Adaptive Performance Modeling on Hierarchical Grid Computing Environments

Wahid Nasri[1], Luiz Angelo Steffenel[2] and Denis Trystram[3]

[1]ESSTT, Département d'Informatique, Tunis, Tunisia
Wahid.Nasri@ensi.rnu.tn
[2]Université Nancy-2, LORIA, AlGorille Team, Nancy, France
Luiz-Angelo.Steffenel@univ-nancy2.fr
[3]Laboratoire ID-IMAG, INPG, Grenoble, France
Denis.Trystram@imag.fr

Abstract

In the past, efficient parallel algorithms have always been developed specifically for the successive generations of parallel systems (vector machines, shared-memory machines, distributed-memory machines, etc.). Today, due to many reasons, such as the inherent heterogeneity, the diversity, and the continuous evolution of the existing parallel execution supports, it is very hard to solve efficiently a target problem by using a single algorithm or to write portable programs that perform well on any computational supports. Toward this goal, we propose a generic framework based on communication models and adaptive approaches in order to adaptively model performances on grid computing environments. We apply this methodology on collective communication operations and show, by achieving experiments on a real platform, that the framework provides significant performances while determining the best combination model-algorithm depending on the problem and architecture parameters.

Keywords. *Cluster computing; Performance modeling; Adaptive techniques; Polymodels of communications; Collective communication operations*

1. Introduction

1.1. Recent Challenges of Efficient Parallel Algorithms

In the last years, there was a huge development in the field of parallel and distributed processing, especially at the architectural level leading to a wide variety of execution supports. The major innovation was the phenomenal spread of architectures like clusters and grids. These platforms represent a reasonable alternative to traditional parallel machines and have become the most cost-effective computing supports for solving a large range of high performance computing applications due the good cost/performance ratio that they provide. However, the introduction of such computational systems has a major impact on the design of efficient parallel algorithms. Moreover, such parallel systems are often upgraded with new generation of processors and network technologies. This fact will change the balance between computations and communications.

Today, due to the increasing diversity of existing parallel systems consisting on collections of heterogeneous machines, it is very difficult and mostly impossible - for a user to choose an adequate algorithm because the execution supports are continuously evolving. One version will be well-suited for a parallel configuration and not for another one. This portability issue becomes crucial because of the frequent changes of the components of the systems. Moreover, the inherent heterogeneity and the diversity of networks of such environments represent a great challenge to model and optimize communications for high performance computing applications. These deferent elements require to revise the classical parallel algorithms which consider only regular architectures with static configurations and to develop more powerful approaches to benefit well from the capacities of these new execution supports.

1.2. Contribution of this Work

Our objective within this work is to propose a generic framework based on communication models and adaptive techniques for dealing with prediction and modeling communications, and integrating scalability and portability issues. More precisely, the contribution of this paper is to propose a methodology with two level adaptation. Indeed, at the first level we proceed, given a target architecture, by determining the more appropriate model from a set of selected ones to better predicting performances. We will finally obtain a Poly-model of communication, as described

later in section 3.2. Next, we will have to determine the best algorithm among multiple algorithmic options for resolving a given problem.

1.3. Related Works

Our framework differs from the other works in a significant aspect. Indeed, all existing adaptive approaches presented in the literature like those of [4, 17, 15, 9] proceed by selecting, using different techniques, one algorithm or eventually combining from a library containing multiple algorithmic choices. In [14], the authors conduct an extensive experiment to determine the model that better represents the execution time of collective communication operations. Indeed, that work provides meaningful insights on the accuracy levels obtained with different performance models, and the importance to avoid relying on a single model. Therefore, to the best of our knowledge, our framework provides the first general methodology for automatically associate the more appropriate communication model to the best algorithm among multiple model and algorithmic options. It determines the best combination model-algorithm and computes an efficient execution scheme that minimizes the overall execution time of a parallel application.

1.4. Organization of the Paper

The remainder of the paper is organized as follows. We begin, in section 2, by presenting the architectural model of the target parallel and distributed system and the performance evaluation models of a parallel algorithm. In section 3, we first define the concept of poly-model, present our adaptive framework for poly-models of communication, and detail its components. Section 4 is devoted to a case study where we apply our adaptive framework on collective communication operations. We present indeed numerical simulations and practical experiments performed on a heterogeneous hierarchical grid proving the interest of this work. Finally, section 5 concludes the paper and discusses some perspectives to extend this work.

