The cumulative running time of jobs that are started by the system up to time t is

$$C^{\mathrm{RS}}(t) = \sum_{j:T_s(j)<t} T_{\mathrm{e}}(j) - T_{\mathrm{s}}(j).$$

The remaining running time of jobs that are started by the system and not yet finished is

$$R^{\mathrm{R}}(t) = \sum_{j:(T_s(j)<t)\wedge(T_e(j)>t)} T_e(j) - t.$$

The workload at time t is the total running time of jobs that were accepted by CE and waiting to start plus the remaining running time of jobs already running and not finished yet.

$$W(t) = C^{\mathrm{RA}}(t) - C^{\mathrm{RS}}(t) + R^{\mathrm{R}}(t).$$

This definition implicitly assumes a homogeneous intra-CE system, by not referencing the dispatch algorithm. In fact, the actual running time

of jobs, as observed in the logs, depends on the capacities of the machine on which they ran, thus on the dispatch system, except if the machine panel is fully homogeneous. The homogeneity assumption would be grossly erroneous at the grid scale; as we will consider the time series individually for each CE, it is acceptable: the grid sites are institutional ones, with reasonable coherency.

C. The data set

This study is based on exhaustive information covering all the gLite monitored jobs in the EGEE grid, from August 2008 to March 2009. The data is collected by the Real Time Monitor project, and is available through the Grid Observatory portal [1], [20].

Significant preprocessing was required for building the workload process. First, jobs that fail to be dispatched are reported with a zero timestamp, and were excluded. Second, and more importantly, as in any real-world large-scale experiment, measurements may in exceptional cases not be accurate. For instance, [36] reports situation where the LogMonitor service become clogged, and is not consistent with the time-stamps provided by the Local Resource Management System (LRMS) service. However, as LRMS information for the entrance in the queue is not available, we choose to use the uniform reporting system provided by LogMonitor. Therefore, an outlier detection procedure had to be applied in order to remove artifacts. Attempts to fit the distributions with classical ones failed, thus there was no theoretical basis for outlier detection. Common knowledge in the EGEE community is that execution times longer than one day should be considered suspicious. Comparison of the LRMS data and LogMonitor data confirmed this intuition, leading to an exclusion threshold of one day. The fractions of excluded outliers were close to 10%.

Descriptive Statistics: Table I presents the statistics of the nine CEs featuring the largest total load (the real names of the CEs are omitted for privacy reason). The percentile refer to the workload. Due to lack of space, we cannot detail the statistics, but all criteria for very high variability (variance, interquartile range, maximum) are met.

TABLE I
DESCRIPTIVE STATISTICS FOR THE TOP CES

	Total [years]	Jobs	percentile [days]		
			$q_{25\%}$	$q_{50\%}$	$q_{90\%}$
CE-A	151.4	551K	0	10	303
CE-B	103.8	87K	16	1331	3999
CE-C	81.9	205K	0	26	408
CE-D	58.4	336K	0	0.20	203
CE-E	51.6	184K	0	2.8	150
CE-F	49.1	155K	0	0.6	87
CE-G	44.7	209K	0	0	73
CE-H	44.6	217K	0	0.1	78
CE-I	42.9	132K	0	3.6	83

For instance, considering the workload, the standard deviation is between 1 and 3.5 times as large as the mean. Moreover, variability as expressed by the standard deviation is positively correlated with the median (correlation coefficient 0.98) and the mean (correlation coefficient 0.99). Similar results are true for the interquartile range.

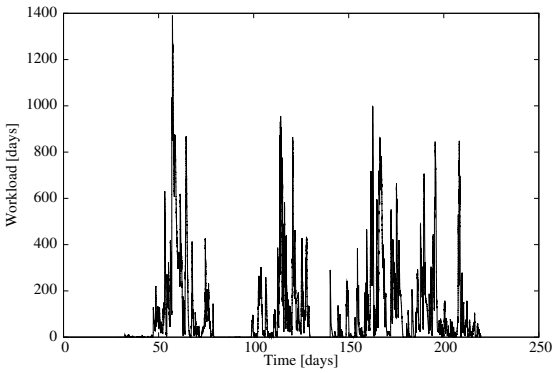

Fig. 1. Time series of the workload for CE-A

Visual inspection of the workload time series indicates that this variability is not uniform, but corresponds to different regimes. Fig. 1 shows a $1,400$-day burst at day 60. Similar but lower, peaks repeat afterwards, and the trace shows an irregular alternation of quiet and loaded segments.

