Scheduling Heuristics for Efficient Broadcast Operations on Grid Environments

Luiz Angelo Barchet-Steffenel[1] and Grégory Mounié[2]

[1]LORIA,Université Nancy-2
Nancy, France
barchet@loria.fr

[2]Laboratoire ID-IMAG
Montbonnot St-Martin, France
mounie@imag.fr

Abstract

The popularity of large-scale parallel environments like computational grids has emphasised the influence of network heterogeneity on the performance of parallel applications. Collective communication operations are especially concerned by this problem, as heterogeneity interferes directly on the performance of the communication strategies. In this paper we focus on the development of scheduling techniques to minimise the total communication time (makespan) of a broadcast operation on a grid environment. We observed that most optimisation techniques present in the literature are unable to deal with the complexity of a large network environment. In our work we propose the use of hierarchical communication levels to reduce the optimisation complexity, while keeping high performance levels. Indeed, we propose three heuristics designed to meet the requirements of a hierarchically structured grid composed of tenths of clusters, a tendency for the next years.

1 Introduction

The *Broadcast* is one of the simplest collective communication operations. Initially, only one *root* process holds the message that should be broadcasted; at the end of the operation, every process has a copy of that message. In homogeneous environment, optimal broadcast trees can be easily depicted if we have the interconnection parameters. In heterogeneous systems, however, the problem of finding an optimal broadcast tree is NP-Complete [6, 5, 19], as the number of possible broadcast schedules for P nodes is exponential.

Due to this restriction, many works focus on the development of approximation techniques that are sufficiently efficient to be used in a real system. Most of these works, however, consider that every process in the system is individually considered during the optimisation. This approach becomes clearly expensive when the number of processes augments, as in the case of computational grids. One technique usually employed to simplify the optimisation consists on grouping processes into logical homogeneous clusters according to their relative performances (communication latency, for example). Indeed, communications on the network are split in two levels, *intra* and *inter*-cluster.

Therefore, in this paper we propose three optimisation heuristics that considers both *inter-cluster* latencies and *intra-cluster* broadcast time. Using the performance model pLogP to measure *inter-cluster* connectivity and to predict the communication cost of *intra-cluster* broadcasts [2, 4], we compare our *grid-aware* optimisation heuristics with traditional techniques from Bhat [6]. Indeed, we show that grid-aware optimisation heuristics are specially adapted to computational grids that span over several clusters.

In the next sections we evaluate different optimisation solutions for the broadcast communication pattern on grid environments. Section 2 discusses previous works on this subject, and their issues on grid-aware performance optimisation. Section 3 presents the formalism used to describe the heuristics, as well as the performance model used to predict *intra-cluster* broadcast performance. Section 4 presents traditional techniques such those proposed by Bhat [6]. In Section 5 we introduce three optimisation heuristics, whose differential is to take into account the communication time inside clusters. Section 6 compares such techniques through simulation, while Section 7 presents the results of an experiment conducted over a real grid environment. Finally, Section 8 presents our conclusions and future works.

1-4244-0054-6/06/$20.00

2 Related Works

The literature presents several works that aim to optimise collective communications in heterogeneous environments. While some works just focus on the search for the best broadcast tree of a network [5], most authors such as Banikazemi [1], Bhat [6], Liu [13], Park [17], Mateescu [16] and Vorakosit [18] try to generate optimal broadcast trees according to a given *root* process, which corresponds to the MPI_Bcast operation.

Unfortunately, most of these works were designed for small-scale systems. One of the first works on collective communication for grid systems was the ECO library proposed by Lowekamp [15, 14], where machines are grouped according to their location. Later, the same principle was used by the MPI library MagPIe [11], where processes are hierarchically organised in two levels with the objective to minimise the exchange of wide-area messages.

