The Characteristics and Performance of Groups of Jobs in Grids

Alexandru Iosup, Mathieu Jan, Ozan Sonmez, and Dick Epema

Department of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, Delft, The Netherlands
Members of the CoreGRID European Virtual Institute on Grid Scheduling
{A.Iosup,M.Jan,O.O.Sonmez,D.H.J.Epema}@tudelft.nl

Abstract. Even though with few exceptions, grid workloads are dominated by single-node jobs, not all of these jobs are necessarily independent or unrelated. For instance, sets of jobs may be grouped because they are submitted by users in batches, e.g., to perform parameter sweeps. However, there is no reported data to confirm the presence and structure of these groupings, despite the large potential impact of such information. To address this lack of information, in this work we present a first investigation into the characteristics of groups of jobs present in grid workloads. First, we define three types of job groupings: batch, continued, and bursty submissions. Then, we analyze the characteristics of these groupings for three long-term traces from currently deployed grid environments. Notably, our results show that the various groupings are responsible for up to 96% of the total CPU time consumption. Finally, we present insights into the performance of real grids in dealing with grouped jobs.

1 Introduction

It has become evident that many currently deployed grids, such as Grid'5000 [1], the NorduGrid [2] and the GLOW[1], are running almost exclusively single-node jobs [3]. However, this does not preclude some jobs in such workloads to be in some way related. For instance, users may submit batches of jobs in order to perform parameter sweeps, or they may submit workflows that consist of sequential jobs with precedence constraints. As a consequence, single-node grid jobs may be logically grouped in some way, e.g., in batches of identical jobs [4]. However, the presence and the structure of such groups of jobs have yet to be analyzed, despite their potentially tremendous impact on the efficiency and the functionality of grid resource management solutions. To address this gap, in this work we first characterize various job groupings that occur in production and research grid environments. Then, we present an assessment of their impact on the performance of real grid systems.

[1] Grid Laboratory of Wisconsin, [Online] cs.wisc.edu/condor/glow/, February 2007.

A.-M. Kermarrec, L. Bougé, and T. Priol (Eds.): Euro-Par 2007, LNCS 4641, pp. 382–393, 2007.

Over the last decade, much work has focused on enabling parallel applications in grids [5,6]. In parallel, scientists have exploited grids for running completely independent jobs. Combining these two extremes, a third direction focuses on running sets of single-node jobs with convenient groupings, e.g., parameter sweeps [4]. With the performance of grid systems being highly dependent on the structure of their workloads [7,8,9,10], it is imperative to establish which types of job groupings exist in real grid workloads, to what extent they exist, and what is the impact of their structure on the performance of real grid systems. Therefore, in this work we answer the following set of questions:

- **What are the dependencies among the jobs submitted by a single user?** Do users submit single, independent jobs, or batches of jobs such as parameter sweeps? How can we characterize groups of related jobs?
- **What is the physical structure of such job groupings?** How many jobs appear in batches? And what are their characteristics, e.g., their duration and resource requirements?
- **What is the impact of the job groupings on the performance of grids?** What is the total processing time of grouped vs. non-grouped jobs? Do job groupings "average out" the waiting time, and achieve a lower slowdown, compared to their non-grouped counterparts?

2 Grid Environments

In order to answer our questions, we have obtained long-term traces from three large grid systems. Each trace records the jobs that have been submitted to the grid, and includes for each job at least the identity of the user submitting it, and the job's time information (the time of the job submission, job start, and job end). In Table 1, we summarize the content of the traces considered in this work: Grid'5000, NorduGrid, and GLOW. Our workload analysis covers almost two million jobs submitted over a period of three years by more than 800 users. Note that the data sources have been selected to represent different types of grids: research for Grid'5000 and production for NorduGrid and GLOW, with GLOW a much more dynamic system than NorduGrid.

The first testbed used in our work is Grid'5000 [1], an experimental grid platform consisting of 9 sites geographically distributed in France. Each site

Table 1. Summary of the content of the studied traces. The ⋆ sign marks restrictions due to data scarcity (see text). UJM and LRM/GRM are acronyms for User Job Manager and Local/Grid Resource Manager jobs log, respectively. GRP and USR represent the number of unique groups/VOs and of users in the workload, respectively.

