

Discovering Linear Models of Grid Workload

Tamás Éltető — Cécile Germain-Renaud — Pascal Bondon

N° 7112

Novembre 2009

Domaine 1

 ***Rapport
de recherche***

Discovering Linear Models of Grid Workload

Tamás Éltető*, Cécile Germain-Renaud†, Pascal Bondon‡

Domaine : Mathématiques appliquées, calcul et simulation
Équipes-Projets TAO

Rapport de recherche n° 7112 — Novembre 2009 — 20 pages

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. The goal of this paper is to 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 assess the quality of these models on a statistically rigorous basis? 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 presents a bootstrapping strategy for building more robust models from the limited samples at hand. This study is based on exhaustive information representative of a significant fraction of e-science computing activity in Europe.

Key-words: Grid workload, Minimum Description Length principle, autoregressive model, segmentation

* LRI, Université Paris-Sud 11, CNRS, INRIA, eltetot@lri.fr

† LRI, Université Paris-Sud 11, CNRS, INRIA, cecile.germain@lri.fr

‡ LSS, Supelec, CNRS, Pascal.Bondon@lss.supelec.fr

No French Title

Résumé : Pas de résumé

Mots-clés : Pas de motclef

1 Introduction

Large-scale distributed computing systems, such as EGEE (Enabling Grid for E-sciencE) [8], allocate computing resources following the matchmaking principle pioneered by Livny [27]: 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: 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 [2] capacities. An extensive body of research e.g. to cite only a few [17, 20, 22, 26] 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 [10] ("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 realworld experimentation is hardly possible. Second, significant experiments with simulators or analysis require large datasets. These datasets may be publicly available, but comparative experiments are nearly unknown in the grid community, (while they are mandatory in other areas such as computer architecture or machine learning) and experiments on high level concepts such as autonomic programming models [12, 18], 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 more robust models from the limited samples at hand. This study is based on exhaustive information covering more than a year of of the flagship EU grid infrastructure EGEE, and can thus be considered as representative of a significant fraction of e-science computing activity in Europe.

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 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 2 describes the dataset, its grid context, and the derivation of the times-series workload process from the empirical data. Section 3 defines the piecewise AR

model and describes a model selection procedure. Section 4 presents the validation methodology. Section 5 discusses the experimental results and presents the bootstrapping strategy. Section 6 discusses related work, before the conclusion.

2 The Workload Process

2.1 EGEE and gLite

For the sake of precision and because the experimental dataset come from EGEE, this section will describe its scheduling under gLite, its major middleware. gLite integrates the sites' computing resources through a set of middleware-level services (the Workload Management System, the WMS), which 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 [9]). As other high performance space-shared systems, most EGEE sites implement their scheduling policies through multiple FIFO queues and complex tuning of configuration files.

2.2 Workload Definition

The workload in grid context is the same as the *backlog* of queuing systems. Backlog at time t has two definitions a) the amount of unfinished work in the system and b) 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

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

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

$$C^{\text{RS}}(t) = \sum_{j:T_s(j)<t} T_e(j) - T_s(j).$$

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

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

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.

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

This definition implicitly assumes an 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 it ran, thus on the dispatch system, except if the machine panel is fully homogeneous. While being inexact, the homogeneity assumption is not very far away from the reality: grid sites are institutional ones, with reasonable coherency.

2.3 The dataset

This study is based on exhaustive information covering all the gLite monitored jobs in the EGEE grid, from August 2008 to March 2009, collected by the Real Time Monitor project, and is publicly available through the Grid Observatory portal. For the purpose of this paper, the significant quantities recorded are as follows: $T_a(j)$ is the `logmonitor_accepted_Epoch` timestamp; $T_s(j)$ is the `logmonitor_running_Epoch` timestamp; $T_e(j)$ is the `logmonitor_done_Epoch` timestamp.

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 [31] reports situation where the logmonitor service become clogged, and is not consistent with the timestamps 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.

Descriptive Statistics

	Total [years]	Jobs	percentile [days]		
			$q_{25\%}$	$q_{50\%}$	$q_{75\%}$
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
CE-J	38.3	125K	0	0	0

Table 1: Descriptive statistics for the top ten CEs

Table 1 presents the statistics the ten CEs featuring the largest total load (the real names of the CE are omitted for privacy reason). For lack of space, we cannot detail the statistics, but all criteria for very high variability (variance, interquartile range, maximum) are met. For instance, 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 mean (correlation coefficient 0.99) workload. Similar results are true for the interquartile range.