A common characteristic of these two implementations is that *inter-cluster* communications are structured in a flat tree, while *intra-cluster* communications benefit from efficient strategies like binomial trees. Hence, to improve communication performances, we must also improve *inter-cluster* communications. One of the first works to address this problem was presented by Karonis [8, 7]. In his work, Karonis defined a multilevel hierarchy that allows communication overlapping between different levels (cf. Table 1). While this structure on multiple levels allows a performance improvement, it still relies on flat trees to disseminate messages between two wide area levels (levels 0 and 1), the same strategy as ECO or MagPIe.

However, a flat tree is far from being an optimal tree shape for heterogeneous systems. Because network heterogeneity does not allow trivial solutions, we are constrained to generate specific broadcast trees for each environment. Due to complexity concerns, we cannot rely on exhaustive search of the optimal tree, which is exponential; we must then rely on optimisation heuristics. In this paper we use as reference some heuristics from Bhat [6], which try to construct efficient communication schedules that minimise the broadcast execution time.

Nevertheless, in this work we explore a different approach to improve communication efficiency. We consider that wide-area latency is no longer the single parameter that contributes to the communication time, as the *intra-cluster* communication time may represent an important optimisation parameter. Indeed, current clusters may be composed by several hundred nodes, and even with high performance network interconnections, a broadcast in a local area network may take several microseconds [4], sometimes as much as a long distance latency. Hence, we propose a smart schedule of wide-area collective communications, which considers both *inter* and *intra-cluster* times to minimise makespan.

3 Description Formalism and Performance Model

To describe the heuristics presented in the next sections, we use a formalism similar to the one used by Bhat [6]. Hence, we consider that clusters are divided in two sets, **A** and **B**. The set **A** contains the clusters that already received a message (i.e., the coordinator of the cluster receives it). In set **B** we found all clusters that shall receive the message. This way, set **A** initially contains only the cluster from the *root* process, while all other clusters are listed on the set **B**. At each communication round, two clusters are chosen from sets **A** (a sender) and **B** (a receiver). After communicating, the receiver cluster is transferred to set **A**. When a cluster does not participate in any other *inter-cluster* communication, it can finally broadcast the message among the cluster processes. This strategy improves the multiplication of data sources, giving more choices to the optimisation heuristics.

To model the communication performance of *intra-cluster* communication, we use the *parameterised LogP* model (*pLogP*) [9], an extension of the LogP performance model that can accurately handle both small and large messages with a low complexity. Hence, all along this paper we use $g_{i,j}(m)$ to represent the communication gap of a message with size m between two clusters i and j. Similarly, we use $L_{i,j}$ to represent the communication latency between these clusters. The *pLogP* parameters used to feed our models were obtained with the method described in [10].

4 Traditional strategies

4.1 Baseline Algorithm - Flat Tree

Used by ECO and MagPIe libraries, this strategy uses a flat tree to send messages at the *inter-cluster* level, i.e., the *root* process sends the message to the coordinators of all other clusters, in a sequential way.

Formally, the *root* process, which belongs to the set **A**, chooses a destination among the clusters in set **B**. At each communication round, the *root* process chooses a new cluster from set **B**, despite the presence of other (potential) sources in set **A**. Once a cluster coordinator receives a message, it broadcasts it inside the clusters using a binomial tree technique.

Table 1. Communication levels according to their latency [12]

Level 0 >	Level 1 >	Level 2 >	Level 3, 4, ...
WAN-TCP	LAN-TCP	localhost-TCP	shared memory Myrinet Vendor MPI

Although easy to implement, this strategy is far from being optimised. Indeed, the diffusion of messages does not take into account the performance of different clusters, neither the interconnexion speeds. Further, this technique depends on how the clusters list is arranged with respect to the root process, and important performance variations can be observed on applications that rotate the role of the broadcast root.