Trace				Number of observed					Consumed
ID	System	Source	Period	Sites	CPUs	Jobs	GRP	USR	CPUTime
T-1	Grid'5000	LRM	05/'04-11/'06	15	∼2500	951K	10	473	651y
T-2	NorduGrid	GRM	05/'04-02/'06	∼75	∼2000	781K	106	387	2443y
T-3	GLOW	UJM	09/'06-01/'07	1⋆	∼1400	216K	1⋆	18	55y

comprises one or several clusters, for a total of 15 clusters inside Grid'5000. We have analyzed traces recorded by all batch schedulers handling Grid'5000 clusters (OAR [11]), from the beginning of the Grid'5000 project up to November 2006. Note that most clusters of Grid'5000 were made available during the first half of 2005.

The second testbed considered is NorduGrid [2], a large scale production grid. In NorduGrid, non-dedicated resources are connected using the Advanced Resource Connector (ARC) as Grid middleware [12]. Over 75 different clusters have been added over time to the infrastructure. We have obtained the ARC logs of NorduGrid for a period spanning from 2003 to 2006. In these logs, the information concerning the grid jobs is logged locally, then transferred to a central database voluntarily. The logging service can be considered fully operational only since mid-2004.

The third source of data is the Grid Laboratory of Wisconsin (GLOW), a campus-wide distributed computing environment that serves the computing needs of the University of Wisconsin-Madison's scientists. This Condor-based pool consists of over 1400 machines shared temporarily by their rightful owners [4]. We have obtained a trace comprising all the jobs submitted by one Virtual Organization (VO) in the Condor-based GLOW pool, in Madison, Wisconsin. The trace spans four months, from September 2006 to January 2007.

3 Groups of Jobs

In this section, we introduce definitions for the three types of groups of jobs that we distinguish in this paper.

Let W be the workload of a grid, which we consider as a set $\{J_i | i = 1, ..., |W|\}$ of jobs ordered according to increasing submission time, so $ST(J_i) < ST(J_j)$ if $i < j$, where $ST(\cdot)$ returns the submission time of a job. A *group of jobs* G is a subset of W, which we assume again to be ordered according to submission time. The *seed* J_G of a group G of jobs is the job in G with the earliest submission time. We define the *arrival time of a group* G of jobs as the submission time of its seed, $ST(J_G)$, and we define the *inter-arrival time between two groups* G, G' as the difference between their arrival times $IAT(G, G') = ST(J_{G'}) - ST(J_G)$. We also define the *duration of a group* G as the difference between $ST(J_G)$ and $LFT(G)$, where $LFT(\cdot)$ returns the finish time of the last job in G.

If the function $user(J)$ returns the identify of the user who has submitted job J, then we denote by $W_u = \{J_i | user(J_i) = u\}$ the *group of jobs submitted by the same user* u. A *batch submission with time parameter* Δ *of user* u is a maximal contiguous subsequence G of W_u such that for any two successive jobs J, J' in G, $ST(J') \leq ST(J) + \Delta$. Similarly, we define a *continued submission with time parameter* Δ *of user* u as a maximal contiguous subsequence G of W_u such for any two successive jobs J, J' in G, $ST(J) \leq FT(J') + \Delta$, where the function $FT(\cdot)$ returns the finish time of a job. Similarly, a *bursty submission with time parameter* Δ is a batch submission of all users jointly. We further define a *Parameter Sweep Application* (PSA) as *batch submission with time parameter* Δ *of user* u with the additional constraint that all jobs execute the same application.

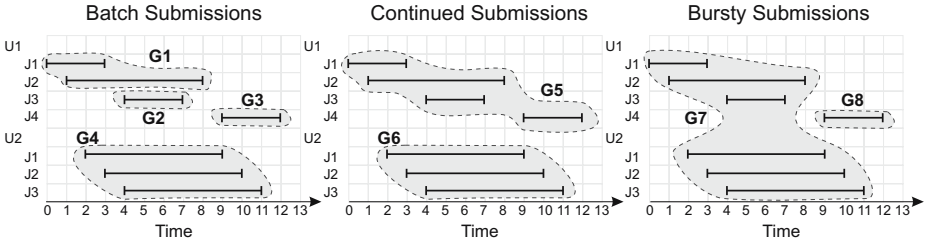

Fig. 1. The structure of a group of jobs: (left) batch submission, (center) continued submission, and (right) bursty submission