Visual inspection of the workload time series indicates that this variability is not uniform, but corresponds to different regimes. Fig. 1 shows a 1400 days burst at day 60. Similar but lower, peaks repeat afterwards. More generally, the trace shows an irregular alternance of quiet and loaded segments.

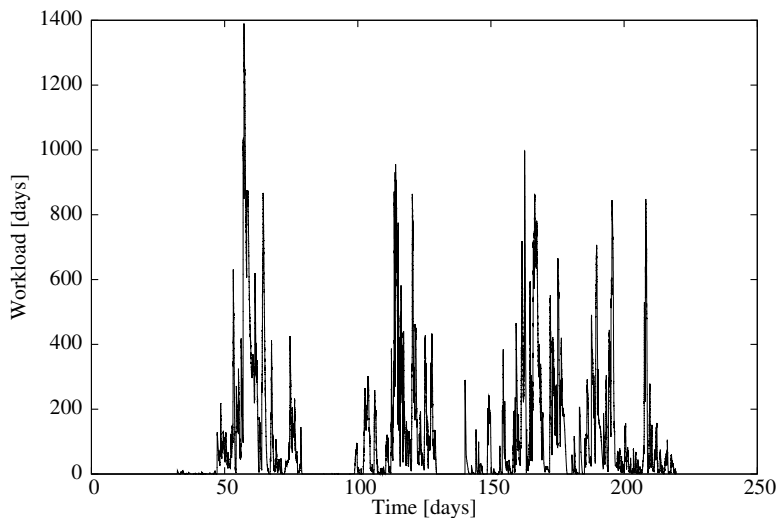


Figure 1: Time series of the workload for CE-A

Detrending

The straightforward way of removing a trend is to calculate the difference series. For this, we had to select an appropriate sampling frequency. The sampling frequency should be selected high enough to make it possible for the analysis to provide practically useful output, but remain below or near to the operational timescale of the analysed system. The average arrival rate of jobs was found to be between 0.0060 and 0.0383 1/s 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 in average, and 2) the practical timescale of interest was the behaviour of the system in the order of hours and above.

3 Model discovery through MDL

. This section sketches the *Auto-PARM* method proposed by Davis et al. [4] for structural estimation of breakpoints in non-stationary time series.

3.1 Piecewise AR models

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

$$X_t = \gamma + \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \sigma \epsilon_t,$$

where ϵ_t is white noise with mean 0 and variance 1. X_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. Thus approximating the empirical time-series with this kind of process has a theoretical foundation.

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 blocks m , the locations of the m breakpoints in the data and the AR orders $(p_i)_{i=1\dots m}$. [4] applies the Minimum Description Length (MDL) principle [23] to select the best model as the one that produces the shortest code length that completely describes the observed data. More precisely, the objective function is derived as

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

where n is the total length of the series, n_j is the number of points in the j th segment, and $\hat{\sigma}_j^2$ is an estimate of the variance of the j th block. Minimizing this function requires a tradeoff between the number of breakpoints and the complexity of the segments they define: 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 n 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 pre-defined number of iteration is reached. To limit the computational complexity, crossover is allowed only inside sub-populations, with period migration across populations (island model).

4 Model Validation Methodology

The MDL procedure optimizes a target function that captures both the segmentation (location 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 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, and which one are not. The indicators should also be concise enough to provide a quantitative measure of accuracy.

4.1 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. In both cases, appropriate statistical tests are applied.

Stationarity

For the AR(p) model to be stationary, the roots of the characteristic polynomial of the AR model $\Phi(z) = 1 - \phi_1 z - \dots - \phi_p z^p$ 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 1, then the fitted model is at the limit of being stationary and therefore it 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 analyse 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 e.g. [3], 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 reliable when the size of the dataset is large enough, because the estimates of the correlation are asymptotically a gaussian white noise. Dufour-Roy (a rank test) make it possible to examine more precisely which part of the data is not explained by the AR model (experiments not reported for lack of space).