4.2 Fastest Edge First - FEF

Proposed by Bhat *et al.* [6], the *Fastest Edge First* heuristic considers that each link between two different processes i and j, corresponds to an edge with weight T_{ij}. Usually, this edge weight T_{ij} corresponds to the communication latency between the processes. To schedule the broadcast communications in a heterogeneous environment, the FEF heuristics order nodes from the set **A** according to their smallest outgoing edge weight. Once this smallest edge is selected, it implicitly designates the sender and receiver processes. When a receiver is chosen, it is transferred from set **B** to set **A**, and the minimal outgoing edge list is sorted again. Hence, the strategy behind this technique is to maximise the number of sender processes, augmenting the number of possible paths that can be explored to reach the more distant processes.

4.3 Early Completion Edge First - ECEF

In the previous heuristics, once the receiver was assigned it was immediately transferred to the set **A**, and could take part in the next communication round. This model, however, is not realistic, as communication delays may prevent a receiver process from having the message immediately. Indeed, it is possible that a process from set **A** is chosen to send a message before it has the message available for retransmission; in this case, communications are blocked until the message becomes available at the sender.

To avoid such situations, Bhat proposed the heuristic called *Early Completion Edge First*, which tries to keep an account of the moment in which a message becomes available to the processes in the set **A**. This way, a *Ready Time* (RT_i) parameter is evaluated conjointly with the transmission time between the processes, and the choice of the sender-receiver pair depends on the earliest possible moment when this transmission may effectively be finished, as stated by:

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

Hence, the final objective of the heuristic is to augment the number of sources that can *effectively* retransmit message to the other processes.

4.4 Early Completion Edge First with lookahead - ECEF-LA

While the precedent heuristic efficiently solves the problem of multiplication of sources that can effectively retransmit a message in a next communication round, it does not offers a guarantee that these new sources would be as efficient to transmit messages as well. To increase the efficiency of the ECEF heuristic, Bhat [6] proposed the use of *lookahead* evaluation functions to make a deep analysis on the scheduling choices.

In the variant called *Early Completion Edge First with lookahead* - ECEF-LA, the algorithm uses a *lookahead function* F_j to characterise each process in set **B**. This way, the sender-receiver pair will be the one that minimises the sum:

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

To define the *lookahead function* we can use several strategies. Bhat [6] proposed, for example, that F_j represents the minimal transmission time from process j to any other process in set **B**. Indeed, this function can be described as:

$$F_j \quad = \quad \min_{P_k \in B} \left(g_{j,k}(m) + L_{j,k} \right)$$

Hence, this lookahead function evaluates the utility of a process P_j if it is transferred to set **A**. Nevertheless, Bhat suggest some other *lookahead* functions like the average latency between P_j and the other processes in **B** or the average latency between processes in sets **A** and **B**, if P_j was transferred to set **A**.

5 Grid-aware strategies

All heuristics presented in the previous section schedule communications in the *inter-cluster* level using as only reference the communication cost between two different clusters. However, the makespan depends not only on the latency between different sites, but also on the broadcast time inside each cluster. In fact, the number of nodes that compose a cluster easily reaches a hundred machines, and the transmission time of a broadcast in such clusters sometimes exceeds the cost of a wide-area transmission.

With this concern in mind, we developed three "grid-aware" scheduling heuristics. We call them grid-aware because their scheduling strategies take into account also the broadcast time in the *intra-cluster* level. To develop these heuristics, we used two different approaches. The first two heuristics propose different *lookahead* functions to be used together with the ECEF-LA heuristic from Bhat [6]. In other words, we try to minimise the sum of the parameters:

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

The third heuristic, however, uses a different logic from the heuristics proposed by Bhat. Indeed, the approach used by Bhat always tries to minimise the factors related to the communication, what gives the priority the fastest clusters. Nevertheless, the critical path of a hierarchical broadcast depends mostly on the slow clusters. By this reason we also evaluate in this paper a heuristic based on a max-min optimisation strategy.