We denote by $\mathbf{G}_u^B(\Delta)$ the *ordered set of batch submissions with time parameter* Δ *of user* u containing all the batch submissions with time parameter Δ of user u, ordered according to their arrival times. Note that for any two successive G, G' in $\mathbf{G}_u^B(\Delta)$, $ST(J_{G'}) > LFT(G) + \Delta$. Similarly, we denote by $\mathbf{G}_u^C(\Delta)$ the *ordered set of continued submissions with time parameter* Δ *of user* u, and by $\mathbf{G}^S(\Delta)$ (S comes from spike) the *ordered set of bursty submissions with time parameter* Δ. From hereon, we call collectively the elements of $\mathbf{G}_u^B(\Delta)$ of any user u in the workload as batch submissions, or simply as *batches*.

We define as *non-batch*, *non-continued* and *non-bursty submissions* (collectively called *non-grouped submissions*) as the individual jobs from W that do not belong to any group of jobs in $\mathbf{G}_u^B(\Delta)$, $\mathbf{G}_u^C(\Delta)$, or $\mathbf{G}^S(\Delta)$.

We now define several performance metrics associated with the concept of job grouping. We define the *runtime of a group* (RT) as the amount of time during which at least one group job is running. We define the *duration of a group* G as $LFT(G) - ST(J_G)$. Then, we define the *idle time of a group* (IT) as the difference between the duration and the runtime of the group. We further define the *slowdown of a group* as the ratio between its duration and its runtime. Last, we define the *average group run time* (ART), the *average group idle time* (AIT), and the *average group slowdown* (ASD) as the average group runtime, group idle time, and group slowdown across all groups.

Figure 1 illustrates these definitions using the jobs submitted by two users, $U1$ and $U2$, over some period of time. According to our definition of batches, user $U1$ submits three batches ($G1$, $G2$ and $G3$), whereas user $U2$ submits only one batch ($G4$). Following the definition of continued submissions, user $U1$ submits only one group of jobs, $G5$. Finally, the definition of bursty submission divides the jobs into two groups $G7$ and $G8$, where $G7$ contains jobs from both $U1$ and $U2$.

4 The Characteristics of Jobs Groupings

In this section, we analyze the structure of batch, continued and bursty submissions. We first report an analysis of grouped submissions at the workload level. Then, we present the characteristics of grouped submission. Finally, we target the characteristics of jobs in grouped submissions. These results are clearly dependent on the value of Δ. Considering the overhead of existing grid middleware,

Table 2. Summary of the sizes of groups of jobs, for $\Delta = 120s$. GRP, USR, and CPUT represent the number of unique groups/VOs, the number of unique users, and the consumed CPU Time (in CPUyears) in the workload, respectively.

Trace ID	Batch Submissions					Continued Submissions				
	Sub.	Jobs	GRP	USR	CPUT	Sub.	Jobs	GRP	USR	CPUT
T-1	26k	808k	10	417	193y	14k	910k	10	417	462y
T-2	50k	738k	82	341	2192y	48k	738k	82	341	2192y
T-3	13k	205k	1	17	53y	13k	205k	1	17	53y

Trace ID	Bursty Submissions				
	Sub.	Jobs	GRP	USR	CPUT
T-1	6k	930k	9	163	581y
T-2	34k	759k	97	338	2325y
T-3	13k	204k	1	17	53y

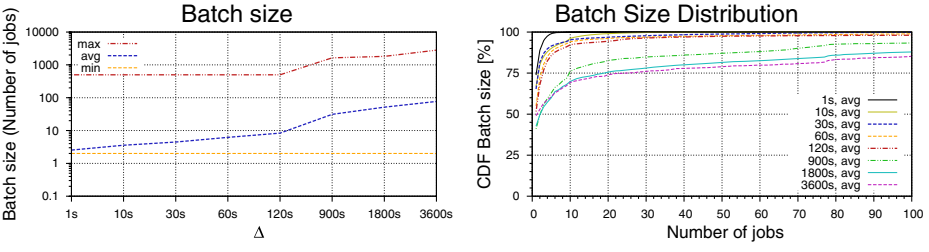

Fig. 2. The impact of parameter Δ on batch size, for trace T-3

which is commonly in the order of tens of seconds [13], we show throughout this section values obtained for $\Delta = 120s$, unless otherwise specified.

