UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Understanding and mastering dynamics in computing grids: processing moldable tasks with user-level overlay

Mościcki, J.T.

Publication date
2011

Link to publication

Citation for published version (APA):
Mościcki, J. T. (2011). *Understanding and mastering dynamics in computing grids: processing moldable tasks with user-level overlay*. [Thesis, fully internal, Universiteit van Amsterdam].

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:23 May 2025

CHAPTER 8

Conclusions and future work

> He looked up and said, "there's an infinite number of monkeys outside who want to talk to us about this script for Hamlet they've worked out."
>
> The Hitchhiker's Guide to the Galaxy by Douglas Adams

8.1 Grid dynamics and its consequences for task processing

The development of the EGEE Grid is a major achievement: the EGEE Grid successfully provides a large-scale computing service to numerous research communities. It is the largest grid in operation and it is representative for other grids as a complex and highly dynamic system.

The main structural reasons for the complecity of large grids are 1) large scale of the infrastructure and 2) decentralized organization and operation. As the result grids are highly heterogeneous and integrate computing sites of varying internal structures, computing capacities and sizes which span several orders of magnitude. In addition, grids are mainly composed of commodity elements which expose high failure rates. Individual elements of large grids are maintained, operated and evolved with limited central coordination, and often, without efficient communication and coordination with other stakeholders. Coordination is a non-trivial issue due to conflicting requirements and

priorities of different user groups: end users, local system administrators, middleware development groups and support teams.

Architecture and implementation of middleware services in the EGEE Grid contributes to end users' perception of unreliable and unpredictable system behavior. This is especially true for temporal dynamics related to the submission of jobs due to the early-binding processing model, and inherently inaccurate nature of data provided by information services for the use in workload management. As a result, heavily-tailed job queuing times contribute to low efficiency of task processing and a perception of lower Quality of Service.

A large number of active user communities contributes to observed spatial and temporal dynamics, which may be both short- and long-range. Heavy user communities, such as the LHC experiments, generate considerable and sustained load on large numbers of grid elements. Despite this substantial background, specific workload distribution structures may be distinguished. Temporary pockets of activity are visible in smaller clusters of grid elements and they correspond to smaller-scale analysis activities. Centralized workload distribution patterns in some VOs are visible as consisting of one central element processing a large number of jobs, and a number of smaller "satellite" elements processing less jobs.

In this study we focused on the temporal dynamics resulting in long tails of distributions of job queuing times. It has profound impact on the efficiency of task processing and may lead to complex distributions of makespan with non-trivial dependencies between their statistical properties such as mean and variance. These properties are effectively the measures of Quality of Service from end user's perspective.

8.2 Contributions of this work

This thesis is a contribution to the debate if Quality of Service in grids may be efficiently implemented at the application level. Here we summarize the results of our research for which detailed objectives were stated in section 1.5.

First, we described and analyzed the spatial and temporal dynamics in the world's largest grid in production, the EGEE Grid (Chapter 2). Then, we developed the task processing model for the late-binding scheduling method which explains *why* late binding significantly improves the distribution of makespan, albeit in statistical sense (Chapter 3). Using simple mathematical modeling we concluded that the late-binding method reduces the mean and standard deviation of makespan on a basis of the Central Limit Theorem (Sec. 3.4). We have shown that it is possible to achieve shorter and more predictable processing times above the raw grid processing services, which at their core, however, remain "best-effort". The processing efficiency may be further improved by combining the late-binding scheduling with adaptive workload-balancing and heuristic resource selection. We have developed a User-level Overlay to make such a combination easy and efficient (Chapter 4).