5.1 ECEF-LAt

The first heuristic adapted to grid environments that we propose is an extension of the ECEF-LA heuristic. This heuristic uses a *lookahead* function that evaluates both the communication cost at the *inter-cluster* level and T_i, the broadcast time inside a cluster i. Hence, we try to find a schedule that minimises the overall communication time to a distant cluster (the small t in the name indicates that we are looking for the minimum), and we use the lookahead function:

$$F_j = \min_{P_k \in B} (g_{j,k}(m) + L_{j,k} + T_k)$$

The reasoning of this strategy is that the receiver should be choose not only because it can efficiently retransmit messages to other clusters, but also because the clusters it can reach will likely complete their broadcasts within a reduced interval of time.

5.2 ECEF-LAT

Although its similarity with the precedent strategy, the ECEF-LAT strategy tries to maximise the sum of the parameters from the lookahead function F_j:

$$F_j = \max_{P_k \in B} (g_{j,k}(m) + L_{j,k} + T_k)$$

In fact, we observe that all precedent techniques tend to select fastest clusters over slowest or more distant ones. However, this behaviour only introduces extra retards to the termination of the clusters broadcast, which impacts directly the communication makespan.

In the ECEF-LAT strategy, we give priority to the clusters that need more time to finish theirs internal broadcasts, and we count on communication overlap at the *inter-cluster* level to limit the overall communication time.

5.3 BottomUp

A close analysis on the previous heuristics reveals that besides the use of different lookahead functions, the ECEF-LA* heuristics rely on *min-max* or *min-min* optimisation techniques. From the ECEF-LAT technique we observe that sometimes it is interesting to distribute messages to slow clusters first, as a mean to reduce the overall slowdown. Nevertheless, it is still necessary to multiply the number of sources, and the best strategy remains to contact faster clusters first. While these strategies seem to be in opposition, they are not mutually exclusive.

To combine these two strategies, we developed a new heuristic called BottomUp, where a *max-min* optimisation strategy is used to choose the sender that can reach the slowest cluster in the minimum time:

$$\max_{P_j \in B} (\min_{P_i \in A} (g_{i,j}(m) + L_{i,j} + T_j))$$

Indeed, this method allows the collective operation to contact slowest clusters as soon as possible, but also to release senders in an earlier moment, ready to be selected again.

6 Simulation

In order to better evaluate the efficiency of the heuristics presented above, we chose in a first moment to simulate the execution of the MPI_Bcast operation. For instance, we provide the heuristics with realistic communication parameters, obtaining a communication schedule that is used to calculate the makespan.

Therefore, at each iteration, the parameters L, g and T are randomly chose among the values presented in Table 2. These parameters correspond to real values measured over the French national grid GRID5000[1], and we our results correspond to the average of 10000 iterations.

Table 2. Performance parameters used in the simulations

	minimum	maximum
L	1 ms	15 ms
g	100 ms	600 ms
T	20 ms	3000 ms

Initially, we evaluate the behaviour of the heuristics in a grid with a reduced number of clusters, ranging from 2 to 10. This number corresponds to the majority of grid environments in use today: for example, the GRID5000 project currently interconnects 10 clusters. Indeed, Figure 1 shows the average completion time for a reduced number of clusters. As expected, a Flat Tree schedule presents the worst performance, as it does not try to improve the *inter-cluster* communication. Further, we also observe that the BottomUp heuristic presents a better performance than the FEF technique; this indicates that in a grid system it is sometimes more important to take into account the performance of slow clusters than the pure interconnection speed. We believe that this technique can be improved with the use of *lookahead* functions similar to those used by the ECEF-LA heuristics; indeed, a *lookahead* function can be used to guarantee that a receiver will be a good sender at his turn.

If Flat Tree and FEF heuristics clearly show their limitations and the BottomUp technique illustrates some interesting research directions, the best performance levels in our simulations were achieved by the ECEF* techniques. Because these heuristics are able to interleave communications from different clusters, the overall communication time does not increase linearly with the number of clusters.

These results can be better observed in Figure 2, which presents the expected performances for grids with up to 50 clusters. Although most grid systems are still composed by a small group of clusters, this number tends to increase in the next years, and therefore it is important to identify efficient techniques to meet these new constraints.