For brevity, we show graphical results only for batch and bursty submissions, as these are expected to form the majority of load arriving in the grid in a relatively short interval of time, and therefore potentially have the maximum impact on the grid's performance. We comment on the results obtained for continued submissions. We have also investigated the presence of PSAs; for T-1, 75% of batches are in fact PSAs. This result indicates that jobs are mostly submitted as bags-of-tasks, and not as workflows, with therefore waiting times mainly not being the consequences of dependencies and synchronizations between jobs. Furthermore, this result also shows that jobs, grouped according to our definitions (see Section 3), are principally related between them. Note that in trace T-1, the main differences between PSAs and batches are an increased group size by 9 in average, and a duration time divided by 5.7, but with an almost double coefficient of variation (1.7).

4.1 Workload-Level Analysis

We first investigate the grouped submissions at the workload level.

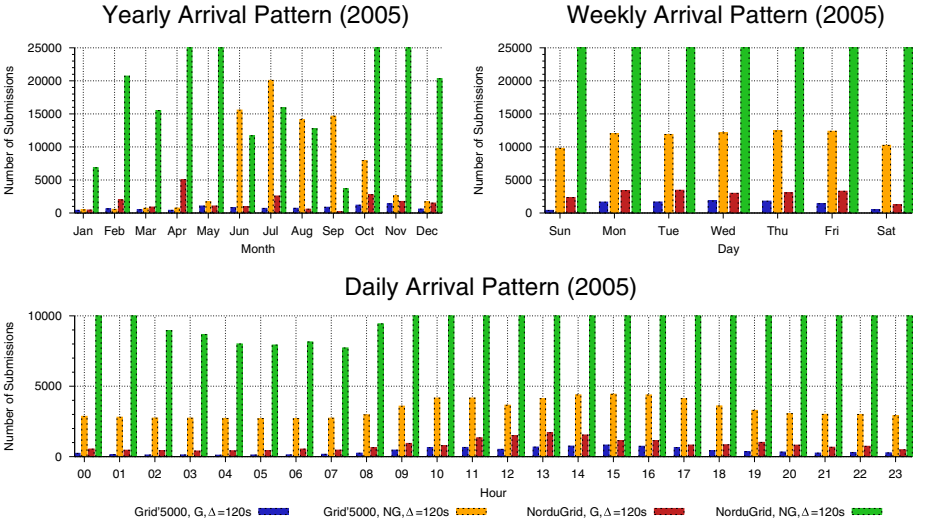

Fig. 3. The yearly, weekly, and daily arrival patterns of batch (caption **G**) and non-batch (caption **NG**) submissions, for all traces; $\Delta = 120s$

Figure 2 shows the impact of Δ on the batch size, for trace T-3. The curve displaying the maximum batch size has a breaking point around $\Delta = 120s$. We have obtained similar results for the other traces. Furthermore, the CDF of batch size for $\Delta = 10s$, $30s$ and $60s$ are almost identical. Therefore, we select $\Delta = 120s$ as the basis for the reported results in this section.

Table 2 shows the summary of the sizes of grouped jobs, for batch, continued, and bursty submissions. Relative to the complete traces, batch submissions are responsible for 85%–95% of the jobs, and for 30%–96% of the total consumed CPU time. Similar values for the number of jobs, and much higher values (from 70% upwards) for the total consumed CPU time can be observed for continued and bursty submissions. The columns *Sub.* and *Jobs* show the number of submissions and of jobs for each type of groups in the trace, respectively. We have further investigated the averages, standard deviation and the extremes for the size of the batches. For T-1, T-2 and T-3, the average size of the batches is 31±110, 15±33 and 15±38, respectively. The maximum size of a batch submission is, however, rather large: 1993, 2000 and 608 for T-1, T-2 and T-3, respectively. The average size for continued submissions are 64, 16 and 16 (for T-1, T-2 and T-3, respectively), and 162, 22 and 16 for bursty submissions (again for T-1, T-2 and T-3, respectively).