4.2 Model Stability

The fitted model is not stable when a repeated model selection procedure might lead to heavily different models. The distribution of results from the internal randomization of the genetic algorithm may give a hint, but does not give an independent indicator. We thus evaluated the stability of the segmentation through *parametric bootstrapping* [7]. The procedure creates k samples of the piecewise AR model, namely the breakpoint locations and the parameters vectors; the size of each sample is n , the size of the original series. Then, the MDL segmentation described in Section 3 is applied to each sample. From these k statistics (e.g. mean, variance) and confidence intervals for the segmentation features (breakpoint locations and AR orders) can be obtained.

5 Experimental Results

5.1 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 $n + 1 \bmod 100$ at every 5th offspring. The convergence criterion is stability of the overall best chromosome for 10 consecutive migrations. In all experiments, the convergence criterion was met before the maximal number of migrations was reached. The complexity of the optimization landscape translates to a high computational complexity: 1 hour is typically required for one model selection.

5.2 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 2.3. Fig. 2, left graph, displays the

CE-A			
Segment	Length	AR order	mean
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			
Segment	Length	AR order	mean
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
31	1	-1	-3.68E+04

CE-C			
Segment	Length	AR order	mean
1	287	0	0.00E+00
2	18	2	1.61E+01
3	20	2	-8.42E+00
4	18	3	0.00E+00
5	16	0	-5.73E+02
6	37	1	4.66E+01
7	20	3	-2.48E+01
8	77	5	9.79E+02
9	99	1	2.57E+05
10	31	2	-7.38E+05
11	22	3	2.37E+02
12	67	1	6.07E+03
13	17	1	1.81E+01
14	31	2	3.50E+01
15	18	3	0.00E+00
16	31	2	1.23E+02
17	41	0	0.00E+00
18	12	1	0.00E+00
19	15	0	0.00E+00
20	68	0	-6.03E+01
21	21	1	1.32E+00
22	25	3	1.76E+03
23	88	2	-5.89E+03
24	64	0	1.14E+03
25	67	1	-5.02E-02
26	13	1	-6.99E+00
27	52	0	0.00E+00
28	46	4	-4.69E+02
29	14	0	0.00E+00
30	19	2	2.85E+03
31	314	3	-5.08E+03
32	34	5	-8.63E+01
33	229	2	-9.24E+03
34	1	-1	-1.19E+05

CE-D			
Segment	Length	AR order	mean
1	25	0	1.41E+02
2	51	3	1.02E+03
3	53	0	-6.63E+03
4	36	4	-7.95E+02
5	32	7	-2.12E+03
6	30	3	3.31E+02
7	60	8	-1.24E+04
8	36	0	0.00E+00
9	18	0	1.15E+00
10	20	0	1.80E+01
11	34	0	-1.53E+00
12	37	0	0.00E+00
13	18	0	-1.65E+02
14	98	1	6.88E+03
15	170	0	-6.80E+03
16	106	0	0.00E+00
17	88	1	-3.81E+03
18	13	1	1.30E+04
19	384	2	1.30E+03
20	22	0	0.00E+00
21	146	1	4.33E+02
22	23	1	-4.26E+04

Table 2: The model parameters for CE-A, B, C and D

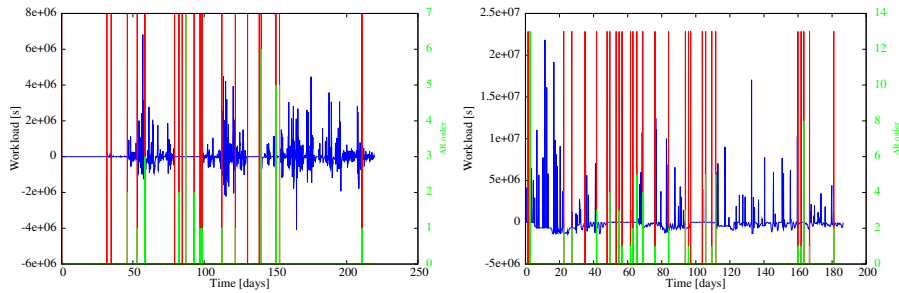


Figure 2: Representation of the AR model for CE-A (left) and CE-B (right): the horizontal axis is time, the left vertical axis is the differentiated series, the right vertical axis is the order of the model in the next segment; e.g. the AR order for the 6th segment is 3