Hence, Figure 2 demonstrates that the Flat Tree approach is clearly inefficient for a large number of clusters. Similarly, the greedy algorithm FEF does not achieve good performance levels, as communication latency is not a sufficient parameter to balance communication times and minimise the makespan.

[1] http://www.grid5000.org

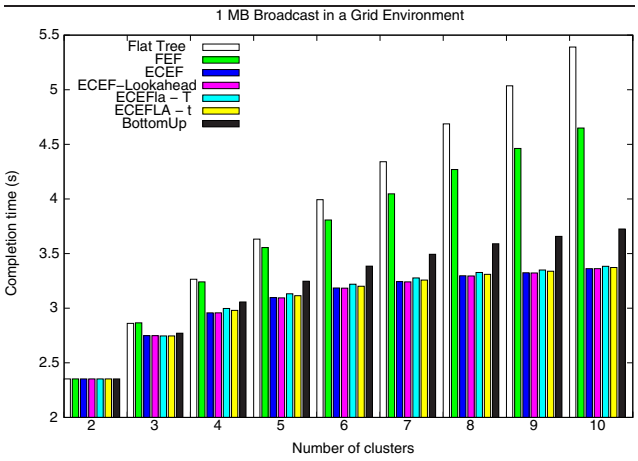

Figure 1. Simulation results for a broadcast with a reduced number of clusters

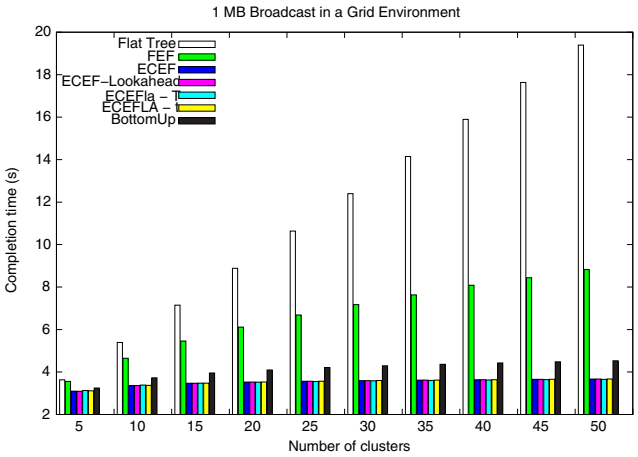

Figure 2. Simulation results for a broadcast in a grid with up to 50 clusters

To better evaluate the behaviour of ECEF-like heuristics, we present in Figure 3 the simulations results for these four techniques. In a first moment, we observe that the number of clusters has a little impact on the completion time. Because the ECEF* techniques try to improve parallel communications, they are able to interleave communications from different clusters in an efficient way. Nevertheless, the average performance of these heuristics is too similar to allow a better analysis.

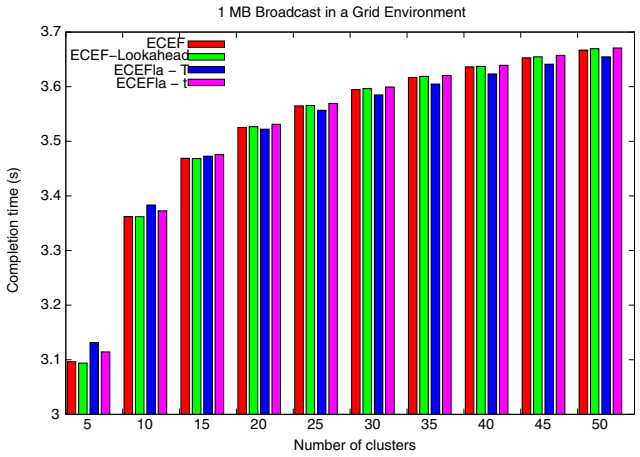

Figure 3. Simulations results for ECEF-like heuristics

As the average sometimes does not allow a better evaluation of the different techniques, we introduce a new comparison parameter called "hit rate". Because it is too expensive to find the optimal schedule when the number of clusters increases, we use the "global minimum" as reference for the efficiency of the heuristics. Therefore, this "global minimum" corresponds to the minimum scheduling time found on each simulation iteration, and the *hit rate* indicates the number of times a technique matches the global minimum. Hence, Figure 4 indicates the number of times each heuristic reaches the global minimum in 10000 iterations.