Figure 3 shows the yearly, weekly, and daily arrival patterns of batch grouped and non-batch submissions, for both T-1 and T-2 traces, and for 2005. Note that T-3 trace is not shown due to its short length (see Section 2). Note that a single batch submission includes several jobs. The yearly arrival pattern shows large variations, with 4-9 times more jobs submitted during peak months (e.g., October for both traces and April for T-2) vs. low-activity months (e.g., January for both traces). For trace T-1, the presence of peak months late in the yearly

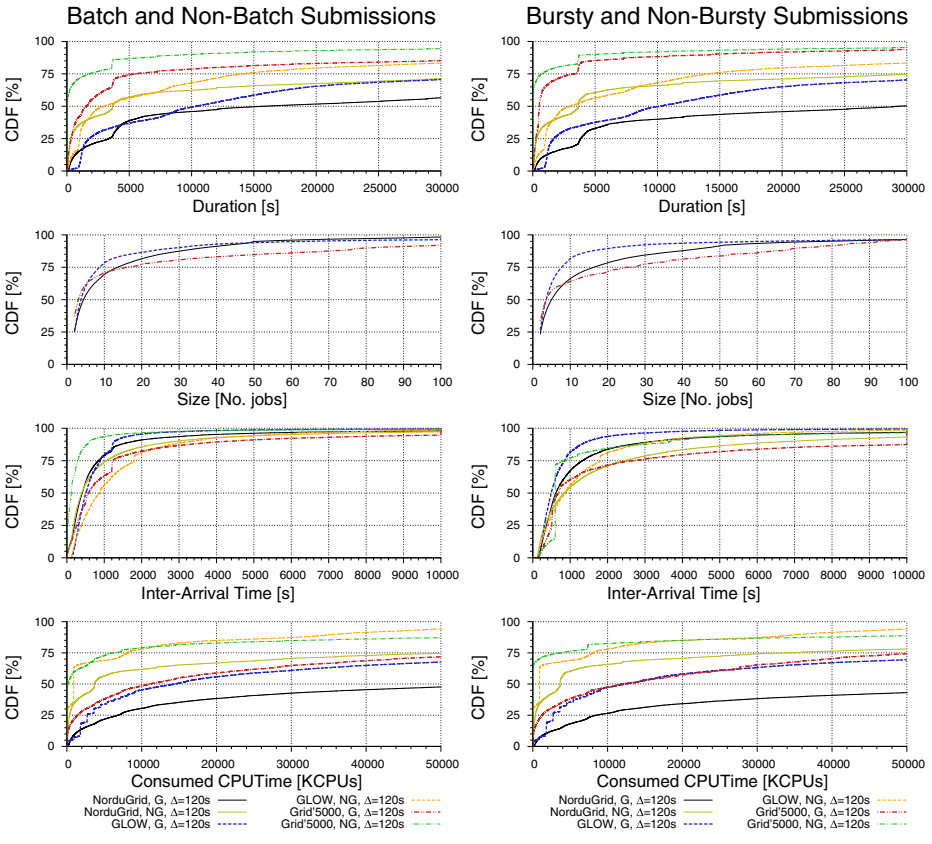

Fig. 4. The characteristics of the grouped (caption **G**) and of the non-grouped (caption **NG**) submissions, for all traces. Row 1: duration per submission. Row 2: the size of grouped submissions. Row 3: the Inter-Arrival Time distributions. Row 4: the consumed CPUTime.

pattern is explained by full availability of Grid'5000 after the first half of 2005 (see Section 2). The weekly pattern is less variable, with only 20-50% more jobs during high- vs. low-intensity days. The high- and low-intensity days correspond to weekday and weekend, respectively. There is a clear variation with daytime, with twice as many jobs submitted during peak vs. low hours. We have obtained similar results for the continued and bursty submissions.

4.2 Group-Level Analysis

We now analyze the characteristics of grouped submissions.

Figure 4 plots the cumulative distribution function (CDF) of the characteristics of the grouped and non-grouped batch and bursty submissions. The duration of batch submissions, shown in row 1, is statistically higher than that of their non-batch counterparts. For instance, for T-2 the average duration of a batch

submission is 1.5 days vs. 1 day for non-batch submissions. The average duration for batch submissions for traces T-1, T-2 and T-3 is 3.85, 1.5 and 0.34 days, respectively.

To complement the results presented in Table 2, we show in row 2 the sizes of batch and of bursty submissions; this metric is not shown in row 2 as it is useless for non-grouped submissions (the value is always 1 (job)!). Surprisingly, these sizes are relatively low: 75% of the batch submissions are size 15-20 (T-1 and T-2), or <10 (T-3); the results for bursty submissions are only slightly larger (except for T-2). This is an indication of a heavy-tail distribution of values: despite the majority of submissions being of low size, there are a few jobs that are oversized (outside the graphic, to the right).