De-trending: Differenciation is a standard tool to de-trend a series. The sampling frequency should be high enough to make it possible for the analysis to provide practically useful output, but should remain close to the operational timescale of the analyzed

system. The average number of job arrivals within an hour is in the range of 22 – 138 for the four top CEs. This suggests a sampling frequency in the order of 10^{-3} - 10^{-4} Hz because 1) each value of the differenced series cover several hundred jobs on average, and 2) the practical timescale of interest was the behavior of the system in the order of hours and above. The time series $W(t)$ is obtained by this subsampling.

III. Model discovery through MDL

This section sketches the *Auto-PARM* method proposed by Davis et al. [5] for structural estimation of breakpoints in non-stationary time series.

An autoregressive model of order p (AR(p)) is defined by

$$W(t) = \gamma + \phi_1 W(t-1) + \ldots + \phi_p W(t-p) + \sigma\epsilon_t,$$

where ϵ_t is white noise with mean 0 and variance 1. $W(t)$ is thus a linear combination of the previous data, and a noise. A piecewise AR model describes a finite length, discrete time, non-stationary time series as consecutive segments of stationary time series that each are independent AR processes. The edges of the segments are the *breakpoints*. An important argument for focusing on piecewise AR models is that they are dense in the class of locally stationary processes with continuous spectral densities and because efficient algorithms exist for fitting an AR model.

Given the breakpoints and the AR orders, the estimation of the model parameters for each segment is straightforward using the Yule-Walker method. Thus, finding a best fitting model from the piecewise AR class is equivalent to finding the number of segments m, the locations of the m breakpoints and the AR orders $(p_i)_{i=1\ldots m}$. [5] applies the Minimum Description Length (MDL) principle [28] to select the best model as the one that produces the shortest code length completely describing the observed data. The objective function is derived as

$$CL = \log m + (m+1)\log n + \sum_{j=1}^{m+1} \log p_j \quad + \sum_{j=1}^{m+1} \frac{p_j+2}{2} \log n_j + \sum_{j=1}^{m+1} \frac{n_j}{2} \log(2\pi\hat{\sigma}_j^2),$$

where n is the total length of the series, n_j is the number of points in the jth segment, and $\hat{\sigma}_j^2$ is an estimate of the variance of the j^{th} segment. Minimizing the code length function requires a tradeoff between the number of breakpoints and the complexity of the segments: segments that extend over different regimes will tend to require higher order AR models, and more variability.

The search space for breakpoints is very large, and the optimization problem is ill conditioned. Davis proposes to tackle the optimization problem by a genetic algorithm, which encodes a solution as a set of chromosomes bearing the order of the AR model for segment j at the selected breakpoints. This encoding is further constrained, so that the length of the segment is large enough to provide good estimates to the parameter of the related AR process (*min_span* parameter), and to limit the order of the process. Termination is decided by empirical convergence (identical best chromosome along a fixed number of generations) or when a predefined number of iterations is reached. To limit the computational complexity, crossover is allowed only inside sub-populations, with periodic migration across the islands. The running time is characterized by M, the number of migrations.

IV. Model Validation Methodology

The MDL procedure optimizes a target function that captures both the segmentation (locations of the breakpoints), and AR models inside each segment. As it has been shown experimentally to be able to correctly detect change of regimes in series which are piecewise, but not AR on each segment, the segment models and the segmentation should be checked independently. The issue is to build indicators that are detailed enough to capture the potentially differentiated accuracy of the model in various locations. For instance, the Mean Squared Error, or any other cumulative indicator, does not reveal which segments are correctly modeled. The indicators should also be concise enough to provide a quantitative measure of accuracy.

A. Model Accuracy

The AR model for each segment is validated by checking first the stationarity of the fitted AR model

inside each segment, and second the independence of the residuals, through appropriate statistical tests.

Stationarity: For the AR(p) model to be stationary, the roots of the characteristic polynomial of the AR model $\Phi(z) = 1 - \phi_1 z - \ldots - \phi_p z^p$ should lie outside the unit circle. Technically the Yule-Walker estimation procedure for the coefficients ensures that this condition is met. It is however known that when characteristic polynomial has root(s) close to unity, the fitted model is at the limit of being stationary and therefore should be examined carefully. The null-hypothesis of the Phillips-Perron test is that there is a unit root of the characteristic polynomial. Thus, we used this test to analyze the stationarity independently of the goodness of fit.

Independence of the residuals: Given an AR model, the residuals in each segment should be white noise. Testing for white noise amounts to checking the autocorrelation of the residuals at all lags. Choosing the appropriate statistics is not a closed question [4], and should take into account the specific properties of the data distribution. We choose to use a combination of Ljung-Box and Dufour-Roy tests. The Ljung-Box test is the classical parametric test, and is considered to be reliable when the size of the data set is large enough, because the estimates of the correlation are asymptotically a gaussian white noise. Dufour-Roy (a rank test) makes it possible to examine more precisely which part of the data does the AR model not explain.