From the analysis of the hit rate we observe that the efficiency of the ECEF, ECEF-LA and ECEF-LAt techniques decreases with the number of clusters. Because these techniques give the priority to clusters with fast connections, their schedules are bounded by the communication delay on slower clusters, giving slightly under-optimal schedules when the number of clusters augments (although their average completion time remains small).

We also observe that the ECEF-LAT heuristic presents a hit rate that remains constant with the number of clusters. Contrarily to the other techniques, this heuristic tries to balance fast and slow clusters, resulting on a hit rate around 45%, i.e., it has a probability of 45% to obtain the best scheduling among all techniques. As result, the average completion time of the ECEF-LAT technique is slightly smaller than the completion time of the other techniques when the number of clusters is high, as shown in Figure 3.

Because the efficiency of the scheduling heuristics depends on the number of interconnected clusters, we suggest a mixed strategy, where the scheduling heuristic is defined according to the problem size. Indeed, we suggest the use of performance-oriented heuristics like ECEF or ECEF-LA when the number of clusters is reduced, and the ECEF-LAT technique for grid systems with more clusters. With this mixed strategy, we can always count on scheduling techniques that have a high probability to give an optimal communication scheduling.

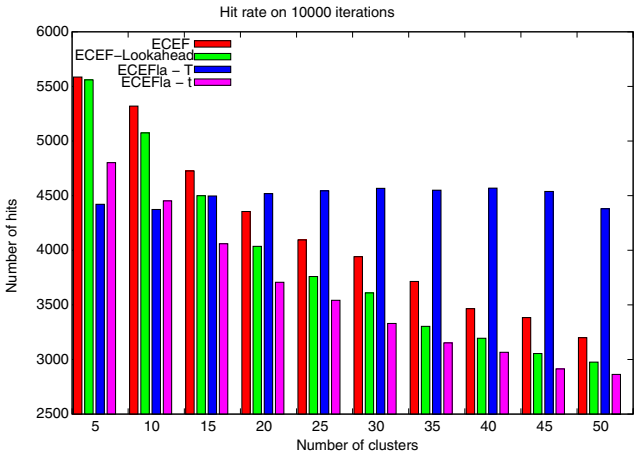

Figure 4. Hit rate of different ECEF-like heuristics

7 Practical Evaluation

While the previous section provides valuable information on the expected efficiency of different communication schedule heuristics, we still cannot evaluate the impact of these techniques on the performance of real implementations. In fact, most of the techniques presented in this paper may induce a scheduling cost that can affect the performance of the MPI_Bcast operation.

In order to evaluate the performance of the heuristics studied in this paper, we implemented these techniques on top of a modified version of the MagPIe library [11]. We improved MagPIe by extending it with the capability to acquire pLogP parameters and to predict the communication performance of homogeneous clusters, as explained in a previous paper [3].

Hence, to evaluate the real performance of the different heuristics, we run a test experiment using 88 machines from the GRID5000 environment, split in 6 homogeneous clusters, according to cluster map provided by Lowekamp's algorithm [14] with a tolerance rate $\rho = 30\%$. As a result, Table 3 indicates the la-

