High-Performance Multi-Rail Support with the NEWMADELEINE Communication Library*

Olivier Aumage, Élisabeth Brunet, Guillaume Mercier, Raymond Namyst

INRIA - LABRI
Université Bordeaux 1
TALENCE, FRANCE
{aumage, brunet, mercier, namyst}@labri.fr

Abstract

This paper focuses on message transfers across multiple heterogeneous high-performance networks in the NEWMADELEINE *Communication Library.* NEWMADELEINE *features a modular design that allows the user to easily implement load-balancing strategies efficiently exploiting the underlying network but without being aware of the low-level interface. Several strategies are studied and preliminary results are given. They show that performance of network transfers can be improved by using carefully designed strategies that take into account NIC activity.*

1 Introduction

Clusters have now widely, if not universally, been adopted in academic and industrial environments. Cluster architectures and particularly their core —the interconnect network— have evolved considerably from the initial Fast Ethernet to high performance dedicated networks, such as the Myrinet family from Myricom[5], the QsNet family from Quadrics[4], the various Infiniband solutions [1], the SCI networks from Dolphinics [9], to name a few, each featuring extra-low latency and substantial bandwidth.

All these network solutions present different characteristics. Quadrics and SCI have long been known for their particularly low latency for instance. The Myri-10G network and some recent Infiniband declinations deliver better bandwidths than the other solutions. It is therefore increasingly interesting to interconnect clusters with multiple heterogeneous networks to get better overall performance over the spectrum of message lengths, and possibly also to benefit from combining the specific features, properties and programming model of each solution.

Therefore, efficiently exploiting an heterogeneous set of networks at the level of a communication library is a non-trivial task. Should the library exclusively use one network for sending packets of a given size or load balance some of these packets onto the multiple networking channels? Should some packets be split? If yes, how should packets be split? For which size of packets is it interesting to perform heterogeneous multi-rail transfers? Do we actually get aggregated bandwidth? Do we get bus contentions?

In this paper, we report on a first exploration of these questions through a set of experiments we have conducted, using a combination of a Elan/QsNetII Quadrics network [16] and a MX/Myri-10G Myricom network [12] with our NEWMADELEINE communication library [6]. Indeed the exclusive layout of the NEWMADELEINE library architecture allowed us to easily experiment with different sets of communication request scheduling and optimization strategies through pluggable communication schedulers. We present the first conclusions we have drawn from these experiments and we expose the scheduling/optimizing strategy for heterogeneous networking that we have incrementally deduced from these conclusions.

The structure of this paper is as follows. In Section 2, we present the NEWMADELEINE communication library. In Section 3 we discuss more deeply the issues raised in transferring data across multiple heterogeneous high-performance networks and our process in elaborating a strategy implemented in NEWMADELEINE that addresses them. Section 4 concludes this paper and discusses future works.

*This work was funded through the ANR-05-CIGC-11 grant.