2. Preliminaries

2.1. Description of the Architectural Model

We assume in this work a generic model of a platform composed by heterogeneous hierarchical clusters as described in [5]. The platform studied enjoys heterogeneity along three orthogonal axes:

1. The processors that populate the clusters may differ in computational powers, even within the same cluster.
2. The clusters comprising the platform are organized hierarchically and are interconnected via a hierarchy of networks of possibly differing latencies, bandwidths and speeds. At the level of physical clusters, the interconnection networks are assumed to be heterogeneous.
3. The clusters at each level of the hierarchy may differ in sizes.

2.2. Adaptive Approaches

The adaptive approaches are a promising answer to the challenges presented previously in section 1.1. The idea is to adapt algorithms together with their execution to the target architecture. They propose adaptive algorithms for solving the same problem which have different behavior and performances and to derive the best one for a target parallel system. These algorithms may be automatically adapted to the execution context (data and support). In a parallel context, the adaptive algorithm should be scalable. Indeed, it should maintain a given efficiency when the size of the problem and the number of processors grow. Other approaches are possible for adapting the algorithms to new supports. For instance, adequate software can be developed in the middleware. We are interested in this work in the adaptability at the algorithmic level.

2.3. Communication Models

Choosing the right algorithm for solving a parallel application requires being able to predict the performances. There are many parallel communication models in the literature that analyze performances based on system parameters [10, 1, 7, 6, 8, 11, 13, 16]. More sophisticated models have been proposed for complex architectures [5, 15]. All these models differ on the assumptions about the computational support parameters, such as latency, heterogeneity, network contention, etc., and therefore are able to cover a great variety of architectures and modeling aspects. For instance, the selection between these models will depend on the data size to communicate, the models accuracy and their relative cost (parameters acquisition and models complexity). We present and summarize in Table 1 a comparison between some of these models.

3. Description of the Adaptive Framework for Performance Modeling

3.1. Overview of the Methodology

In this section, we describe our framework for adaptively modeling communications in an execution environ-

Table 1. Communication models for performance prediction

	Hockney	LogP	LogGP	LoPC	LoGPC	pLogP	Model of [5]	Model of [15]
Heterogeneous environment	No	No	No	No	No	Yes	Yes	Yes
Hierarchical	No	No	No	No	No	Yes	Yes	Yes
Latency	α	L	L	L	L	L	λ	L
Gap	$1/\beta$	g	g+G	g	g+G	g	included	g
Sender overhead		o	o	o	o	o_s	π	o_s
Receiver overhead		o	o	o	o	o_r	π	o_r
Contention modeling	No	No	No	Yes	Yes	No	No	Yes*

* depending on the communication pattern

ment characterized by its heterogeneity and its and its hierarchical organization. An overview of the methodology is sketched in Figure 1. The processing is separated in two successive phases. During the first one, we aim to partition the target execution platform to form subnets of similar characteristics by automatically discover the network topology. Then, when executing the second phase, we have to determine for each subnet (i.e. cluster) the more appropriate communication model to better predicting performances. We will finally obtain a Poly-model of communication. In the sequel, we more detail the major components of the framework.

3.2. Concept of Poly-Model

Generally, in order to predict and model the overhead due to communications in a parallel application, we use a single communication model, like, for instance, Hockney [10], LogP [7], pLogP [11], etc. Due to the wide variety of existing parallel machines, which requires determining multiple parameters to get more precise results, a sophisticated model is necessary. On the other hand, using such a model on "simple" architectures is not useful due to the cost of determining their parameters. For that reasons, we introduce the concept of Poly-model which determines adaptively the more appropriate model for a target platform according to the problem and architecture parameters. In fact, the poly-model that we propose is equivalent to a combination of multiple models, to be used on different clusters. It chooses an adequate model in terms of several information, including the size of data to communicate, the type of interconnection network (with contention or not), the number of nodes, etc. Let us remark that the generated poly-model which uses adaptive techniques will:

1. Better model the communications in terms of the characteristics of the hardware resources of the target parallel system.
2. Reduces the cost of modeling parallel applications on complex architectures.
3. Provides precise prediction results.

We finish this description by mentioning that the complexity of the overall process of the scheme adaptation does not affect the global cost of the application to be modeled.