B. Model Stability

The fitted model is not stable when repeating the model selection leads to heavily different models. The distribution of results from the internal randomization of the genetic algorithm may give a hint, but is not an independent indicator. We thus evaluated the stability of the segmentation through *parametric bootstrapping* [8]. This procedure creates k samples of the piecewise AR model, namely the breakpoint locations and the parameter vectors; the size of each sample is n, the size of the original series. Then, *Auto-PARM* is applied to each sample. These k segmentations provide statistics (e.g. mean, variance) and confidence intervals for the breakpoint locations and AR orders.

V. Experimental Results

A. Experimental setting

AutoParm features internal randomization (decision on mutation etc.). Thus, for each experiment, the procedure is repeated 20 times and the results providing the smallest description length is selected. The parameters are as follows: 100 islands of size 50, the 2 best chromosomes on island n migrates to island $(n+1) \mod 100$ at every 5^{th} offspring. The convergence criterion is the stability of the overall best chromosome for 10 consecutive migrations, and was met in all experiments. The complexity of the optimization landscape translates to a high computational complexity: 1 hour is typically required for one model selection.

B. First examples

We first go through the results of one run of AutoParm on CE-A and CE-B, which correspond to two different modes of grid usage, as seen in Section II-C. Fig. 2 displays the differentiated workload and the breakpoints, together with the AR orders. Table II gives the parameters of the models. For CE-A, the first result is that low-order AR models are the most frequent: seven segments are white noises (*i.e.* AR(0)) and six are AR(1). White noises totalize 49% of the whole measurements. These weak correlations, and the fact that the estimated variance for most segments is very high, typically twenty times larger than the mean, can be interpreted as the result of a poor, but effectively mixing, load balancing policy, or as an intrinsic feature of the job arrival process. It is important to notice that the size of the corresponding segments is large enough to have authorized for a much higher order (e.g. the *min_span* parameter is 20 for order 6). Segments 18, 19 and 20 actually exhibit higher orders (respectively 6, 5 and 5), showing that the procedure is able to discover more correlated models when adequate. CE-B involves much more long jobs than CE-A and the resulting workload model is more complex, both with respect to the number of breakpoints (30 instead of 21), and to the AR orders: for instance, the third segment is

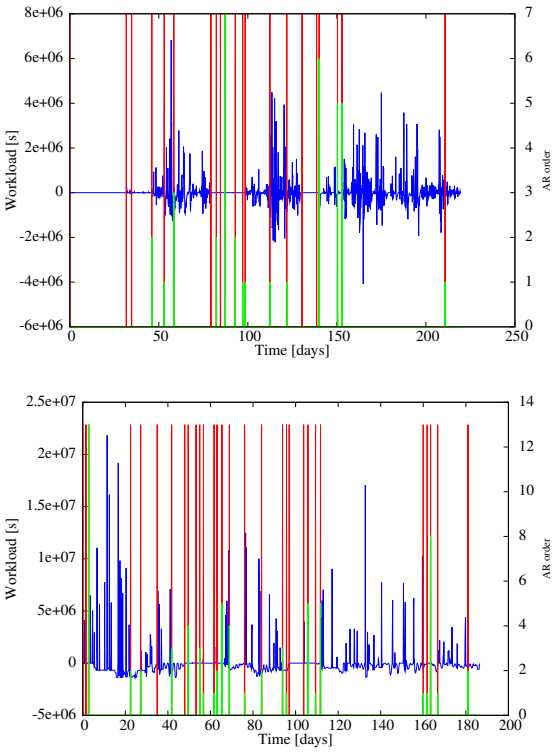

Fig. 2. Representation of the AR model for CE-A (upper) and CE-B (lower): the horizontal axis is time, the left vertical axis is the differentiated series, the right vertical axis is the order of the model in the next segment; e.g. the AR order for the 6th segment is 3 in the upper graph.

AR(13), indicating a correlation with three days old load. Nonetheless, the order of half of the segments is 0 or 1, comforting the diagnostic of a weakly correlated load.