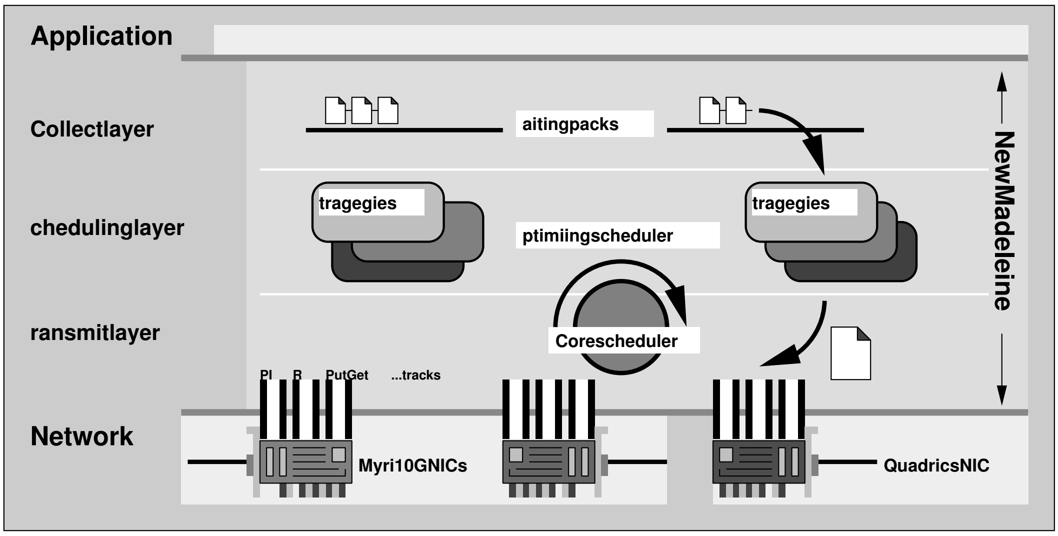


Figure 1. NEWMADELEINE **architecture overview**

2 Overview of the NEWMADELEINE Communication Library

In the following paragraphs, we present our NEWMADELEINE communication library. NEWMADELEINE is a complete re-design of the MADELEINE [3] communication library, to help in elaborating communication request schedulers and optimizers.

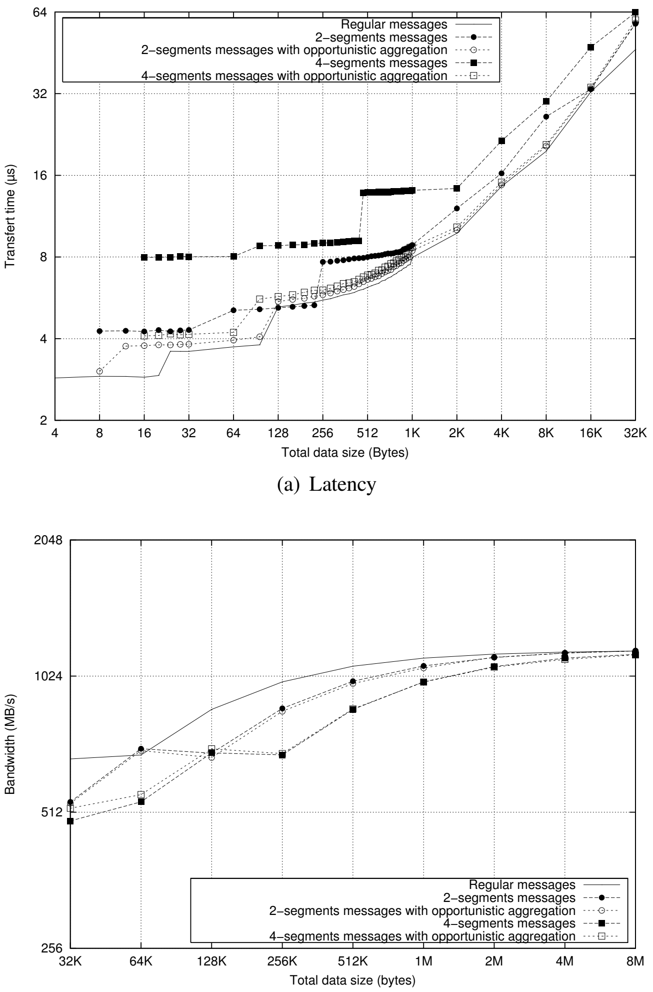
The library architecture adopts a three-layers layout (see Figure 1). The top level layer is responsible for *collecting* application communication requests. It provides the top-level programming interface. Since NEWMADELEINE is organized in a modular fashion, several flavors of APIs may be implemented. For the benchmarking programs used in this paper, we employed an API providing a message-passing oriented model. Messages may be constituted of one or more segments through incremental message construction/extraction commands.

At the lowest part, the *transmit* layer is in charge of interfacing NEWMADELEINE with the networking hardware and associated specific APIs through a set of drivers. NEWMADELEINE currently provides drivers for the Quadrics Elan API [10], the Myricom Myrinet Express [12] and GM-2 APIs [17], the Dolphinics SiSCI API [9] and the legacy socket API on top of TCP/IP.

The middle layer is the most important part of the NEWMADELEINE architecture. It is made of interchangeable modules, each implementing an optimizing scheduler. The selected optimization scheduler is in charge of rewriting collected application requests in accordance with some optimizing policy —we use the term *strategy* for this— and of generating actual packets to be sent (or received) at the network level.

Communication library engines traditionally run in close relationship with the application requests: upon a call to its API, the engine usually decides whether to process the request immediately or to defer its processing to some subsequent API call. Such was, for instance, the model followed by the MADELEINE 3 communication library [3].