3.3. Platform Partitioning

Since the target parallel system may be heterogeneous at many levels (computing powers, interconnection network performances, etc), it is very difficult to manage such platform towards a high performance computing. One way to answer this problem and to minimize the inherent heterogeneity is to subdivide the network in homogeneous subnets (or logical clusters), as described below. At the end of this phase, we will get a set of logical clusters of homogeneous nodes and accurate interconnection network, which will be used to adaptively modeling communications inside each cluster during the second phase of the framework.

Network Performance Measurements
The framework starts by collecting available information from the target execution environment to be used in the step of clustering (see next section). There exist many tools for network monitoring, such as NWS (Network Weather Service) [18]. These tools permit to determine many useful parameters of the target parallel system like the current network status, the communication latency, the speeds of the processors, the CPU load, the available memory, etc. For instance, the communication latency and throughput permit to identify groups of machines with similar communication parameters.

Clustering
One reason to construct logical clusters is that machines may behave differently, and the easiest way to optimize communications is to group machines with similar performances [2, 15]. In order to classify nodes in logical clusters, we can use a clustering algorithm similar to the one presented in [12]. This algorithm analyzes each interconnection on the distance matrix containing the latencies between links in order to group nodes for which their incident edges

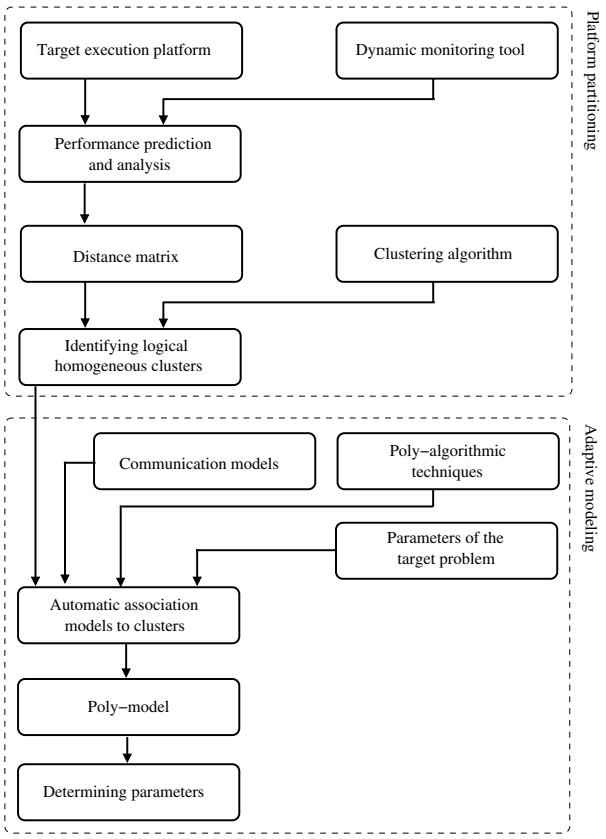


Figure 1. Adaptive framework.

respect a latency bound (by default 20%) inside that subnet. Note that the distance matrix was obtained when applying NWS on the clusters to determine the network information.

3.4. Adaptive Modeling

Once the platform is partitioned in separated homogeneous hierarchical clusters, we determine, using an adaptive approach, an adequate model from the set of selected models for each cluster. Indeed, we modeled and implemented several algorithms from the literature, which perform differently according to the network environment. By selecting the best adapted algorithm to each different cluster in our grid, we contribute to a poly-algorithmic modeling of communications in a grid environment. Any necessary characteristics are measured during the first phase corresponding to the network partitioning. We recall that the algorithm selection is made in terms of information which is interesting to the problem, such as the size of data to communicate, the type of interconnection network, the number of nodes, etc.

It is also important to take into account the cost involved on the acquisition of models parameters. For instance, both LogP and LogGP models rely on a reduced number of measurements (they extrapolate the cost per byte from a few measurements), while pLogP requires several measurements to cover a large range of message sizes. Hence, while the later model is most expensive, it can be more precise when the communication cost does not varies linearly with the message size (a typical case with MPI, whose transmission policies depend on the message size).

For instance, we are able to define a communication schedule that minimizes the overall execution time through the analysis of the inter-cluster communication performance and the intra-cluster performance prediction. Once again we implement different schedule policies, which are selected according to their estimated termination time. The framework allows, indeed, implementing scheduling heuristics that perform on different communication levels according to the target communication pattern.

4. Application on Collective Communications

We apply our adaptive framework on collective communication operations by determining the best combination model-algorithm depending on the problem and architecture parameters. We refer here by algorithm a possible method to resolve the operation, as for example Pipeline, Binomial, Binary and Linear for Broadcast. At this level, our framework automatically associates the more appropriate model to the best algorithm among multiple model and algorithmic options.