C. The optimization landscape

The repeated runs (*restarts*) of AutoParm provide a first approximation of the optimization landscape for each data set. A complete sensitivity analysis would have to run experiments with different initialization values; due to the high computational cost of the method, we focus on the internal randomization. Table III presents the summary statistics for the four CEs. The values both for the algorithm indicators and for the model parameters are clearly consistent inside each experiment, and this holds for the four

TABLE II
THE MODEL PARAMETERS FOR CE-A AND B
j: SEGMENT INDEX, n_j: SEGMENT LENGTH, p_j: AR ORDER, γ_j: SEGMENT MEAN

CE-A			
j	n_j	p_j	γ_j
1	274	0	0.00E+00
2	26	0	-5.98E+02
3	98	0	5.98E+01
4	60	2	2.93E+04
5	47	1	1.69E+05
6	180	3	-3.18E+04
7	26	0	0.00E+00
8	20	2	-2.40E+01
9	21	0	0.00E+00
10	51	7	5.68E+01
11	36	2	0.00E+00
12	12	1	-6.99E+00
13	120	1	3.18E+03
14	82	1	4.82E+04
15	74	1	-3.66E+04
16	71	0	0.00E+00
17	12	0	-3.94E+01
18	89	6	3.61E+02
19	22	5	3.17E+03
20	500	5	-4.68E+03
21	74	1	-2.55E+03

CE-B			
j	n_j	p_j	γ_j
1	14	0	4.98E+05
2	12	0	0.00E+00
3	171	13	2.96E+05
4	42	2	-1.07E+06
5	68	2	-2.71E+05
6	60	0	3.86E+05
7	54	3	-3.96E+05
8	15	0	0.00E+00
9	33	4	4.78E+03
10	16	0	0.00E+00
11	16	3	2.41E+04
12	44	1	-3.85E+03
13	13	1	1.35E+05
14	21	2	0.00E+00
15	31	5	8.92E+05
16	63	4	-3.16E+05
17	70	1	3.36E+05
18	86	2	-2.52E+05
19	17	3	-6.82E+05
20	12	1	-4.22E+05
21	60	0	-1.56E-01
22	18	0	6.08E+05
23	32	5	0.00E+00
24	21	1	-3.73E+04
25	418	5	2.66E+04
26	17	1	-1.82E+05
27	15	1	2.10E+01
28	30	8	-2.98E+05
29	124	1	6.97E+04
30	49	2	-2.95E+05

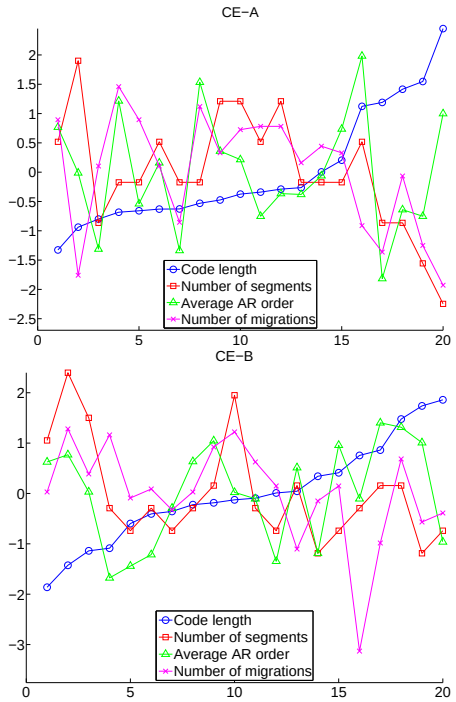

Fig. 3. Results of each restart for CE-A and B . The horizontal axis is the restart number ordered by increasing CL, the vertical axis corresponds to all parameters after standardization

TABLE III
MEAN AND STANDARD DEVIATION (BRACKETED) OF THE MODEL PARAMETERS AND ALGORITHM INDICATORS, RESCALED (SEE TEXT) OVER THE RESTARTS OF THE GA. m IS THE NUMBER OF SEGMENTS, $\overline{p}$ IS THE AVERAGE AR ORDER, CL IS THE CODE LENGTH, M IS THE NUMBER OF MIGRATIONS.

	CE-A	CE-B	CE-C	CE-D
m	20.25 (1.41)	27.65 (2.17)	29.60 (1.96)	20.65 (1.68)
$\overline{p}$	1.57 (0.31)	2.12 (0.40)	1.95 (0.35)	1.49 (0.45)
CL	2.04E+04 (4.45E+01)	2.17E+04 (6.56E+01)	1.84E+04 (7.55E+01)	1.75E+04 (3.93E+01)
M	128 (17.7)	150 (16.8)	156 (26.8)	118 (16.8)

experiments. Fig. 3 plots the detailed results for CE-A and B, together with the number of migrations. The values have been standardized (transformed to zero average and unit variance) in order to visualize the trends; the restarts have been ordered by increasing CL, thus the first points are the best fits. The rightmost (worst) five restarts for CE-A show a significantly larger code length, together with a smaller number of segments and a smaller number of migrations. In these cases, AutoParm gets soon stuck into sub-optimal solutions where the variance of the noise is high. This confirms the need for the restart procedure. From this point, the results are reported only for the best restart.