The NEWMADELEINE communication engine radically breaks with this traditional scheme. The engine now runs in relationship with the NIC (or NICs) activity. Request processing has been disconnected from the API functions called from the application. A transversal global scheduler is in charge of controlling the overall functioning of the library in link with the drivers, for NICs monitoring. When some NICs become idle, the global scheduler ensures that the optimizing scheduler is queried for some new packet — the most appropriate one in accordance with the optimizing scheduler strategy— to send or to receive in order to keep the NICs busy. This way, the communication support accumulates packets while the NIC is busy and once the NIC becomes idle, the optimizer processes the backlog of accumulated packets and picks a request that is submitted to the NIC, making it resume its work. This approach seamlessly allows the building of a packet optimization window to work on during phases when application execution is communication-bounded while keeping the cost of communication requests low when application execution is CPU-bounded.



(a) Latency

(b) Bandwidth

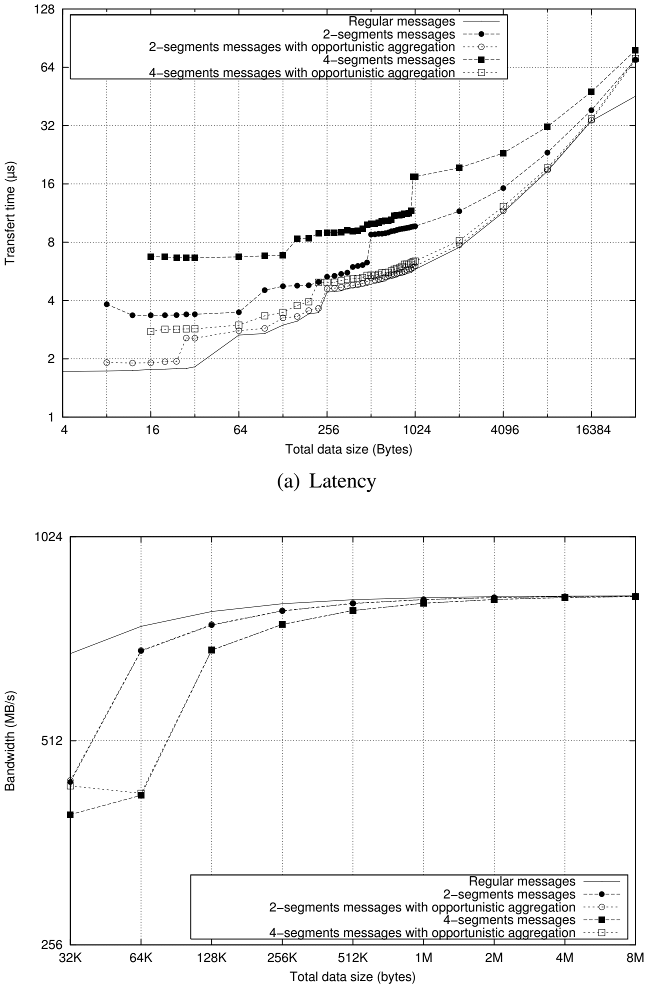
Figure 2. Raw performance of NEWMADELEINE **over Myri-10G for regular and multi-segments messages.**

3 Challenges with High-Performance Multi-Rail Management

The availability of several networks within the same cluster is potentially advantageous since the performance of the multiple networks can be aggregated in order to achieve a higher level of performance. The NEWMADELEINE communication library supports efficiently a significant set of high-performance networking technologies and it is therefore tempting to create a software element that can take advantage of multi-rail configurations. Since the scheduling in NEWMADELEINE is independent from the low-level network drivers, we could easily achieve the development of a basic packet scheduler tailored for multi-rail clusters. However, in order to design a better scheduling policy that could efficiently apply to all packet sizes, we decided to follow an incremental approach. We thus started from a basic scheme and enriched it with experimental feedback so as to create a more powerful strategy.

3.1 Experimental platform

All experiments presented in this paper have been carried out on the same platform, a set of two dual-core 1.8 GHz OPTERON boxes with 1MB of L2 cache and 1GB of main memory. The OS is Linux, with kernel version 2.6.17. Both nodes are interconnected with two high performance networks: a MYRI-10G network interface card (NIC) with the MX1.2.0 driver and a QUADRICS QM500 NIC with the ELAN driver. The benchmark is a regular ping-pong