differentiated workload and the breakpoints, together with the AR orders. Table 2 gives the parameters of the models. The first result is that low-order AR models are the most frequent: seven segments are AR(0), and six AR(1). Order 0 means that the series randomly fluctuates around a trend, the mean. These segments 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), despite for instance the large size of segment 20, showing that the procedure is able to discover more correlated models when adequate. Fig. 2, right graph, gives the same information for CE-B. The workload involves much more long jobs than CE-A and the resulting 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 AR(13), indicating a correlation with three days old load. Nonetheless, the order of half of the segments are is 0 or 1, comforting the diagnostic of a weakly correlated load.

5.3 The optimization landscape

	CE-A	CE-B	CE-C	CE-D
NS	20.25 (1.41)	27.65 (2.17)	29.60 (1.96)	20.65 (1.68)
ARO	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)
NM	128 (17.7)	150 (16.8)	156 (26.8)	118 (16.8)

Table 3: Mean and standard deviation (bracketed) of the model parameters and algorithm indicators, rescaled (see text) over the restarts of the GA. NS is the Number of Segments, ARO is the AR order, CL is the Code Length, NM is the number of migrations.

The repeated runs (*restarts*) of AutoParn provide a first approximation of the optimization landscape for each dataset. 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 3 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 experiments. Fig. 3 plots the detailed results for CE-A, B, C and D 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 in the upper right graph of Fig. 3 show a significantly larger CL, 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.

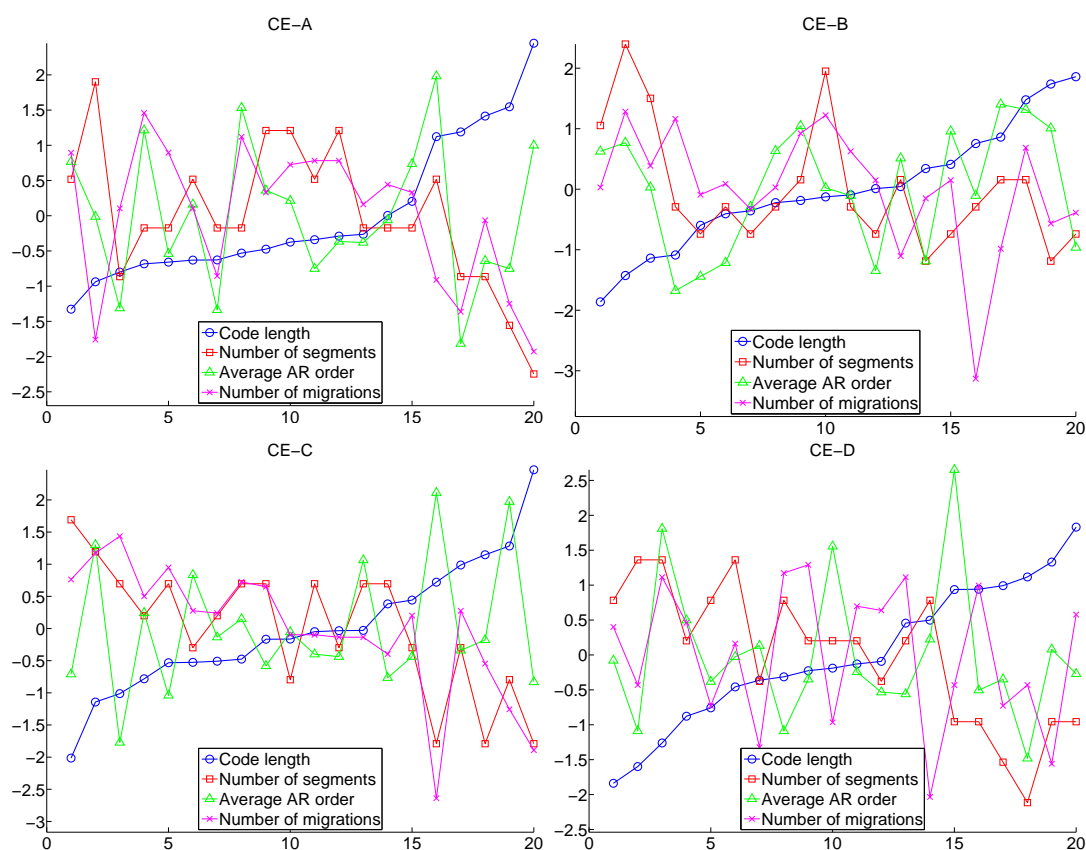


Figure 3: Results of each restart for CE-A, B, C and D. The horizontal axis is the restart number ordered by increasing CL, the vertical axis corresponds to all parameters after standardization

From this point, the results are reported only for the best restart.