As shown in row 3, the inter-arrival time between consecutive groups (see Section 3) is high: 50% of the values are between 400 and 700s, and 25% between 200 and 500s, for all traces. This is expected, given that the interarrival time between two groups represents the size of the group and, for the same user, a period of time with no submissions of at least $\Delta = 120s$.

Finally, the CPU time consumed by grouped jobs is much higher than that consumed by non-grouped jobs. Note that the CPU time provided by one node during one day of continuous work is 86.4 KCPUs (or roughly 100 KCPUs). We detail in Section 4.4 the average CPU time consumption per grouped and non-grouped jobs, for each trace.

4.3 Job-Level Analysis

Our third analysis part targets the characteristics of jobs in grouped submissions.

Figure 5 shows the characteristics of individual jobs, per submission type. Unexpectedly, there is no clear distinction between the consumed CPU time of batch and bursty submissions jobs, and that of their non-grouped counterparts, shown in row 1. For T-1, grouped jobs consume statistically less resources than their non-grouped counterparts, however note that grouped jobs consume negligible time (more than 90% are below 10 KCPUs). For T-2, grouped jobs consume more resources, and for T-3 they consume the same amount. The CDFs of CPU time for both grouped and non-grouped submissions converge on the high spectrum (over 100 KCPUs, or roughly 1.25 days). The average runtime of batch, continued and bursty submissions jobs (CDFs shown in row 3) are 0.66±6.65, 0.66±6.56 and 0.73±6.97 for T-1 respectively; 1.04±3.16, 1.04±3.16 and 1.04±3.18 days for T-2 respectively, and 2.27±5.59 hours in all cases for T-3. The average wait time (CDFs shown in row 2) of group jobs are higher than the runtime: between 30% to 45% for T-1 and 71% for T-3; with also increased standard deviations: from 6 for T-1 up to 16 for T-3, for durations expressed in hours. Note that trace T-2 does not provide any information about the waiting time of jobs.

We further answer the question: *do parallel jobs inside batches exist, or every grouping is conveniently parallel?* We define a *node* as a schedulable resource unit, e.g., processor for NorduGrid and Grid'5000, or virtual node in Condor. There exist parallel jobs in grouped submissions only for T-1 and T-2; T-3 is

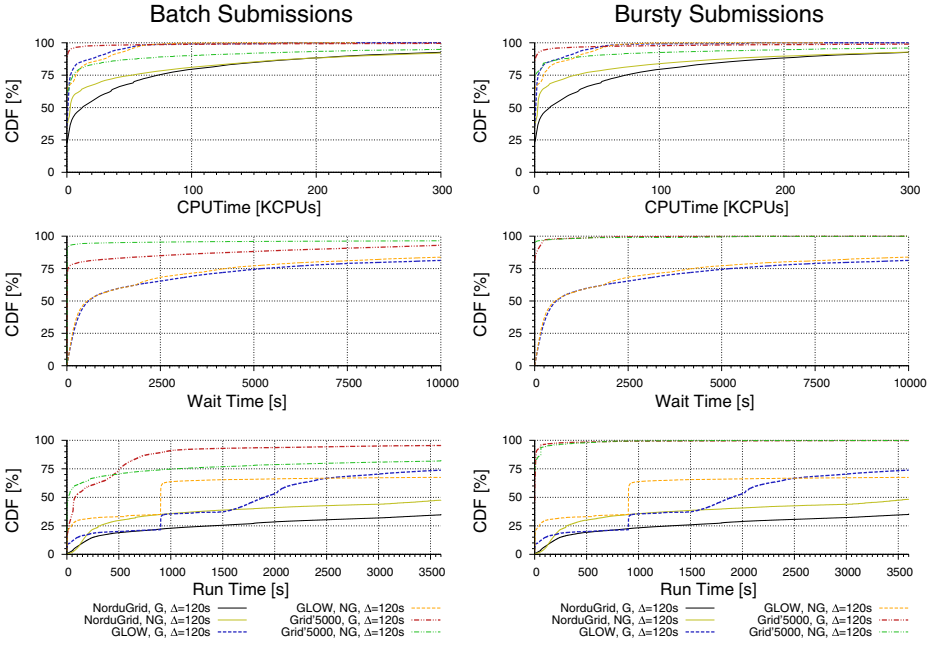

Fig. 5. The characteristics of individual jobs inside grouped (caption **G**) and non-grouped (caption **NG**) submissions, for all traces. Row 1: consumed CPU Time. Row 2: wait time. Row 3: run time.