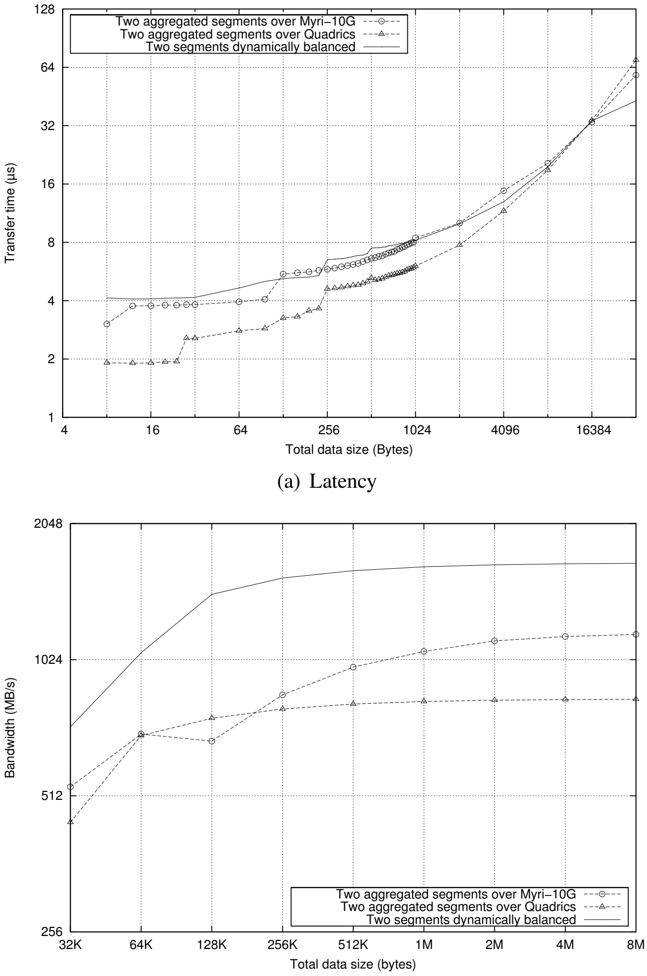
(a) Latency

(b) Bandwidth

Figure 3. Raw performance of NEWMADELEINE **over Quadrics for regular and multi-segments messages.**

program where the send (resp. recv) sequence is a serie of *non-blocking send* (resp. *non-blocking recv*) operations. We compare the transfer of regular messages (i.e. composed of a single contiguous memory segment) with the transfer of messages composed of multiple segments of the same size. This latter case represents situations when the application has to transmit non-contiguous messages or when multiple non-blocking send operations are performed within a short time interval.

Figures 2(a) and 2(b) show the raw performance of NEWMADELEINE over Myri-10G. For multi-segments messages, the size given on the abscissa is the accumulated size of all the segments. On Myri-10G, NEWMADELEINE exhibits a latency of 2.8μs and a maximal bandwidth of approximately 1200MB/s. The most interesting point is to observe the gap between the time to send a single segment and the time to send the same amount of data in the case of multi-segments messages. Actually, for "small" messages (i.e. with a size lesser than 16 KB in our test), the best solution is to copy the segments into a contiguous memory area and to send them as a single chunk. Figure 2(a) shows the results obtained using such an *opportunistic aggregation* for messages composed of respectively two and four segments. Note that the overhead incurred by memory copies is very low. Figures 3(a) and 3(b) show the same experiment over the Quadrics network. The latency is 1.7μs and the maximal bandwidth is approximately 850MB/s. One can notice that the gain of aggregating small packets on Quadrics is even bigger than on Myri-10G.



(a) Latency

(b) Bandwidth

Figure 4. Performance of the greedy balancing strategy with 2-segments messages.