The Philips-Perron test for unitary roots has been run on segments that are not AR(0). The null hypothesis is that 1 is a root of the characteristic polynomial of the autoregressive model, thus the smaller (null hypothesis rejected) the better (stationary process). In nearly all cases, the process in each segment can be safely considered as stationary, the exception belonging to segments that are otherwise problematic.

Fig. 4 show the p-values of hypothesis tests analysing the whiteness of the residuals of the AR models fitted to the different segments. The null hypothesis of the tests is that the neighbouring



Figure 4: Whiteness analysis of the segments in CE-A, B, C and D. The vertical axes show the p-values of the Dufour-Roy tests over the residuals of the fitted AR models for each segment.

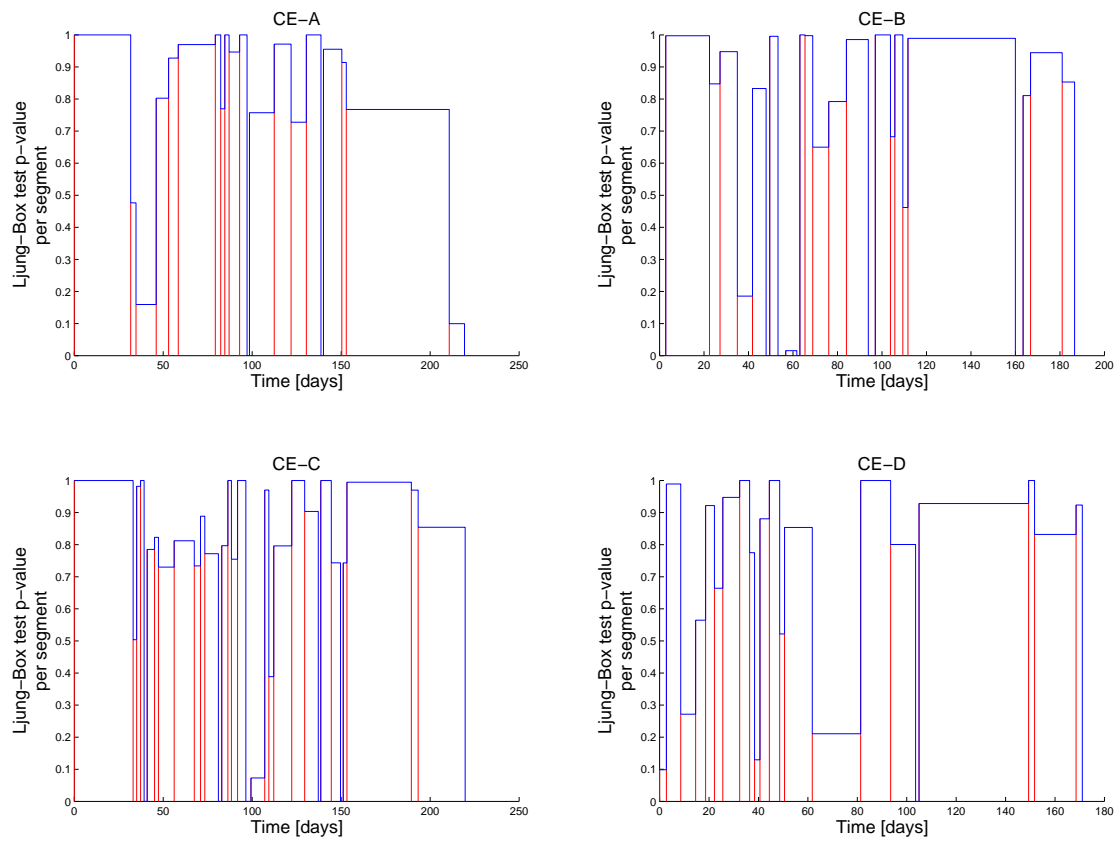


Figure 5: Independence of the residuals: the Ljung-Box test

residuals are uncorrelated, thus the larger (null hypothesis not rejected) the better. The residual series of each segment are cut into 1 day long pieces containing 9 data items and the lag 2 correlation is analysed. The number of data for one test is rather small therefore the Dufour-Roy test was used in the analysis because it is known to work well for small sample sizes. Using the graphs in Fig. 4 we are able to determine to what extent the different parts of the residuals within the segments conform with the null hypothesis.