Fig. 4 shows the p-values of the Ljung-Box test for the whiteness of the residuals considering the whole segments. The null hypothesis is that the neighboring residuals are uncorrelated, thus the larger p-value, the better. The segments too short to run the tests are omitted (1% of the measurement time for CE-A and 9% for CE-B). The p-values are typically far from 0 therefore the whiteness hypothesis cannot be rejected for most of the segments. Nevertheless, the results for a number of segments lead to the rejection of the whiteness hypothesis at the 5% significance level. In these cases, the AR model is likely to be an approximation of a more complex model. These results are confirmed by a more detailed analysis with the Dufour-Roy test, omitted here by lack of space, and available in [26].

As there is no obvious relationship between the AR order of the segments and the test results, it is unlikely that the MDL method is in this case biased against high order models. There is some relationship with the length of the segments. In Table IV column 5% (resp. 10% and 20%) contains the fraction of the total length of the time-serie formed by the segments for which the p-value is above or equal .95 (resp .90 and .80); column $\geq$ 50% contains the sum of the length of the segments for which the p-value is less than .50. Except for CE-A, the p-value of the test results is over 0.80 for the largest part of the traces.

TABLE IV
FRACTION OF THE TRACE COVERED BY SEGMENTS WITH HYPOTHESIS OF UNCORRELATED RESIDUALS NOT REJECTED AT SIGNIFICANCE LEVEL $1 - \alpha$

α	5%	10%	20%	$\geq$ 50%
CE-A	41.1%	47.4%	50.6%	11.7%
CE-B	52.3%	64.1%	74.7%	16.1%
CE-C	45.2%	48.7%	68.0%	9.1%
CE-D	16.7%	50.2%	74.6%	18.7%

D. Stability

The previous results show that the piecewise AR model adequately describes a significant part of the experimental data. The question is now if the descriptions are not exceedingly accurate: would a small change in the experimental data induce significant changes in the model? In this case, the procedure would have over-fitted the data, and the model will be considered to be *unstable*. Yet the motivations for possible variability are multiple, for instance because the scheduler randomly break ties, and also because of the possible transient errors in measurements, thus testing stability is required to further validate the models. In this section, we will assess the stability of the segmentation itself: how frequent is a breakpoint across the segmented samples? We will also analyze the variability of the order parameters.