3.2 Distributing data among heterogeneous rails: a first approach

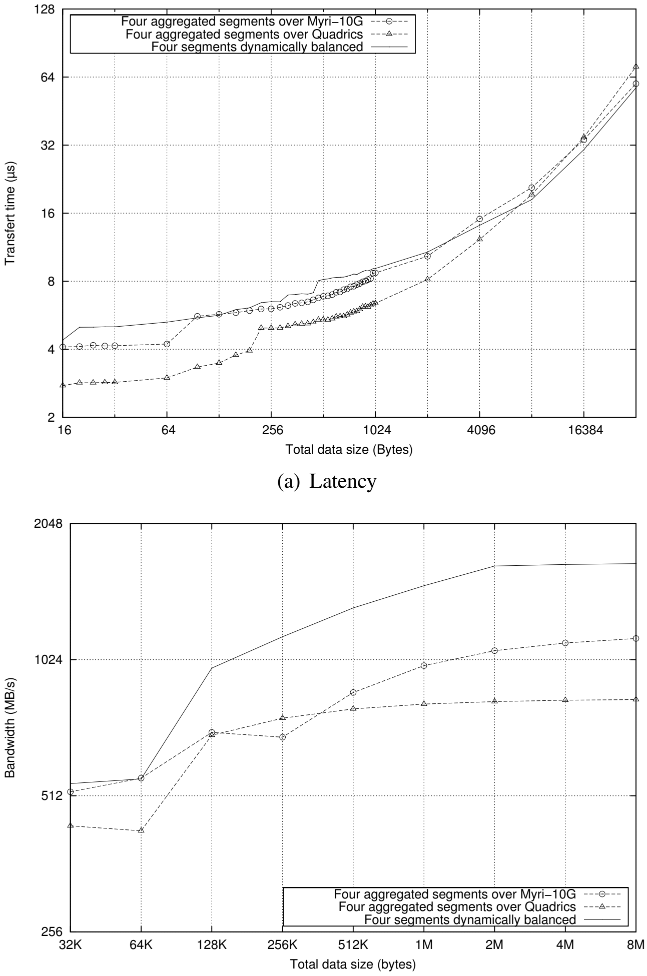
As a first step towards the design of an efficient multi-rail communication engine, we did implement a NEW-MADELEINE strategy that simply balances data segments on the sender side following a *greedy* policy: each time a NIC becomes idle, the strategy code is invoked and simply sends the first available segment (if any) on the corresponding network.

Figures 4(a) and 4(b) present the results obtained using this strategy on our 2-rails platform with 2-segments ping-pong. In this test, the greedy strategy leads to send the two segments simultaneously over separate networks. As a reference point, we also give the results when we force all the segments to be sent sequentially over a single network. The message size on the graphs represents the total size of the cumulated segments.

One can observe that the *greedy* strategy is able to achieve a higher maximum bandwidth (1675 MB/s) than a strategy that would use only one of the available networks, thanks to the I/O bus of our motherboard which is theoretically able to support data transfers up to approximately 2GB/s.

The second point revealed by the curves is that using simultaneously Myri-10G and Quadrics is only valuable when the amount of data is greater than 16KB, that is, for segments greater than 8KB. Actually, this is due to the way messages are sent to the NIC at the driver level: small messages are transferred from host memory to the NIC using a *Programmed I/O* (PIO) operation. This technique, in contrast with the *Direct Memory Accesses* (DMA) technique, monopolizes the CPU and prevents the overlapping of part of the message transfer with other computations.

To confirm these observations, we conducted the same experiment with 4-segments messages (Figures 5(a)



(a) Latency

(b) Bandwidth

Figure 5. Performance of the greedy balancing strategy with 4-segments messages.

and 5(b)). As expected, the results exhibit the same overall behavior. Note that in the case of large data transfers, the bandwidth achieved is still interestingly rather high in spite of the additional processing due to the handling of a larger number of elementary transfers.

3.3 Aggregation of small messages

In order to submit the small messages as a single request to the fastest network, an improvement, would be –according to results shown in Section 3.2– to *aggregate* the small messages. We therefore implemented a second version of our strategy which aggregates small messages as soon as they are submitted, favoring their transfer on the fastest network (that is, Quadrics) and proceeding afterward in a greedy fashion. Basically, when running the same test as in Section 3.2, the small messages are aggregated into one message which is sent over Quadrics. As for the large messages, they are balanced over the 2 NICs (one over MX/Myri-10G and one over Elan/Quadrics). Figure 6 shows the performance improvement for the small messages.

However, there is a gap between our strategy and the Quadrics NIC-only solution: despite the fact that our strategy does not entrust any communication request to the Myri-10G NIC, NewMadeleine has to take it into account. This overhead is therefore mainly due to a polling operation on the Myri-10G NIC. This penalty is mandatory if one wants to effectively use the multi-rail feature of NewMadeleine.

This strategy's second version is efficient for both small and large messages. Nevertheless when comparing the bandwidth gap between Elan/Quadrics and MX/Myri-10G in the large message case, some improvements could still be made if we straightforwardly split the packing operations into several chunks and balance them on the different NICs.