For reference, the graphs in Fig. 5 show the p-values of the Ljung-Box test for the whiteness of the residuals. The null hypothesis here is that the neighbouring residuals are uncorrelated. Since the Ljung-Box test needs large sample size, we calculated one p-value for each segment. Nevertheless, some segments were still too short to run the tests therefore they are omitted.

Both in Fig. 4 and in Fig. 5, the p-values are typically far from 0 therefore we cannot reject the whiteness hypothesis for most of the segments. Nevertheless, the results for a number of segments lead to the rejection of the whiteness hypothesis with significance level of 5%. In these cases, the AR model is likely to be an approximation of a more complex model; as there is no obvious relationship between the 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 4 column 5% (resp. 10% and 20%) contains the sum of the length of 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.

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%

Table 4: Fraction of the trace covered by segments with hypothesis of uncorrelated residuals accepted at significance level $1 - \alpha$

5.4 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 overfitted 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.

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 allows to create a sample of mock realizations of the process. In general, bootstrapping [7] is the technique which resamples from original data with replacement, assuming that the experimental data faithfully describe the population. Given the size, inhomogeneity and intrinsic correlation structure of the series

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%

Table 5: Distance between nearest neighbor breakpoints in the bootstrapped sample from the four CEs

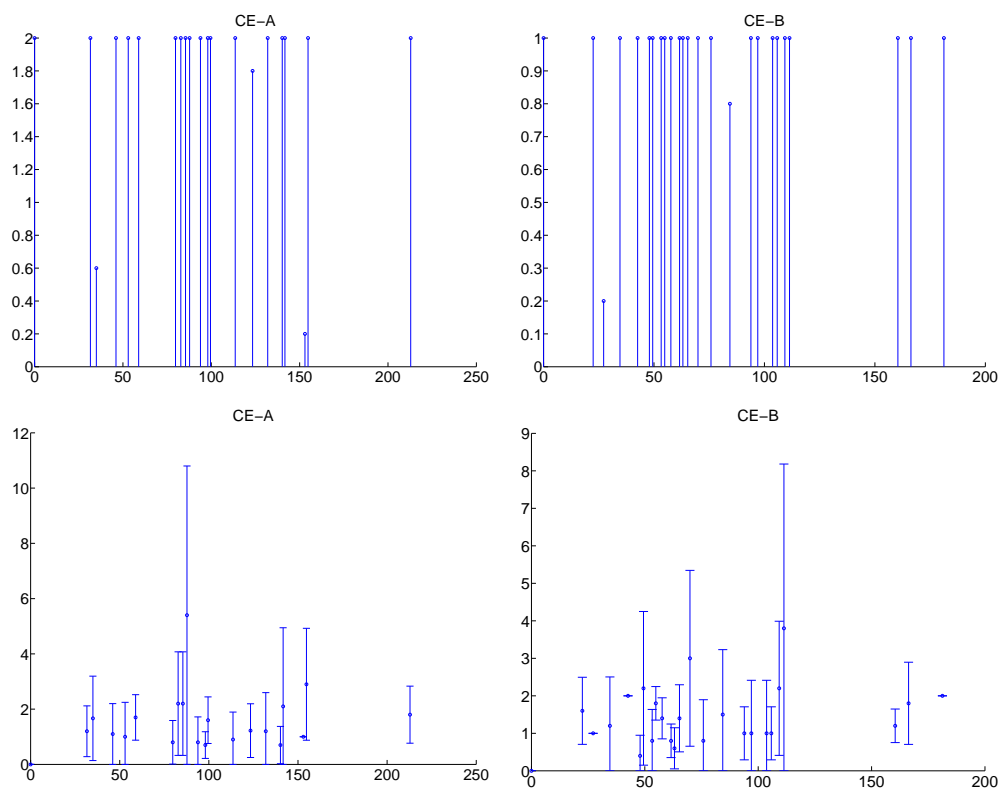


Figure 6: Stability analysis. Upper graphs: frequency of the breakpoints. Lower graphs: AR order \pm one standard deviation. Left graphs: CE-. Right graphs: CE-B. The horizontal axis is the time in days