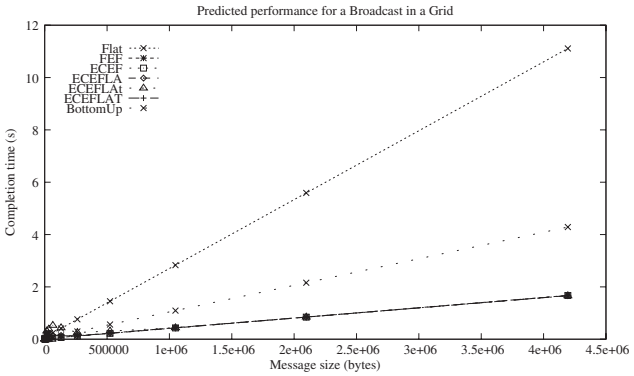

Figure 5. Predicted time for a 88 machine grid

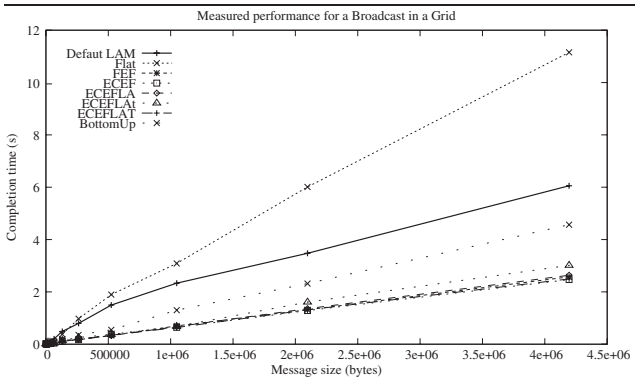

Figure 6. Measured completion time for a 88 machine grid

tency between every two clusters or between two machines in the same cluster (except for the clusters that have only one single machine). Therefore, some clusters like IDPOT were subdivided in different homogeneous clusters, according to their real communication performance [3].

As a result of this experience, Figure 5 presents the performance predictions for these heuristics as stated by the communication models, while in Figure 6 we present the measured times. To better evaluate the performance speed-up obtained with the use of scheduling heuristics, we compare the performance of the "pure" MPI_Bcast operation, which uses a simple "grid unaware" binomial tree.

We observe that performance predictions fit with a good precision the practical results, which is especially true for simple techniques like the Flat Tree heuristic, which induce a little overhead on the MPI_Bcast implementation. Hence, the algorithm complexity is a factor that must be considered when implementing more elaborate techniques like ECEF-LAT.

Nevertheless, we observe that ECEF-like heuristics achieved the best performance levels, with less than 3 seconds for a 4 MB message; at the opposite side, the Flat Tree strategy required almost six times more time to execute the broadcast, performance that is even worst than the "grid-unaware" binomial tree algorithm traditionally used by MPI.

Therefore, this experience illustrates the fact that the Flat Tree technique cannot be used to efficiently optimise communications on a grid system. In fact, this technique is too dependent on the topology description provided at the beginning of the execution, and therefore cannot adapt to changes that occur during the execution of the parallel application, as for example the use of different *root* processes for the broadcast.

8 Conclusions and Future Works

The growing popularity of heterogeneous parallel processing environments like clusters and grids emphasises the impact of network heterogeneity on the performance of parallel applications. Collective communication operations are especially concerned by this problem, as heterogeneity interferes directly on the performance of the communication strategies.

In this paper, we compare different optimisation heuristics using both simulations and practical experiments. Indeed, we show that traditional "grid-aware" techniques are far from being optimal, and that therefore we need to develop new heuristics specially designed for large-scale grid systems. Hence, we propose three techniques especially designed to take into account the communication performance of both *intra* and *inter-cluster* communications.

Further, we investigate the behaviour of these techniques when the number of interconnected clusters augments, a tendency for the next years. We show that most of existent techniques tend to provide sub-optimal schedules if the number of clusters augments too much. Indeed, we demonstrate that one of the techniques proposed in this work guarantees a constant efficiency rate in spite of the number of interconnected clusters. This result is especially important to direct the development of next-generation optimisation techniques.

Therefore, we should continue our work on the development and evaluation of grid-aware collective communication. We are particularly interested on the development of efficient communication schedules for other communication patterns like *scatter* and *alltoall*. These communication patterns are widely employed by parallel scientific applications and can benefit from grid-aware optimisations.