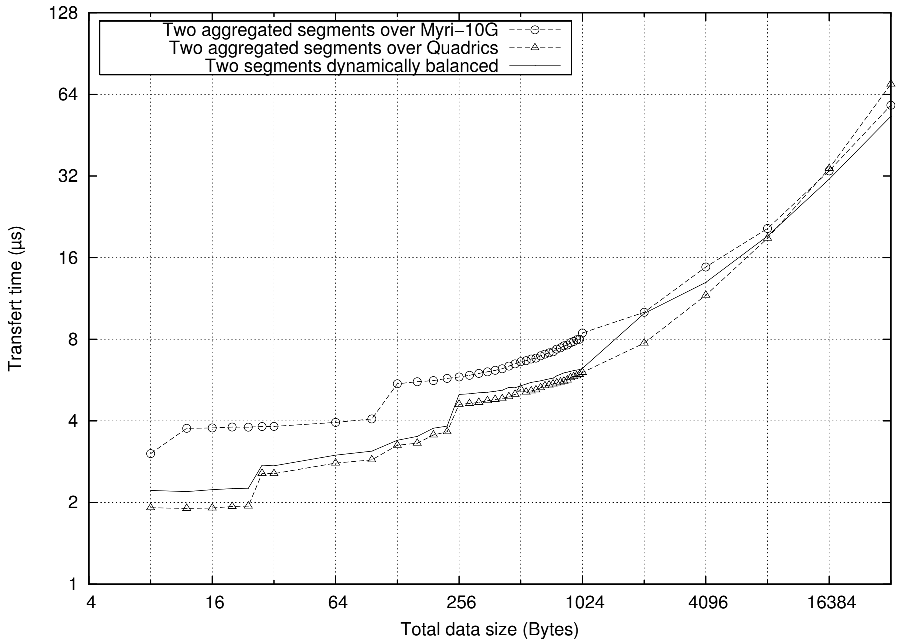


Figure 6. Aggregated eager messages on the fastest NIC and balanced large messages on available NICs - Latency.

3.4 Packet stripping with adaptive threshold

The third version of the multi-rail strategy applies only to large messages. They are stripped into packs large enough in order to avoid the transfer of the different chunks with a PIO operation and thus be falling in the same configuration as in Section 3.2. The packs stripping policy is to get fragments for which transfer time are equivalent on their respective networks. With the heterogeneous nature of the available technologies, the various packs are likely to be split into chunks of different sizes. Indeed, according to samplings performed on the different available NICs (this step is done at the NEWMADELEINE initialization time), an adaptive stripping ratio can be determined. We therefore refined our strategy: it now splits the packs and sends the chunks on the relevant NICs based on those ratios.

As MX/Myri-10G bandwidth performance is better than Elan/Quadrics', the major part of the initial segment must be sent through Myri-10G. Figure 7 presents the results of this strategy on a ping-pong program. The bandwidth is indeed improved when the chunks are adaptively formed from preliminary network samplings and afterward transferred over the two NICs. This last experiment proves the relevance of our scheme for the large messages. It must therefore be employed in coordination with the previous strategy in order to gain enhanced performance whatever the message size is. Finally, one clever balancing strategy over Myri-10G and Quadrics is to massively aggregate the small messages, to favor the sending of the resulting message over Quadrics, to split the large ones following some previously processing ratios when both NICs are available and if not, to send them over the first free one.

3.5 Related Work

Several projects have been exploring a way to support multi-rail features: LA-MPI[2, 7] is an MPI implementation that is able to send messages over heterogeneous networks. It is also able to strip a single MPI message over an homogeneous multi-rail network, but not over an heterogeneous one. This project is now part of the Open MPI consortium, providing helpful experience in this matter. As such, Open MPI is able to effectively perform heterogeneous stripping of MPI messages [8]. However, this project addresses for now only the large messages case. Another MPI implementation claims to support such a feature: MPICH-VMI2[15, 14]. However, we were unable to assess if this support is actually implemented in the current release. Moreover, Municluster [11] allows load balancing between multiple networks channels including some packet stripping strategies but only through the Socket APIs.

