1. Problem statement and related work

Related work

My research in the Heterogeneous Computing Laboratory covers a number of topics - high performance computing, parallel programming models, collective operations, communication trees, communication performance models for predicting and optimizing communication, and heterogeneity of communication networks and communication models. A short introduction to each of these topics will be given here.
• **High-performance computing** is an area of computing focused on large-scale systems and applications. It involves different disciplines including computer science, mathematics and many others. Computer science and mathematics are needed to develop and optimize efficient algorithms for various problems defined in other disciplines like medicine, biology, engineering etc. Typical high performance applications include a huge range of simulations in a wide range of sciences. The complexity of algorithms for such applications requires the use of fast computers. The computers with highest processing capacity today are known as supercomputers. The TOP500 [11] project started in 1993 provides a ranking of the fastest supercomputing platforms in the world. The current number 1 in the list, the K Computer in Japan, performs as one million linked desktop computers. It “will be used in the natural sciences (physics, chemistry, and biology), nanoscience, the life sciences, engineering, and the prediction of environmental change and natural disaster” [22].

![K Computer](image)

*Figure 1: The fastest supercomputer in the 2011 TOP500 list – the K Computer in Kobe (Japan)*

However, significantly less efficient platforms are more accessible to scientists around the world for research and development, for example grid infrastructures [13].

• **Parallel programming models:** The most popular parallel programming model in high-performance computing is message passing. In a typical scenario, a complex problem is divided among (possibly distributed) processes with distributed memory. Since the memory is not shared, they can only exchange information through explicitly passing messages between each other. The Message Passing Interface [10] is the standard for writing parallel programs using message passing in the last decade. The research in this work focuses on MPI. The second most popular parallel programming model is shared-memory programming, and OpenMP [16] is the most popular interface for shared-memory programs today. Many alternative parallel programming interfaces exist, including UPC, Co-Array Fortran, High Performance Fortran, and others. More recently accelerator-based interfaces like CUDA and OpenCL have emerged.

• **Collective communication operations** are part of the MPI standard. In the standard a collective communication is defined as communication that involves a group of processes. The collectives in the standard include broadcast, reduce, scatter, gather etc. The motivation for the inclusion is to provide a simple, correct, portable and efficient implementation for these higher-level operations since they are widely used in parallel algorithms. For example, collective
communication is used in matrix-matrix multiplication, which is the basis for many applications in different areas.

- **Communication trees** are an important theoretical and practical component for collective communication. They represent a schedule for performing a collective operation as a sequence of point-to-point calls, since only point-to-point calls can be directly implemented on modern clusters of workstations. A node of the communication tree represents an MPI process, and a directed edge represents a point-to-point communication between two such processes. For example, the simple binomial tree in Figure 2 is one possible way to perform collective operations like broadcast. In three simple steps, all processes receive the message broadcast from process 0. A communication tree as a special directed acyclic graph is one way (but not the only way) to implement most of the collective operations defined in MPI. Alternative communication patterns for collective operations can build cycles, e.g. ring communication for the allgather communication [1].

![Figure 2: A simple binomial tree for scheduling collective communication](image)

- **Communication performance models** are not a requirement for designing parallel programs, but are important for understanding and improving their performance. The simplest communication model which expresses the main properties of communication is the Hockney model [17]. The main properties of this model are:
  - the latency $\alpha$ - a constant delay to transfer first byte from sender to receiver
  - bandwidth $\beta$ - time to transfer a byte from sender to receiver

According to the model, a message of $n$ bytes is then transferred in time

$$T(n) = \alpha + n \times \beta$$
A good example of using such models to estimate collective communication algorithms is [1]. The paper uses models for estimating the complexity of various collective algorithms. For example, let us consider the collective communication operation broadcast. A root process distributes data of size $n$ to all $p$ participating processes. A commonly used algorithm is the binary tree algorithm: in the first step, the root sends data to process $(root + p/2)$. This process and the root then act as new roots within their own subtrees and recursively continue. The communication takes a total of $\log(p)$ steps, and at any step $n$ data is communicated. Therefore, the time taken by this algorithm is $\alpha n^2 \beta$. The paper demonstrates that the question of predicting the runtime of even a simple algorithm is not trivial and both message size and number of participating processes play an important role in the estimation. If the complexity of different algorithms differs in the order of magnitude (mathematically a difference in complexity of algorithms in their $O$-notation [12]) then a model is sufficient for choosing the more efficient algorithm. But sometimes a model delivers the same complexity for different algorithms. In this case experiments are the only option to determine the best algorithm to use. The paper was presented in the Research Seminar Series.

- The area of heterogeneous high performance computing is a specific area of high performance computing which explores the ever increasing heterogeneity of HPC systems. We observe two main directions of heterogeneity – on the one hand, there are different processing/memory properties on the nodes. On the other hand, communication media differ in their properties as well. An example of a heterogeneous system is a supercomputer with both multi-core and accelerator processing units. In 2011, 20 of the supercomputers in TOP500 are built this way, with numbers of systems with accelerators expected to increase. On a lower scale, affordable clusters of computers connected to build a larger infrastructure also show a very high heterogeneity. Modern grid infrastructures for example are an important platform for scientists in different disciplines around the world (see Grid’5000 [13]).