Table 3. Latency between different clusters (in microseconds)

	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
	31 x Orsay	29 x Orsay	6 x IDPOT	1 x IDPOT	1 x IDPOT	20 x Toulouse
Cluster 0	47.56	62.10	12181.52	12187.24	12197.49	5210.99
Cluster 1	62.10	47.92	12181.52	12198.03	12195.22	5211.47
Cluster 2	12181.52	12181.52	35.52	60.08	60.08	5388.49
Cluster 3	12187.24	12198.03	60.08	-	242.47	5393.98
Cluster 4	12197.49	12195.22	60.08	242.47	-	5394.10
Cluster 5	5210.99	5211.47	5388.49	5393.98	5394.10	27.53

References

[1] M. Banikazemi, V. Moorthy, and D. K. Panda. Efficient collective communication on heterogeneous networks of workstations. In *International Conference on Parallel Processing (ICPP'98)*, pages 460–467, 1998.

[2] L. Barchet-Estefanel and G. Mounié. Fast tuning of intra-cluster collective communications. In *Proceedings of the Euro PVM/MPI 2004*, LNCS Vol. 3241, pages 28–35, 2004.

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

[4] L. Barchet-Estefanel and G. Mounié. Performance characterisation of intra-cluster collective communications. In *Proceedings of the 16th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2004)*, pages 254–261, 2004.

[5] O. Beaumont, L. Marchal, and Y. Robert. Broadcasts trees for heterogeneous platforms. In *International Parallel and Distributed Processing Symposium (IPDPS 2005)*, 2005.

[6] P. B. Bhat, C. Raghavendra, and V. Prasanna. Efficient collective communication in distributed heterogeneous systems. *Journal of Parallel and Distributed Computing*, (63):251–279, 2003.

[7] N. T. Karonis, I. Foster, B. Supinski, W. Gropp, E. Lusk, and S. Lacour. A multilevel approach to topology-aware collective operations in computational grids. Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, 2002.

[8] N. T. Karonis, B. Supinski, I. Foster, W. Gropp, E. Lusk, and J. Bresnahan. Exploiting hierarchy in parallel computer networks to optimize collective operation performance. In *14th International Conference on Parallel and Distributed Processing Symposium*, pages 377–384, 2000.

[9] 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.

[10] T. Kielmann, H. Bal, and K. Verstoep. Fast measurement of logp parameters for message passing platforms. In *4th Workshop on Runtime Systems for Parallel Programming*, LNCS Vol. 1800, pages 1176–1183, 2000.

[11] T. Kielmann, R. Hofman, H. Bal, A. Plaat, and R. Bhoedjang. Magpie: Mpi's collective communication operations for clustered wide area systems. In *7th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 131–140, 1999.

[12] S. Lacour, N. T. Karonis, and I. Foster. Mpich-g2 collective operations performance evaluation, optimizations. Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, 2001.

[13] P. Liu, D.-W. Wang, and Y.-H. Guo. An approximation algorithm for broadcast scheduling in heterogeneous clusters. In *Real-Time and Embedded Computing Systems and Applications, 9th International Conference (RTCSA 2003)*, LNCS 2968, pages 38–52, 2004.

[14] B. Lowekamp. Discovery and application of network information. Technical report, Carnegie Mellon University, 2000.

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

[16] G. Mateescu. A method for mpi broadcast in computational grids. In *International Parallel and Distributed Processing Symposium (IPDPS 2005)*.

[17] J.-Y. L. Park, H.-A. Choi, N. Nupairoj, and L. M. Ni. Construction of optimal multicast trees based on the parameterised communication model. In *International Conference on Parallel Processing (ICPP 1996)*, pages 180–187, 1996.

[18] T. Vorakosit and P. Uthayopas. Generating an efficient dynamic multicast tree under grid environnement. In *Euro PVM/MPI 2003*, LNCS 2840, pages 636–643, 2003.

[19] J.-J. Wu, S.-H. Yeh, and P. Liu. Efficient multiple multicast on heterogeneous network of workstations. *The Journal of Supercomputing*, (29):59–88, 2004.