Our approach is original in several respects. Firstly, we apply our optimization strategy to the whole communica-

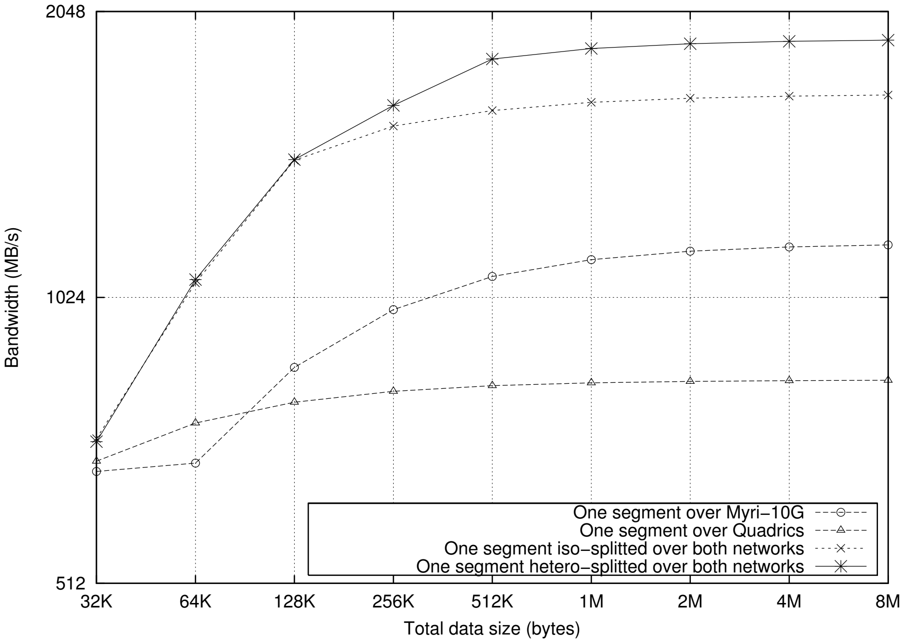


Figure 7. Packet stripping with adaptive threshold - Bandwidth.

tion flow between pairs of machines, regardless of the communication channels used by the upper layers. Secondly, the optimization engine is triggered only when one NIC becomes idle, so we take our scheduling decisions *just-in-time*. Finally, although the strategy code is a generic plug-in, it uses data sampling and driver capabilities provided by the underlying layer to determine the appropriate thresholds when aggregating or splitting message segments.

4 Discussion and Future Work

This paper presents some promising experiments using the NEWMADELEINE communication library to perform multi-rails data transmissions over multiple, heterogeneous, high-speed networks. The NEWMADELEINE library was designed to ease the development of portable data transfer optimizations on top of high-speed networks. We have incrementally developed and tuned an optimization *strategy* that tries to balance network traffic across the available physical links so as to both minimize communication latency and increase data throughput.

The versatility of the NEWMADELEINE communication engine allowed us to easily investigate aggressive optimizations. Data segments can be aggregated into the same physical packet even if they belong to different logical channels (e.g. different MPI communicators). They can be reordered so as to group small segments, or even sent out-of-order. Finally, large data segments can be split on the sending side (and later reassembled on the receiving side) into several chunks that may be sent through different networks.

The experiments conducted over Myri-10G and Quadrics exhibit very good overall performance and show the benefits of using multiple physical networks when exchanging data starting from 32KB-length messages. However, our current implementation is unable to take advantage of concurrent data transfers that do not involve DMA operations. We are currently designing a multi-threaded implementation that will process parallel PIO transfers on multiprocessor machines.

In the short term, we also plan to update our implementation of MPICH-Madeleine [13] so as to use the multi-rail capabilities of NEWMADELEINE. This will allow us to further experiment and enhance our techniques onto a wide range of applications.

References

[1] I. T. Association. Infiniband technology overview, 2005. `http://www.infinibandta.org/`.

[2] R. T. Aulwes, D. J. Daniel, N. N. Desai, R. L. Graham, L. D. Risinger, M. A. Taylor, T. S. Woodall, and M. W. Sukalski. Architecture of la-mpi, a network-fault-tolerant mpi. In *Proc. 18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*, April 2004.

[3] O. Aumage, L. Bougé, A. Denis, L. Eyraud, J.-F. Méhaut, G. Mercier, R. Namyst, and L. Prylli. A portable and efficient communication library for high-performance cluster computing. *Cluster Computing*, 5(1):43–54, 2002.

[4] J. Beecroft, D. Addison, F. Petrini, and M. McLaren. QsNetII: An Interconnect for Supercomputing Applications. In *Proceedings of the 15th Hot Chips International Conference*, 2003. `http://hpc.pnl.gov/people/fabrizio/papers/hot03.pdf`.

[5] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W.-K. Su. Myrinet: A gigabit-per-second local area network. *IEEE-Micro*, 15(1):29–36, Feb. 1995.