Collective communication operations encompass a wide range of possible algorithms. The optimal implementation of such operations for a given parallel system depends on many factors, including for example, physical topology of the system, number of processes involved, message sizes, and the location of the root node [14, 15]. Figure 2 shows how to implement a collective communication operation using our framework. Indeed, the adaptive algorithm selection is based on two types of models: the analytical and experimental ones. Experimental techniques use information derived from previous operation executions to optimize processing for future problem instances, a service similar to an optimization cache. The analytical models will be useful to validate actual versus predicted performances.

4.1. Description of the Execution Platform

We have considered a Grid platform from to the project Grid'5000[1] to achieve our experiments. The architecture is initially composed of four clusters, distributed over sites in France: GRENOBLE, ORSAY, SOPHIA-ANTIPOLIS and TOULOUSE. This platform will first be partitioned into logical homogeneous clusters. The latencies intra and inter-clusters are presented in Table 2.

[1] *http://www.GRID5000.fr*

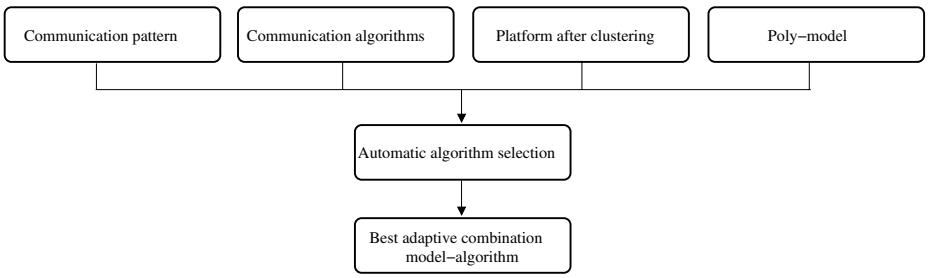


Figure 2. Execution of a collective communication operation using the framework.

Table 2. Latency (microseconds) intra and inter-clusters.

	C1	C21	C22	C23	C3	C4
	20 x Orsay	11 x Grenoble	7 x Grenoble	1 x Grenoble	20 x Toulouse	19 x Sophia
C1	48.39	6577.49	6586.49	6592.51	5211.94	8602.73
C21	6577.49	35.52	59.96	59.96	5387.48	2736.56
C22	6586.49	59.96	60.08	79.51	5393.98	2740.26
C23	6592.51	59.96	79.51	0*	5405.78	2745.98
C3	5211.94	5387.48	5393.98	5405.78	26.94	3630.51
C4	8602.73	2736.56	2740.26	2745.98	3630.51	35.04

* this "logical cluster" has only one machine.

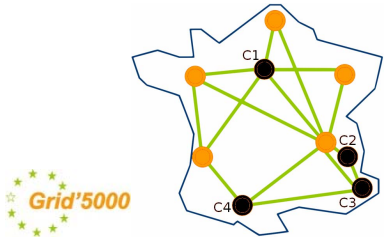


Figure 3. Grid'5000 sites.

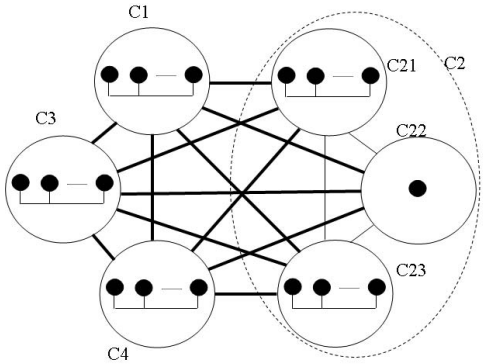


Figure 4. Platform after partitioning.

4.2. Network Partitioning

We applied our approach on a target parallel system composed of four clusters - C1 (20 machines ORSAY), C2 (19 machines GRENOBLE), C3 (19 machines SOPHIA-ANTIPOLIS) and C4 (20 machines TOULOUSE) with two levels of hierarchy distributed over four sites in France. Figure 3 shows the organization of the clusters.

The first phase of the framework leads to a new organization of the machines (see Figure 4). The cluster C2 will be partitioned into three sub-clusters, C21 (11 machines), C22 (1 machine) and C23 (7 machines), according to the latencies of the links between machines. Once the clustering phase is done, we have six logical clusters with homogeneous resources.