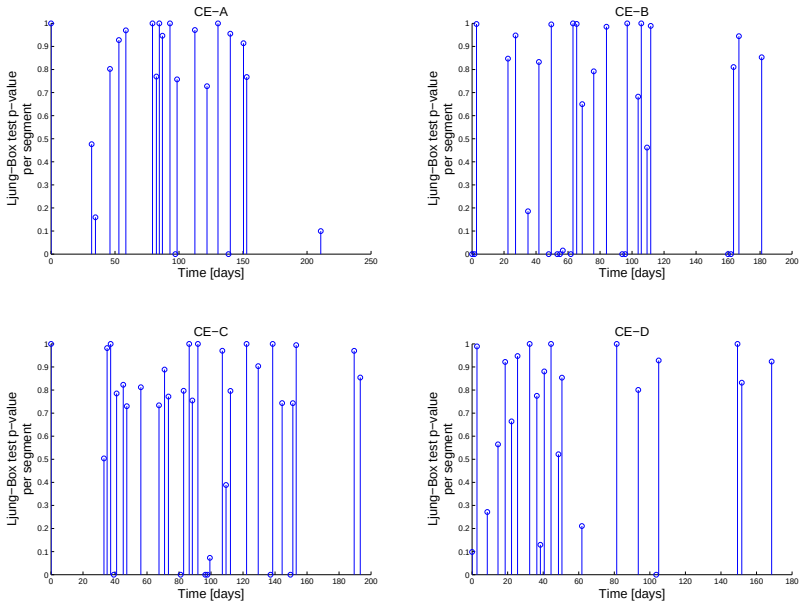

Fig. 4. Independence of the residuals: the Ljung-Box test

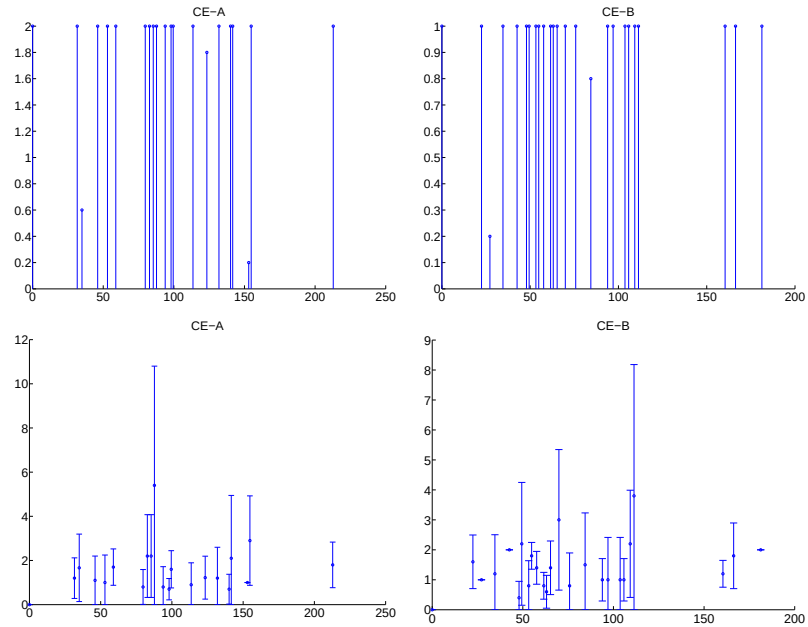

Fig. 5. Stability analysis. Upper graphs: frequency of the breakpoints. Lower graphs: AR order $\pm$ one standard deviation. The horizontal axes show the time in days

TABLE V
DISTANCE BETWEEN NEAREST NEIGHBOR BREAKPOINTS IN THE BOOTSTRAPPED SAMPLE FROM THE FOUR CES

Distance	Frequency			
	CE-A	CE-B	CE-C	CE-D
0	63.35%	46.22%	46.51%	59.55%
1	21.99%	15.97%	20.93%	8.99%
2	2.09%	5.04%	5.43%	3.37%
3	1.05%	3.36%	1.55%	3.37%
4	1.05%	2.52%	0.78%	0.00%
5	0.00%	3.36%	0.78%	1.12%
6	0.00%	0.84%	0.00%	0.00%
7	0.00%	0.00%	0.00%	1.12%
8	0.00%	1.68%	0.78%	0.00%
9	0.00%	0.84%	0.00%	0.00%
10	0.52%	0.00%	1.55%	0.00%
> 10	9.95%	20.17%	21.71%	22.47%

Evaluating stability would require other samples of the load process but no other realization of the experimental data is available. To cope with this difficulty, we have at least to assume that the experimental data are a reasonable representation of the "population" of scheduling actions and measurements. If this hypothesis holds, bootstrapping creates a sample of mock realizations of the process. In general, bootstrapping [8] is the technique which resamples from original data with replacement, assuming that the experimental data faithfully describe the population. Given the size, lack of homogeneity, and intrinsic correlation structure of the series (which is precisely the motivation for the piecewise model), naive resampling would not create a reasonable realization. Parametric bootstrapping can: new and truly piecewise AR processes are created from the model, namely the breakpoints, segment lengths and parameter estimates, the variability coming from the truly white residuals. Each of these realizations is then segmented with the AutoParm procedure, with restarts. The final result is an ensemble of models

$$\mathcal{S} = \{m_i, (n_i^j), (p_i^j), 1 \leq i \leq k, 1 \leq j \leq m_i\}$$

where k is the number of samples, m_i is the number of breakpoints in sample i, n_i^j the size of segment j in sample i and so on.

Breakpoints defined by $\mathcal{S}$ can be very close, but not identical across the bootstrapped samples. For instance, in CE-A, some samples provide the segment [46.18, 53.13], while other provide [46.06, 53.24] (the unit is the day). These segments should be considered as variants of the same one. On the other hand, some segmented samples feature a breakpoint in the range 34.84-35.30, while the other samples find no breakpoint between 32.00 and 46.06, denoting a true disagreement between the segmentation results. The distances between one breakpoint and its closest neighbor (coming from possibly another sample), are shown in Table V for CE-A, B, C and D. There are clearly two regimes, small distances (variants) and large ones (true breakpoints). The close breakpoints must be clustered before deciding which breakpoints are frequent. The clustering threshold is conservatively fixed at 10 points, as it is the lower bound for fitting the simplest AR models (0 or 1) with statistical significance. Note that the physical unit depends on the sampling frequency, but is consistent across the k bootstrapped samples of the same series. Fig. 5 (upper graph) displays the frequency of the breakpoints after clustering, for CE-A and CE-B. Despite their notable difference concerning the suspicion on the independence of residuals, they are remarkably stable: only 2 or 3 breakpoints are not recognized in all samples. Finally, Fig. 5 (lower graph) shows the variability of the AR order.

E. Building robust models

As we have seen in the previous results, the segmentation of the bootstrapped samples are generally in good accordance, but not identical. Moreover, the AR order may show significant variability. Bootstrapped aggregation, or *bagging* [2] gives theoretical foundations to model reconciliation, either by averaging or voting. Here, the number of models k is bound to be small due to computational time, thus voting should be preferred [17]. The choice of the best voting strategy is (and is likely to remain) an open question; in our case, the simple majority voting will be used, with a random choice for breaking ties. Fig. 6 gives the parameters (order reported at breakpoint locations) of the resulting models.