[6] E. Brunet, O. Aumage, and R. Namyst. Short paper : Dynamic optimization of communications over high speed networks. In *HPDC-15, The 15th IEEE International Symposium on High Performance Distributed Computing*, Paris, June 2006. Poster Session.

[7] S. Coll, E. Frachtenberg, F. Petrini, A. Hoisie, and L. Gurvits. Using Multirail Networks in High-Performance Clusters. In *Proceedings of the 3rd IEEE International Conference on Cluster Computing*, pages 15–24. IEEE Computer Society, 2001.

[8] R. L. Graham, G. M. Shipman, B. W. Barrett, R. H. Castain, G. Bosilca, and A. Lumsdaine. Open MPI: A high-performance, heterogeneous MPI. In *Proceedings of the Fifth International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks*, Barcelona, Spain, September 2006.

[9] D. Interconnect. Sisci documentation and library. `http://www.dolphinics.no`.

[10] Q. Ltd. Elan programming manual, 2003. `http://www.quadrics.com/`.

[11] N. Mohamed. Self-configuring communication middleware model for multiple network interfaces. In *29th Annual International Computer Software and Applications Conference (COMPSAC'05)*, 2005.

[12] I. Myricom. Myrinet express (mx): A high performance, low-level, message-passing interface for Myrinet. 2006.

[13] Olivier Aumage and Guillaume Mercier. MPICH/MadIII: a Cluster of Clusters-Enabled MPI Implementation. In *Proc. 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003)*, pages 26–35, Tokyo, May 2003.

[14] S. Pakin and A. Pant. VMI 2.0: A dynamically reconfigurable messaging layer for availability, usability, and management. In *The 8th International Symposium on High Performance Computer Architecture (HPCA-8), Workshop on Novel Uses of System Area Networks (SAN-1)*, Cambridge, Massachusetts, Feb. 2, 2002. Available from `http://www.c3.lanl.gov/PAL/publications/papers/Pakin2002:VMI.pdf`.

[15] A. Pant and H. Jafri. Communicating efficiently on cluster based grids with mpich-vmi. In *IEEE International Conference on Cluster Computing (CLUSTER 2004)*, pages 23–33, September 2004.

[16] F. Petrini, E. Frachtenberg, A. Hoisie, and S. Coll. Performance Evaluation of the Quadrics Interconnection Network, April 2003.

[17] R. Zamani, Y. Qian, and A. Afsahi. An evaluation of the myrinet/gm2 two-port networks. In *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*, pages 734–742, Washington, DC, USA, 2004. IEEE Computer Society.

Biographies

Olivier AUMAGE got its Ph.D in 2002 at the École normale supérieure de Lyon, France in the LIP laboratory. 2002-2003, he held a post-doc position at Vrije Universiteit (VU) in Amsterdam, NL, on the Ibis project in the team of Henri Bal. Since 2003, he holds a full-time junior researcher position at INRIA in Bordeaux, France, in the LABRI laboratory.

Élisabeth BRUNET is a PhD student, member of the INRIA Runtime team of LaBRI (University of Bordeaux, France) since october 2005 (`http://www.labri.fr/perso/brunet/`). Her primary research interests focus on high performance networking, generic communications algorithms for high-speed networks and communication middleware programming. She is a main designer of the NEWMADELEINE high performance communication library.

Guillaume MERCIER is currently Assistant Professor at Bordeaux's Engineering School (ENSEIRB, Bordeaux, France) where he teaches networks and distributed computing. His research interests cover high-speed networks, cluster computing and parallel programming environments and tools. He is actively involved in the development of MPI implementations with projects such as MPICH-Madeleine and the more recent MPICH2-Nemesis communication subsystem.

Raymond NAMYST received his Ph.D. in Computer Science from Lille in 1997. He held the position of assistant professor in the Computer Science Departement of the Ecole Normale Superieure of Lyon (1997-2002). In 2002, he joined the Computer Science Laboratory of Bordeaux (LaBRI) where he holds a Professor position. Raymond Namyst is the head of the Runtime INRIA research project, devoted to the design of high performance runtime systems for parallel architectures. His main research interests are in parallel computing, thread scheduling on multiprocessor architectures, communications over high speed networks and communications within Grids. He has played a major role in the development of the PM2 software suite, and has written numerous papers about the design of efficient runtime systems.