4.3. Model Selection

We have considered the Broadcast operation. With Broadcast, a single process, called *root,* sends the same message of size m to all other $(P-1)$ processes. Classical implementations of the Broadcast operation rely on fixed shapes such as Linear (Flat Tree) for small number of nodes and Binomial Trees for $P > 3$.

In our work we developed the communication models for some current techniques and their "flavors", which are presented on Table 3 in terms of pLogP parameters (which can be easily adapted to other models such as LogP and LogGP). Hence, we chose to compare in this paper four of the most known techniques, namely the Linear, the Binary, the Binomial and the Pipeline Broadcasts. Indeed, we perform the MPI_Bcast optimization in two hierarchical levels, as described below:

Table 3. Communication models for the *Broadcast* operation

Strategy	Communication Model
Linear (Flat Tree)	$L + (P-1) \times g(m)$
Segmented Chain (Pipeline)	$(P-1) \times (g(s)+L) + (g(s) \times (k-1))$
Binary Tree	$\leq \lceil log_2 P \rceil \times (2 \times g(m) + L)$
Binomial Tree	$\lceil log_2 P \rceil \times L + \lfloor log_2 P \rfloor \times g(m)$

First hierarchical level:
First, we deal with local-area communications, where a cluster "coordinator" will be charged to broadcast locally the message to all processes composing the cluster. Later, we integrate these clusters through the generation of efficient inter-cluster communication schedules.

Therefore, at the first hierarchical level, we proceed by selecting the most appropriate performance model and broadcast algorithm for each cluster according to two steps:

Poly-adapt-1: For a given problem (message size and number of processes) and for each implementation algorithm, we select the most accurate performance model such that its predictions correspond to our base of experimentations;

Poly-adapt-2: From the performance models selected in the previous step we select the most efficient implementation algorithm (i.e., the algorithm that terminates earlier).

For instance, we select the algorithm that minimizes the completion time of the operation, as summarized in Table 4, using the most accurate performance model for each problem instance (as we can observe in Figure 5).

Second hierarchical level:
Once ***Poly-adapt-2*** selected the best algorithm for each cluster, we must determine an efficient inter-cluster communication scheduling. Using inter-cluster communication parameters, we can construct an optimized broadcast tree between clusters using scheduling heuristics, an approach that provides better performances on grid environments than traditional grid-unaware algorithms found on most MPI distributions [3].

Indeed, in this example we rely on the *Early Completion Edge First* - ECEF heuristic, proposed by [4]. This heuristic proceeds by selecting a pair *sender-receiver* where the sender is available and the choice of the sender-receiver pair depends on the earliest possible moment when this transmission may effectively be finished. Therefore, we use a *Ready Time* (RT_i) parameter, evaluated conjointly with the transmission time between the processes, such that the pair i, j minimizes the following expression:

$$t = RT_i + g_{i,j}(m) + L_{i,j}$$

We can eventually integrate the communication time inside each cluster, as stated by the ECEF-LAt heuristic [3], where the communication schedule minimizes the overall broadcast time by taking into account both inter and intra-cluster communication times

4.4. Performance Analysis

As stated above, our politic of performance evaluation consists in implementing the broadcast when applying a one level of adaptation on the poly-model, i.e. choosing the more appropriate model with a fixed algorithm (*Poly-adapt-1*), and then a second level of adaptation, i.e. both model and algorithm are adaptive (*Poly-adapt-2*). Therefore, Figure 5 represents the completion time of a broadcast executed with different algorithms on the ORSAY clusters with different message sizes. We denote by **Poly1** the first adaptation and **Poly2** our full adaptation. The curves show that **Poly1** adapts to the performance model that closely predicts the performance of real experiences kept in our experiments base. Similarly, **Poly2** represents the expected performance of the best algorithm, proving the interest of the two-level adaptation. Indeed, this two-level adaptation mechanism is applied at each cluster, giving therefore the fastest implementation strategy for each environment.