composed entirely from single-node jobs. The average parallelism of all grouped submissions are 1±1, 2±7 and 1 for T-1, T-2 and T-3 respectively. Looking at the CDF function of the individual job sizes, 37% of the batch submissions in T-1 are of size 2, and 9% are of a size higher than 2, with a maximum of 325 nodes. The maximum size for T-2 is 64.

4.4 The Performance Impact of Grouped Submissions

We now turn our attention to investigate performance impact of grouped submissions, compared to that of non-grouped submissions. Expectedly, the average CPU time consumed by batch submissions for trace T-1 is much higher than their non-batch counterparts. The coefficient of variation is relatively high: 9.18. Other traces follow a same pattern, but are not shown for brevity.

Table 3. Summary of performance characteristics of the Grid'5000 for batches and non-batches

	Batch Submissions				Non-Batch Submissions			
Trace ID	CPU Time [KCPUs]	ART [s]	AIT [s]	ASD [s]	CPU Time [KCPUs]	ART [s]	AIT [s]	ASD [s]
T1	236	14181	568483	2156	101	4127	4233	280

By submitting enough jobs in the same batch, the waiting time of the individual jobs can be "averaged out", that is, at least one job in the batch is active at any given moment throughout the duration of the batch [14]. This leads to an ideal AIT of 0. Surprisingly, in the studied trace the batches display an high AIT value: over 4000% (**!**) of the ART. This situation also adversely impacts the ASD value: it is above 2000, and much higher than the ASD of non-batches. We would like to draw the attention of the research community to the problem of minimizing the AIT for batch submissions, by taking into account, to some extend, these groups of submissions in the design of scheduling policies for instance.

5 Related Work

In the past decades, a great deal of attention has been given to understanding and detailing the characteristics of workloads of large-scale multi-processor systems, and on the evaluation of their impact on existing and potential architectural solutions [7,8,9,10,15]. In particular, the seminal work of Feitelson et al. [7,8,10] lead to the comprehension of the workload modeling importance, and to a community-accepted model for rigid jobs in parallel environments [8,10]. However, there is relatively little work on characterizing grouped submissions in large-scale environments [9,14,15,16]. To the best of the authors' knowledge, this work is the first attempt to fully characterize grouped submissions in grids.

The results most closely related to our work are reported in [9], which characterize the size of batch submissions for $\Delta = 7200s$. However, our grid traces would include just a few batches at this interval, as most jobs would be grouped together. In [15], authors characterize applications that can be found in several scientific job batches. Still, they do not estimate the characteristics investigated in this work. Finally, the performance of running PSAs of genomics (i.e., BLAST) and of high-energy physics applications in grid environments have been explored in [14] and [16], respectively.

6 Conclusion and Ongoing Work

In this paper, we have presented a first investigation into the characteristics of groups of jobs present in grid workloads as well as insights into their impact on the performance of grids. First, we have formally defined three types of job groups: batch (and PSAs), continued and bursty submissions. Then, we have analyzed traces coming from one large research grid, namely Grid'5000, and two large production grids, NorduGrid and GLOW. Our analysis 1) shows the presence of arrival time patterns for groups of jobs and 2) reveals the characteristics of the groups and their composing jobs. Notably, we found that batch submissions account for up to 96% of the CPU time consumed in the analyzed traces. Finally, we have contrasted the performance obtained by grouped submissions with that of their non-grouped counterparts. Our results show that batch submissions have high idle times. This points to the need for new, more practical,

results in an important area of research: the minimization of the idle time for grouped jobs. We plan to investigate more thoroughly the impact of grouped submissions on grid performance, both with simulations and with experiments in our DAS grid environment in the Netherlands.

Acknowledgements

This research work is carried out in the context of the Virtual Laboratory for e-Science project (`www.vl-e.nl`), which is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W), and which is part of the ICT innovation program of the Dutch Ministry of Economic Affairs (EZ).

We gratefully acknowledge and thank Dr. Franck Cappello and the Grid'5000 team, Dr. Balász Kónya and the NorduGrid team, and Dr. Miron Livny, Dan Bradley, and the Condor team, for providing us with the grid traces used in this work.