We demonstrated that adaptive workload balancing may give an additional advantage in specific cases, where the scheduling overheads effectively impose limits on task granularity. Unlike other approaches, fine tuning of the workload-balancing algorithm

capability	application
scaling out on demand	ITU RRC06
short-deadline computing	ITU RRC06
application-aware task prioritization	LatticeQCD
autonomous processing	LatticeQCD
heuristic resource selection	LatticeQCD
Grid/HPC interoperation	LatticeQCD
operational efficiency	AvianFlu
web portal integration	AvianFlu
workflow service integration	GATE
shell environment integration	HEP/GANGA
increased predictability of job completion times	Geant4
flexible application interfacing modes	Geant4

Table 8.1: Extended processing capabilities achieved with DIANE/GANGA User-level Overlay.

is done in the interaction with the execution environment. Automatically collected benchmarks allow to respond to possibly varying application and resource characteristics (Sec. 4.8). The resource selection uses simple but efficient heuristics to improve the duty cycle of an application, yet without a priori knowledge of the application itself (Sec. 4.7.3 and 7.6).

The User-level Overlay was developed as an application hosting environment to facilitate gridification of applications within existing boundary conditions for the largest production grids (Sec. 4.1). This strategy has proven efficient and successful in a number of computing activities in diverse application domains (Chapter 5). It allows for various additional capabilities to be achieved, as summarized in Tab. 8.1. It also allows to achieve application-level interoperability between different computing infrastructures, as summarized in Tab. 8.2.

The impact of our work was validated by several examples of successful use of the ideas and tools, which was sometimes undertaken by independent teams in diverse application areas. While presenting such examples we also point out specific characteristics and properties of the User-level Overlay, which may be used as inspiration or even guidelines for future applications.

A unique feature of our User-level Overlay is an efficient support of radically different use-cases:

- capability computing – where tens of thousands of very short tasks must be completed on a strict deadline (Chapter 6), and
- capacity computing – where cumulative resource usage reaches hundreds of CPU-years over several weeks of uninterrupted service (Chapter 7).

We have laid out the key elements of the software design and architecture which allow to achieve such a flexibility with a strict control of the efficiency trade-offs. They

resources	application
interactive clusters, batch farms, EGEE Grid	ATLAS, LHC*b*, Geant4
desktop-PC farm, EGEE Grid	ITU RRC06
HPC, EGEE, TeraGrid	LatticeQCD
PRAGMA, EGEE Grid	BLAST
Amazon EC2 clouds	Google SoC09

Table 8.2: Summary of cases of application-level interoperability between various distributed infrastructures achieved with DIANE/GANGA User-level Overlay.

include layering with asynchronous component interaction, lose component coupling via a clear separation of concerns and a correct choice of foundation technologies.

Programming which exploits common application patterns [101] in the context of loosely-coupled distributed systems is becoming increasingly important in scientific applications. New trends such as Many Task Computing attempt at bridging the gap between High Performance and High Throughput worlds. The User-level Overlay based on GANGA/DIANE which implements the late-binding task processing model allows to handle such applications with improved efficiency and reliability.

Summing up, in this way the central research hypothesis of this work defined in Sec. 1.5 has been demonstrated. We believe that our strategy represents a serious step forward bridging the gap between the system domain – infrastructure and middleware – and the application domain in large production grids.

8.3 Open issues

In the Introduction we defined the MTAs as a class of applications targeted by the User-level Overlay. Clearly, the Master/Worker coordination backbone which is well adapted for structured forms of parallelism (such as iterative applications, DAGs or workflows), is not ideal for irregular or speculative parallelism were worker processes must communicate in complex ways with one another. Additionally, routing of communication via a central point in a wide-area network may create bottlenecks for applications with very high communication to computation ratio. The User-level Overlay may not be an efficient method for traditional, MPI-style applications although we indicated applications which use the MPI for implementation but may be easily ported to the User-level Overlay.

The User-level Overlay approach is a design trade-off between usability for application providers, typically computing experts within application communities, and runtime efficiency. Our approach is placed mid-way between a generic scheduling service and an embedded scheduler tightly-coupled with the application code. Predefined, standard modules provided with the User-level Overlay allow a straightforward interfacing of legacy and black-box applications. However, applications with complex communication structures must be refactored and instrumented with the scheduling functionality. In spite of many differences, this may be compared to implicit parallelism provided by OpenMP versus explicit refactoring of application code for MPI. Some form of com-

piler support for User-level Overlay, for example similar to OpenMP directives, would be desirable to automate the porting process.