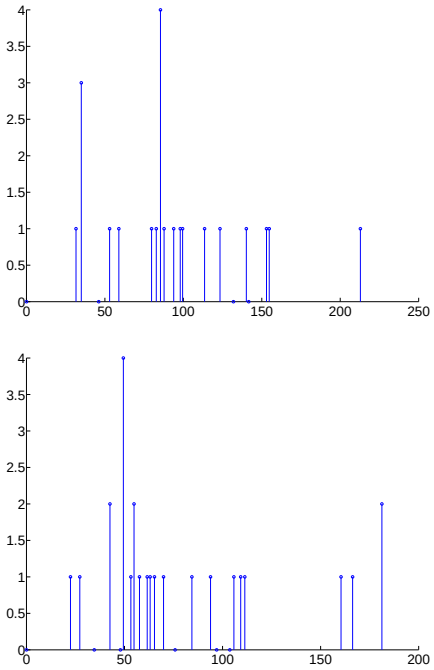

Fig. 6. Bagged model for CE-A (upper) and CE-B (lower)

VI. Related Work

Explanatory models of the workload in HPC systems [7], [29] characterize the distributional properties of the quantities of interest for job behavior (e.g. inter-arrival time, queuing delay, or execution time) using different parametric models. More recently, efforts [14], [18], [19], [23], [30] address grid systems along the same path. Another extensive literature targets *predictive* models, either by time series analysis methods [6], [18], [25], [34] or statistical ones [3], [16], [35]. This direction of research selects a specific view of the system (short time range for time-series, or features of the job and target execution support) in order to improve the predictive accuracy at the expense of a general model. Finally, in the context of Data Center, research in Reinforcement Learning scheduling [31], [33] creates an implicit model of the offered workload inside the value function discovered by the learner.

Our work shares the explanatory goal of the first approach, and the techniques of time-series analysis of some of the second one. It differs in two significant ways, which as far as we know have not yet been explored. First, we aim at discovering the structural breaks in the model, and we exploit an unified method for discovering both the model and its ruptures, rather than assuming stationary processes or decoupling the models and discovery techniques for changes of regime and intra-regime behavior. Second, the bootstrapping strategy addresses the lack of confidence associated with the uncertainties and non-reproducibility of the acquisition process.

VII. Conclusion

We have presented a workload measurement obtained from the Grid Observatory. We evaluated the performance of MDL-based model selection for the workload of the four most heavily loaded CEs. The results were validated by whiteness and autoregressive model tests. Also, a parametric bootstrap method was proposed for analyzing the stability of the model. The main contribution of our evaluation is to show that our workload can be explained by piecewise autoregressive models to a large extent. Moreover, the order of the models is mostly low to moderate. Finally, we showed that the bootstrapped samples can be reconciliated through bagging. These conclusions apply to the EGEE workload. Future work will explore their validity on the activity profiles of other infrastructures and research communities, in particular those provided by the Grid Workloads Archive [9].

The most significant limitation of the method is the poor scalability of the genetic algorithm with respect to the length of the time series. Systematic exploitation, on all sites and at various time scales, or transposition to the prediction context, calls for much faster model selection. We are currently exploring an alternative optimization algorithm along the same MDL principle. We will then propose a continuous segmentation of the grid traffic as part of the *building behavioral models* activity of the Grid Observatory.

ACKNOWLEDGMENTS

This work was partially funded by the DIM program of Region Ile de France and the Digiteo Fundation. The data sets have been provided by the Grid Observatory. The Grid Observatory is part of the EGEE-III EU project INFSO-RI-222667.

REFERENCES

[1] The Grid Observatory Portal. www.grid-observatory.org.

[2] L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2), 1996.

[3] J. Brevik, D. Nurmi, and R. Wolski. Predicting bounds on queuing delay in space-shared computing environments. In *IISWC*, pages 213–224, 2006.

[4] P. Burns. Robustness of the ljung-box test and its rank equivalent. In *The Journal of Derivatives*, pages 7–18, 2002.

[5] R. A. Davis, T. Lee, and G. Rodriguez-Yam. Structural break estimation for nonstationary time series models. *J. American Statist. Assoc.*, 101:229–239, 2006.

[6] P. A. Dinda and D. R. O'Hallaron. Host load prediction using linear models. *Cluster Computing*, 3(4):265–280, 2000.

[7] A. B. Downey. Using queue time predictions for processor allocation. In *IPPS'97, JSSPP'97*, pages 35–57, 1997.

[8] B. Efron. Bootstrap. another look at jackknife. *Ann. Statist.*, 7(1):1–26, 1979.

[9] A. Iosup et al. The Grid Workloads Archive. *Future Gener. Comput. Syst.*, 24(7):672–686, 2008.