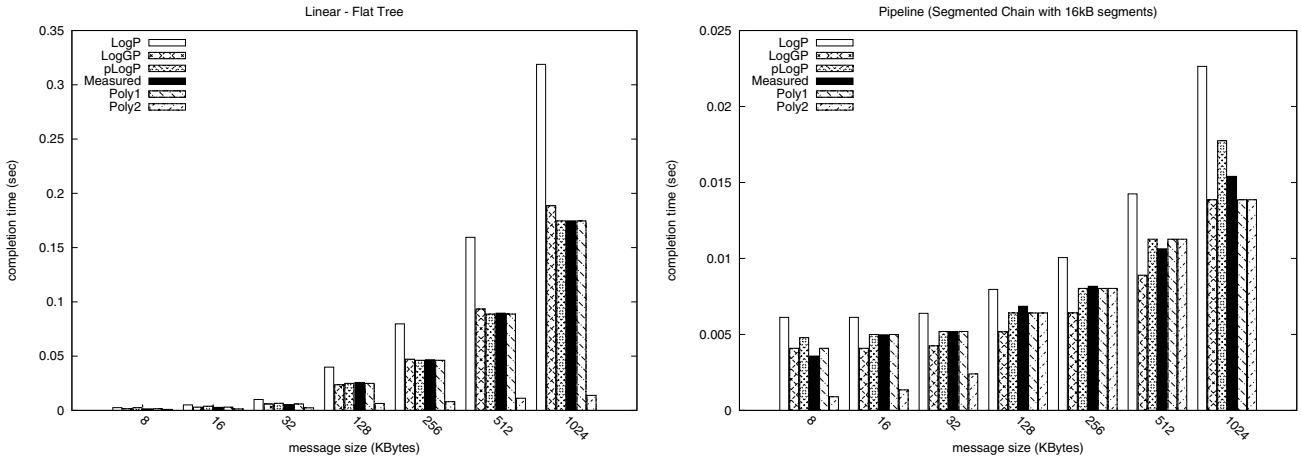
As a result, Figure 6 depicts the completion time of a broadcast executed on two layers with **Poly2** and the ECEF heuristic on the global platform composed by many clusters of different characteristics and network hierarchies. We observe that the optimized broadcast operation easily outperforms the traditional "grid-unaware" binomial broadcast implemented on most MPI distributions, and giving this example, we proceed at least twice as fast as the binomial tree. Summing up, the obtained results confirm the remarks mentioned previously, concerning the power and the benefits of a multi-level adaptive approach.

5. Concluding Remarks and Future Works

We have presented in this paper a new adaptive framework for dealing with performance modeling on grid computing platforms. The proposed methodology, based on communication models and adaptive approaches, proceeds in two level adaptation to automatically associate the more appropriate model to the best algorithm among multiple model and algorithmic options for each part of a large parallel execution support being used. The best identified combination model-algorithm and the determined execution scheme permit to minimize the overall execution time of a

Table 4. Summarize of the adaptive execution scheme for m=8kB and m=512kB.

Cluster		C1	C21	C22	C23	C3	C4	inter-clusters
m=8k	Selected model	LogP	LogP	-	LogP	LogP	LogP	Hierarchical [3]
	Selected algorithm	binomial	binomial	linear	binomial	binomial	binomial	ECEF
m=512k	Selected model	LogGP	pLogP	-	pLogP	pLogP	pLogP	Hierarchical [3]
	Selected algorithm	pipeline	pipeline	linear	pipeline	pipeline	pipeline	ECEF



(a) Linear

(b) Pipeline

(c) Binary

(d) Binomial

Figure 5. Completion time of Broadcast in the cluster ORSAY: (a) Linear, (B) Pipeline, (c) Binary and (d) Binomial

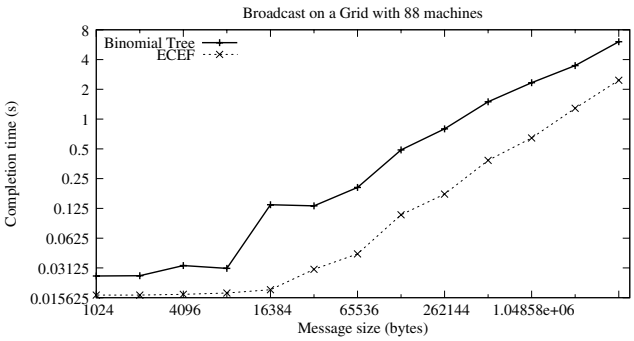


Figure 6. Completion time of the overall execution of a Broadcast on the global hierarchy platform.

target problem. This approach was applied on an important collective communication pattern, the broadcast operation, proving the interest of a multilevel adaptive approach and the worthy of this work.