User-level Overlay is a powerful tool which may be abused by a careless user by creating large pools of idle worker agent jobs. This may happen for example by choosing inappropriate configuration parameters or using erroneous application plugins. The risk may be mitigated with off-the-shelf plugins developed and tested by power users and domain experts. Incorrect usage of grid resource is a general concern, however, because grids are open for easy abuse even by fairly unsophisticated means. This is because the fair share at a user-level is not currently implemented in the large production grids, and there are limited ways of controlling the resource usage for individual users.

Existing fair-share models developed for batch systems may be a basis for an additional study of how User-level Overlay affects fair share in grids. This could be beneficial in preparation for the future introduction of fair share in the EGEE Grid, which – will most likely be based on techniques known from batch systems. This could be achieved with a simulation of an environment where a group of "standard" users competes for resources with a group of User-level Overlay users. Such as study could also include a possible job "starvation" and its consequences, especially in situations where the number of available resources is much smaller than the number of user jobs. This could lead to the question if and how the worker agent lifetime should be limited and what is the impact of such a limit. The current implementation allows to define the worker agent time limit as a configuration parameter for individual runs. A strategy for coordination of worker time limits in a user community could require some form of distributed knowledge and decentralized decision process to be put in place.

We observed a practical problem for some users with running of the RunMaster service in their institutes. RunMaster must be instantiated in a host with inbound access from the outside networks. Enabling such access typically requires opening of the firewall by a local system administrator and is not an automatic procedure. This problem could be overcome by the development of a generic virtualization service for running RunMaster components remotely. This requires, however, to understand and manage the right balance between the security and flexibility.

8.4 Future work

In this thesis we focused on analyzing the impact of the job waiting time on task processing, deliberately simplifying other parameters of the model. In the future, for a higher precision of the model, more parameters need to be considered to reflect the distribution of processing speed of grid resources (heterogeneity), variable processing efficiency (time sharing of multi-code nodes) and the presence of resubmission and runtime errors.

Quantitative evaluation and simulation of G and U models for High Throughput application use-cases is also a possible extension of current work. In the High Throughput case, the total workload is so large that for K simultaneously submitted worker agent jobs the maximum allowed usage time of individual computing resources is exceeded, $\frac{W}{K} \gg \nu_{max}$.

Further work is required to quantitatively study the interaction of late-binding

method with fair-share scheduling and mutual exclusion of users competing for the same, scarcely available resources.

Characterizing the late-binding task processing method from the queuing theory perspective is another interesting area of future investigations. User-level Overlay effectively creates a single queue dedicated to one user above a set of distributed queues in the Grid which are shared between many users.

Finally, we want to explore the exponential property of τ distribution and provide more detailed and deeper analysis of the Grid using methods elaborated in the complex systems research area.

8.5 Postscriptum

Stepping away and grasping a macroscopic picture of a large grid, an analogy to a large river immanently appears. The main current – global infrastructures, middleware and virtual organizations – is powerful but slow and steady. At the same time, on the surface fast local currents flow in various directions – local resource providers and user communities – creating whirlpools and backflows. Some travelers – capability computing users – try to cross from one bank to another in a shortest time possible. Some others – capacity computing users – need to cross the river many times and must be sure that this process is trustworthy and easy to foretell. However, the dynamics on the surface are unpredictable and chaotic, so some traveler communities may build dams or bridges – specialized VO services – which then change local dynamics for other travelers. We may see that the system is dynamic but the large-scale landscape is steady. The river lazily swells. But then a new paradigm-shift comes to turn the river around and make it flow in another direction![1].

We hope that this thesis provides new insights on understanding and improving fundamental properties of large grids. We also hope that it will be possible to carry them through and to apply, some part of them, equally efficiently in the future generation computing systems, after a future, inevitable paradigm shift.

[1] *F. Golden, E. Amfitheatrof – Making Rivers Run Backward; TIME June, 14 1982*