[10] E. Laure et al. Programming the Grid with gLite. *Computational Methods in Science and Technology*, 12(1):33–45, 2006.

[11] F. Gagliardi et. al. Building an Infrastructure for scientific Grid computing: status and goals of the EGEE project. *Phil. Trans. of the Royal Society A*, 1833, 2005.

[12] S. Andreozzi et al. Glue Schema Specification, V1.3. Technical report, Open Grid Forum, 2008.

[13] R. Gott III. Implications of the copernican principle for our future prospects. *Nature*, (363):315–319, 1993.

[14] L. Ilijašić and L. Saitta. Characterization of a computational grid as a complex system. In *Procs. of GMAC '09*, pages 9–18, 2009.

[15] S. Jha, M. Parashar, and O. Rana. Investigating autonomic behaviours in grid-basedcomputational science applications. In *Proceedings of GMAC'09*, pages 29–38, 2009.

[16] B.-D. Lee and J. M. Schopf. Run-time prediction of parallel applications on shared environments. In *CLUSTER*, pages 487–491, 2003.

[17] T-W Lee and Y. Yang. Bagging binary and quantile predictors for times series. *Journal of econometrics*, 135(1-2):465–497, 2006.

[18] H. Li and M. Muskulus. Analysis and modeling of job arrivals in a production grid. *SIGMETRICS Perform. Eval. Rev.*, 34(4):59–70, 2007.

[19] D. Lingrand, T. Glatard, and J. Montagnat. Modeling the latency on production grids with respect to the execution context. *Parallel Computing*, 35(2009):493–511, 2009.

[20] C. Loomis. The Grid Observatory. In *Grids Meet Autonomic Computing workshop at ICAC'09*. ACM, 2009.

[21] M. Macias, O. Rana, G. Smith, J. Guitart, and J. Torres. Maximising revenue in grid markets using an economically enhanced resource manager. *Concurrency and Computation: Practice and Experience*, 2008.

[22] J. Meng, S. T. Chakradhar, and A. Raghunathan. Best-effort parallel execution framework for recognition and mining applications. In *IPDPS*, pages 1–12, 2009.

[23] N. Mi, G. Casale, L. Cherkasova, and E. Smirni. Injecting realistic burstiness to a traditional client-server benchmark. In *Proceedings of ICAC '09*, pages 149–158, 2009.

[24] A. Mutz, R. Wolski, and J. Brevik. Eliciting honest value information in a batch-queue environment. In *Proceedings of GRID '07*, pages 291–297, 2007.

[25] F. Nadeem, M. M. Yousaf, R. Prodan, and T. Fahringer. Soft benchmarks-based application performance prediction using a minimum training set. In *e-science'06*, 2006.

[26] T. Éltető, C. Germain-Renaud, P. Bondon, and M. Sebag. Discovering linear models of grid workload. RR 7112, INRIA, 2009.

[27] J. Perez, C. Germain-Renaud, B. Kégl, and C. Loomis. Utility-based reinformcement learning for reactive grids. In *The 5th IEEE ICAC Autonomic Computing*, 2008.

[28] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, 1989.

[29] W. Smith, V. E. Taylor, and I. T. Foster. Using run-time predictions to estimate queue wait times and improve scheduler performance. In *IPPS/SPDP '99, JSSPP'99*, pages 202–219, 1999.

[30] O. Sonmez, N. Yigitbasi, A. Iosup, and D. Epema. Trace-based evaluation of job runtime and queue wait time predictions in grids. In *Proceedings of HPDC '09*, 2009.

[31] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani. On the use of hybrid reinforcement learning for autonomic resource allocation. *Cluster Computing*, 10(3):287–299, 2007.

[32] D. Thain, J. Bent, A. Arpaci-Dusseau, R.Arpaci-Dusseau, and M. Livny. Gathering at the well: Creating communities for grid i/o. In *Procs of Supercomputing*, 2001.

[33] D. Vengerov. A reinforcement learning approach to dynamic resource allocation. *Eng. Appl. Artif. Intell.*, 20(3), 2007.

[34] R. Wolski, N. T. Spring, and J. Hayes. Predicting the cpu availability of time-shared unix systems on the computational grid. *Cluster Computing*, 3(4):293–301, 2000.

[35] L. Yang, J. M. Schopf, and I. Foster. Conservative scheduling: Using predicted variance to improve scheduling decisions in dynamic environments. In *SC '03: Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, page 31, 2003.

[36] X. Zhang, C. Furtlehner, J. Perez, C. Germain-Renaud, and M. Sebag. Toward autonomic grids: analyzing the job flow with affinity streaming. In *Proc. of the 15th ACM SIGKDD*, pages 987–996, 2009.