As future prospects, we intend to perform experiments on other collective communication operations, and apply the framework on other types of target problems. We also plan to integrate other existing adaptive approaches to our framework to benefit well from the powerful of these techniques. Further, we intend to evaluate the impact of our framework on the performance of real scientific applications which perform intensive communication among the nodes.

References

[1] A. Alexandrov, M. Ionescu, K. Schauser, and C. Scheiman. LogGP: Incorporating long messages into the LogP model - one step closer towards a realistic model for parallel computation. In *Proceedings of the 7th Annual Symposium on Parallel Algorithms and Architecture (SPAA'95)*, 1995.

[2] L. Barchet-Steffenel and G. Mounie. Identifying logical homogeneous clusters for efficient wide-area communication. In *Proceedings of the Euro PVM/MPI 2004*, LNCS Vol. 3241, pages 319–326, Budapest, Hungary, 2004.

[3] L. A. Barchet-Steffenel and G. Mounie. Scheduling heuristics for efficient broadcast operations on grid environments. In *Proceedings of the Performance Modeling, Evaluation and Optimization of Parallel and Distributed Systems Workshop - PMEO'06 (associated to IPDPS'06)*, Rhodes Island, Greece, April 2006. IEEE Computer Society.

[4] P. Bhat, V. Prasanna, and C. Raghavendra. Adaptive communication algorithms for distributed heterogeneous systems. In *Proceedings of the IEEE International Symposium on High Performance Distributed Computing (HPDC 1998)*, 1998.

[5] F. Capello, P. Fraigniaud, B. Mans, and A. Rosenberg. An algorithmic model for heterogeneous hyper-clusters: Rationale and experience. *International Journal of Foundations of Computer Science*, 16(2):195–215, 2005.

[6] M. Clement, M. Steed, and P. Crandall. Network performance modelling for PM clusters. In *Proceedings of Supercomputing*, 1996.

[7] D. Culler, R. Karp, D. Patterson, A. Sahay, K. E. Schauser, E. Santos, R. Subramonian, and T. von Eicken. LogP - a practical model of parallel computing. *Communication of the ACM*, 39(11):78–85, 1996.

[8] M. I. Frank, M. Agarwal, and M. K. Vernon. LoPC: Modeling contention in parallel algorithms. In *Proceedings of 6th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pages 276–287, Las Vegas, Nevada, June 1997.

[9] O. Hartmann, M. Kuhnemann, T. Rauber, and G. Runger. Adaptive selection of communication methods to optimize collective mpi operations. In *Proceedings of the 12th Workshop on Compilers for Parallel Computers (CPC'06)*, La Coruna, Spain, 2006.

[10] R. Hockney. The communication challenge for MPP: Intel paragon and meiko cs-2. *Parallel Computing*, 20:389–398, 1994.

[11] T. Kielmann, H. Bal, S. Gorlatch, K. Verstoep, and R. Hofman. Network performance-aware collective communication for clustered wide area systems. *Parallel Computing*, 27(11):1431–1456, 2001.

[12] B. Lowekamp and A. Beguelin. ECO: Efficient collective operations for communication on heterogeneous networks. In *Proceedings of the 10th International Parallel Processing Symposium*, pages 399–405, 1996.

[13] C. A. Moritz and M. I. Frank. LoGPC: Modeling network contention in message-passing programs. *IEEE Transactions on Parallel and Distributed Systems*, 12(4):404–415, 2001.

[14] J. Pjesivac-Grbovic, T. Angskun, G. Bosilca, G. E. Fagg, E. Gabriel, and J. J. Dongarra. Performance analysis of MPI collective operations. In *Proceedings of the Wokshop on Performance Modeling, Evaluation and Optimisation for Parallel and Distributed Systems (PMEO), in IPDPS 2005*, 2005.

[15] L. A. Steffenel. *LaPIe: communications collectives adaptées aux grilles de calcul*. PhD thesis, INPG, France, December 2005.

[16] J. D. Teresco, J. Faik, and J. E. Flaherty. Resource-aware scientific computation on a heterogeneous cluster. *Computing in Science and Engineering*, 7(2):40–50, 2005.

[17] R. C. Whaley, A. Petitet, and J. J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001. Also available as University of Tennessee LAPACK Working Note #147, UT-CS-00-448, 2000 (`www.netlib.org/lapack/lawns/lawn147.ps`).

[18] R. Wolski, N. Spring, and C. Peterson. Implementing a performance forecasting system for metacomputing: The network weather service. In *Proceedings of the Supercomputing*, 1997.