(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]. 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 5 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 e.g. deciding which breakpoints are frequent. The clustering threshold is conservatively fixed at 10, as it is the lower bound for fitting the simplest AR models (0 or 1) with statistical significance. Fig. 6 (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 by in all samples. Finally, Fig. 6 (lower graph) shows the variability of the AR order.

5.5 Building robust models

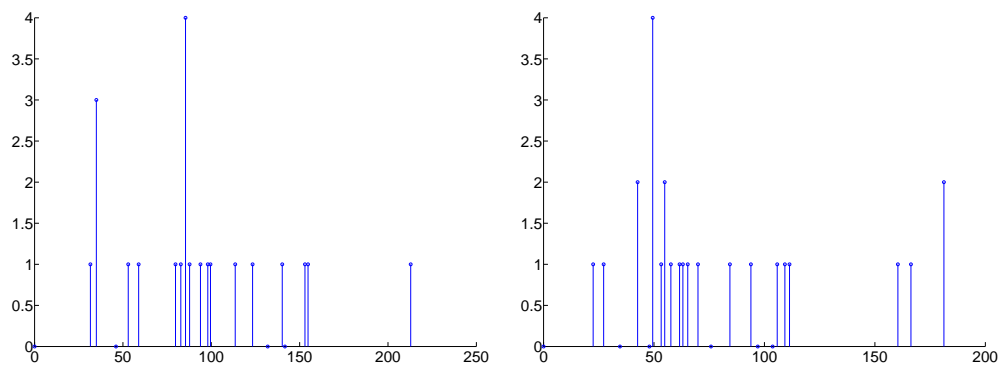


Figure 7: Bagged model for CE-A and CE-B

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* [1] gives theoretical foundations to model reconciliation, either by averaging or voting. In our case, the number of models k is bound to be small due to computational time, thus voting should be preferred [14]. The choice of the best voting strategy is (and is likely to remain) an open question, and in our case, the simple majority voting will be used, with a random choice for breaking ties. Fig. 7 gives the parameters (order reported at breakpoint locations) of the resulting models for CE-A, B, C and D.

6 Related Work

Explanatory models of the workload in HPC systems [6, 24] characterize the the distributional properties of the quantities of interest for job behavior (eg inter-arrival time, queuing delay, or execution time) using different parametric models. More recently, efforts [11, 15, 16, 19, 25] address grid systems along the same path. Another extensive literature targets *predictive* models, either by time series analysis methods [5, 15, 21, 29] or statistical ones [2, 13, 30]. 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 [26, 28] 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 behaviour. Second, the bootstrapping strategy addresses the lack of confidence associated with the uncertainties and non-reproducibility of the acquisition process.

7 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 analysing the stability of the model. The main contribution of our evaluation is to show that grid 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.

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 sales 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 Digeo Foundation. The datasets used in this work have been provided by the Grid Observatory www.grid-observatory.org. The Grid Observatory is part of the EGEE-III EU project INFSO-RI-222667.

Contents

1	Introduction	3
2	The Workload Process	4
2.1	EGEE and gLite	4
2.2	Workload Definition	4
2.3	The dataset	5
3	Model discovery through MDL	6
3.1	Piecewise AR models	6
4	Model Validation Methodology	7
4.1	Model Accuracy	7
4.2	Model Stability	8
5	Experimental Results	8
5.1	Experimental setting	8
5.2	First examples	8
5.3	The optimization landscape	10
5.4	Stability	14
5.5	Building robust models	16
6	Related Work	17
7	Conclusion	17