References

1. Bolze, R., Cappello, F., Caron, E., Daydè, M., Desprez, F., Jeannot, E., Jègou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., Talbi, E.G., Irena, T.: Grid'5000: a large scale and highly reconfigurable experimental grid testbed. International Journal of High Performance Computing Applications 20(4), 481–494 (2006)
2. Eerola, P., Kónya, B., Smirnova, O., Ekelöf, T., Ellert, M., Hansen, J.R., Nielsen, J.L., Wäänänen, A., Konstantinov, A., Herrala, J., Tuisku, M., Myklebust, T., Ould-Saada, F., Vinter, B.: The nordugrid production grid infrastructure, status and plans. In: The 4th International Workshop on Grid Computing (Grid2003), Phoenix, AZ, USA (2003) 158–165
3. Iosup, A., Dumitrescu, C., Epema, D., Li, H., Wolters, L.: How are real grids used? the analysis of four grid traces and its implications. In: IEEE/ACM, pp. 262–269. IEEE Computer Society Press, Los Alamitos (2006)
4. Thain, D., Tannenbaum, T., Livny, M.: Distributed computing in practice: the Condor experience. Concurrency and Computation: Practice and Experience 17(2-4), 323–356 (2005)
5. Karonis, N.T., Toonen, B.R., Foster, I.T.: MPICH-G2: A grid-enabled implementation of the message passing interface. J. Parallel Distrib. Comput. 63(5), 551–563 (2003)
6. van Reeuwijk, K., van Niewpoort, R.V., Bal, H.E.: Developing Java Grid applications with Ibis. In: Cunha, J.C., Medeiros, P.D. (eds.) Euro-Par 2005. LNCS, vol. 3648, pp. 411–420. Springer, Heidelberg (2005)
7. Feitelson, D.G., Rudolph, L., Schwiegelshohn, U., Sevcik, K.C., Wong, P.: Theory and Practice in Parallel Job Scheduling. In: Feitelson, D.G., Rudolph, L. (eds.) Job Scheduling Strategies for Parallel Processing. LNCS, vol. 1291, pp. 1–34. Springer, Heidelberg (1997)
8. Feitelson, D.G.: Packing schemes for gang scheduling. In: Feitelson, D.G., Rudolph, L. (eds.) JSSPP. LNCS, vol. 1162, pp. 89–110. Springer, Heidelberg (1996)
9. Squillante, M., Yao, D., Zhang, L.: Analysis of job arrival patterns and parallel scheduling performance. Perform. Eval. 36-37(1-4), 137–163 (1999)

10. Lublin, U., Feitelson, D.G.: The workload on parallel supercomputers: modeling the characteristics of rigid jobs. Journal of Parallel and Distributed Computing 63(11), 1105–1122 (2003)
11. Costa, G.D., Georgiou, Y., Huard, G., Martin, C., Mouniè, G., Neyron, P., Richard, O.: A batch scheduler with high level components. In: Proceedings of the 5th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid '05), Cardiff, UK, pp. 776–783. IEEE Computer Society Press, Los Alamitos (2005)
12. Ellert, M., et al.: Advanced Resource Connector middleware for lightweight computational Grids. FGCS 23(2), 219–240 (2007)
13. Dumitrescu, C., Raicu, I., Ripeanu, M., Foster, I.: DiPerF: An automated distributed performance testing framework. In: Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID'04), Pittsburgh, PA, USA, pp. 289–296. IEEE Computer Society Press, Los Alamitos (2004)
14. Dumitrescu, C., Raicu, I., Foster, I.: Experiences in Running Workloads over Grid3. In: Zhuge, H., Fox, G.C. (eds.) GCC 2005. LNCS, vol. 3795, pp. 274–286. Springer, Heidelberg (2005)
15. Thain, D., Bent, J., Arpaci-Dusseau, A.C., Arpaci-Dusseau, R.H., Livny, M.: Pipeline and batch sharing in grid workloads. In: HPDC, pp. 152–161. IEEE Computer Society, Los Alamitos (2003)
16. Bonacorsi, D., et al.: Running CMS software on GRID Testbeds. Computing in High Energy and Nuclear Physics, 24-28, March 2003, La Jolla, CA, USA (2003)