References

- [1] L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2), 1996.
- [2] J. Brevik, D. Nurmi, and R. Wolski. Predicting bounds on queuing delay in space-shared computing environments. In *IISWC*, pages 213–224, 2006.
- [3] P. Burns. Robustness of the ljung-box test and its rank equivalent. In *The Journal of Derivatives*, pages 7–18, 2002.
- [4] 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.
- [5] P. A. Dinda and D. R. O’Hallaron. Host load prediction using linear models. *Cluster Computing*, 3(4):265–280, 2000.
- [6] A. B. Downey. Using queue time predictions for processor allocation. In *Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, pages 35–57, 1997.
- [7] B. Efron. Bootstrap. another look at jackknife. *Ann. Statist.*, 7(1):1–26, 1979.
- [8] F. Gagliardi et. al. Building an Infrastructure for scientific Grid computing: status and goals of the EGEE project. *Philosophical Transactions of the Royal Society A*, 1833, 2005.
- [9] S. Androzzzi et al. Glue Schema Specification, V1.3. Technical report, 2008.
- [10] R. Gott III. Implications of the copernican principle for our future prospects. *Nature*, (363):315–319, 1993.
- [11] L. Ilijašić and L. Saitta. Characterization of a computational grid as a complex system. In *Procs. of workshop GMAC ’09, in association with 6th IEEE ICAC*, pages 9–18, 2009.
- [12] S. Jha, M. Parashar, and O. Rana. Investigating autonomic behaviours in grid-based computational science applications. In *GMAC’09: Grids Meet Autonomic Computing, workshop associated with the 6th IEEE ICAC*, pages 29–38, 2009.
- [13] B.-D. Lee and J. M. Schopf. Run-time prediction of parallel applications on shared environments. In *CLUSTER*, pages 487–491, 2003.
- [14] T-W Lee and Y. Yang. Bagging binary and quantile predictors for times series. *Journal of econometrics*, 135(1-2):465–497, 2006.
- [15] H. Li and M. Muskulus. Analysis and modeling of job arrivals in a production grid. *SIG-METRICS Perform. Eval. Rev.*, 34(4):59–70, 2007.
- [16] 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.
- [17] M. Macias, O. Rana, G. Smith, J. Guitart, and J. Torres. Maximising revenue in grid markets using an economically enhanced resource manager. 2008.
- [18] J. Meng, S. T. Chakradhar, and A. Raghunathan. Best-effort parallel execution framework for recognition and mining applications. In *IPDPS*, pages 1–12, 2009.

- [19] N. Mi, G. Casale, L. Cherkasova, and E. Smirni. Injecting realistic burstiness to a traditional client-server benchmark. In *ICAC '09: Proceedings of the 6th international conference on Autonomic computing*, pages 149–158, 2009.
- [20] A. Mutz, R. Wolski, and J. Brevik. Eliciting honest value information in a batch-queue environment. In *GRID '07: Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, pages 291–297, 2007.
- [21] F. Nadeem, M. M. Yousaf, R. Prodan, and T. Fahringer. Soft benchmarks-based application performance prediction using a minimum training set. In *International Conference on e-Science and Grid Computing*, page 71, 2006.
- [22] J. Perez, C. Germain-Renaud, B. Kégl, and C. Loomis. Utility-based reinforcement learning for reactive grids. In *The 5th IEEE ICAC Autonomic Computing*, 2008.
- [23] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. 1989.
- [24] W. Smith, V. E. Taylor, and I. T. Foster. Using run-time predictions to estimate queue wait times and improve scheduler performance. In *Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, pages 202–219, 1999.
- [25] O. Sonmez, N. Yigitbasi, A. Iosup, and D. Epema. Trace-based evaluation of job runtime and queue wait time predictions in grids. In *HPDC '09: Proceedings of the 18th ACM international symposium on High performance distributed computing*, 2009.
- [26] 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.
- [27] 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.
- [28] D. Vengerov. A reinforcement learning approach to dynamic resource allocation. *Eng. Appl. Artif. Intell.*, 20(3), 2007.
- [29] R. Wolski, N. T. Springer, and J. Hayes. Predicting the cpu availability of time-shared unix systems on the computational grid. *Cluster Computing*, 3(4):293–301, 2000.
- [30] 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.
- [31] X. Zhang, C. Furtlehner, J. Perez, C. Germain-Renaud, and M. Sebag. Toward autonomic grids: analyzing the job flow with affinity streaming. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 987–996, 2009.



Centre de recherche INRIA Saclay – Île-de-France
Parc Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 Orsay Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
<http://www.inria.fr>
ISSN 0249-6399