- Heterogeneous communication performance models: While heterogeneity of HPC systems grows, many communication algorithms (e.g. the collective communication algorithms presented in [1]) are designed for homogeneous communication networks. When such algorithms are applied to a heterogeneous network of computers, they are often inefficient. Heterogeneous communication performance models capture the differing properties of a heterogeneous communication network. Otherwise, a model might not be accurate. There are two main approaches to address heterogeneity. We can either use existing homogeneous models and apply them to the heterogeneous network or we can use communication performance models designed with heterogeneity in mind:

Figure 3: Visualization of the Hockney model
• Heterogeneous communication performance models can easily be derived from the homogeneous models by modelling each link separately. For example, we can use the simple Hockney model to get the model parameters separately for every link of a heterogeneous network. Thus, we extend the Hockney model to a heterogeneous Hockney model. The main issue with this approach is that prediction of collective operations has not been accurate [3]. Most likely, this is because a simple model aggregates different contributions from network and processor which need to be separated for predictions of collectives.

• A heterogeneous communication performance model called LMO [9] has been created by our Heterogeneous Computing Laboratory. It creates model parameters for every link, and also separates the different contributions from network and processor. The main challenge in this case is to estimate accurately the increasing number of model parameters in a set of experiments.

Problem statement

In the area of high-performance computing, the general problem is how to optimize parallel applications on the available hardware platforms. There is an enormous body of research in the area of such optimizations. We focus on a particular area of optimizations:

My work is on optimizations in the communication of HPC applications, particularly optimizations derived from observations on the heterogeneity of the underlying communication network.

With this focus in mind, the related work in optimizations in the high performance computing domain becomes easier to describe. We list the two main types of optimizations we have observed in the area:

1. Collective communication algorithms can schedule intelligent communication based on network heterogeneity:
   - Explicit meta-computer configuration: E.g. If two clusters are connected through a wide-area link, and all processes running on the clusters are involved in a collective communication, it is reasonable to use the wide-area link as little as possible, since it yields higher latency and lower bandwidth. [14] and [15] are projects that demonstrate this approach with significant impact on performance in some cases.
   - Model-based algorithm: more sophisticated communication can be done based on communication models providing the properties of the network. This can be either done by just remapping MPI processes along a fixed tree or by dynamically determining the structure of a communication tree ([18], [19]).

2. Prediction-based optimizations of collective communication: If heterogeneous communication models are accurate, then different algorithms can be compared through model predictions and the optimal algorithms for collective communication can be chosen. However, the requirement for model accuracy is usually very difficult to meet. We present [3] and [4] as they both address the area of prediction-based optimizations.
is a closely-related dissertation on MPI collective operations and how they can be used in an efficient way in high-performance computing. The main goal of the work is choosing an optimal algorithm for any collective MPI operations for all message sizes and communicator sizes. The work uses so called decision maps as a decision tool. Three main strategies are proposed to construct a decision map (Figure 4):

- **Parallel communication models:** The analytical model-based approach is simple – predictions can be made efficiently (during runtime), and based on these predictions the best communication algorithm can be chosen for an algorithm. However, real communication is not always accurately predicted by models. In particular, the simple Hockney model often provides inaccurate results when used with collective operations. The PLogP model [20] shows more accurate predictions for non-linear behaviour of communication, but is more expensive to use.

- **Graphics Encoding Schemas:** Quadtrees are used to efficiently store and look up the recommended algorithms for a particular process number and message size based on experiments. Experiments can be “saved” by creating a coarser quadtree, which in turn could reduce the accuracy of decision making.

- **Statistical Learning Methods:** C4.5 decision trees are used as an alternative to quadtrees for decision making on algorithms. The decision is also made based on experimental results. This method sometimes offers more flexibility than quadtrees.

![Figure 4: Decision function construction as proposed in [3]](image)

The work clearly shows the strengths and weaknesses of using analytical or experimental data for decision making. The analytical approach is elegant and efficient, but tends to be inaccurate in many cases. The experimental approach can be more accurate but is often very expensive, and becomes even infeasible when a very large decision map needs to be constructed.

[4] is another closely related dissertation on heterogeneous communication performance models. The work focuses on the model-based approach described above. It uses the heterogeneous LMO model presented in [9] which addresses the main challenges connected with all new analytical
models. First, the model parameters need to be estimated in an efficient and reliable way. Second, the accuracy of the model needs to be examined. The main strength of the LMO model is that it separates processing and networking contributions in different parameters. This allows more flexibility when predicting collective communication. However, [4] also observes a number of issues when benchmarking different communication operations. For example, there is a nonlinear behaviour for some collective operations and some message sizes on standard clusters. For example, figure 5 shows that scatter and gather communications sometimes need to be divided into different ranges of message sizes to solve the nonlinearity issue. The work uses robust and flexible software packages for benchmarking and optimizing MPI communication based on different models.

![Figure 5: Observations on scatter and gather challenging a unified model for all message sizes [4]](image)

2. Description of proposed solution

In the previous section, the problem statement and the two main approaches in related work were described. In both of these approaches, valuable contributions can be made:

- In the area of heterogeneity-aware communication algorithms, the existing work has not reached a mature level in following aspects:
  - [14] and [15] do not use models to determine the properties of the communication network. A meta-computer with fast or slow links is specified manually. The process can be improved significantly through the use of communication models to determine the properties of the communication network automatically.
  - Another shortcoming of these works is that they only differentiate between two types of links – slow and fast links – when constructing a communication tree. Communication trees can be much more varied and a more flexible communication algorithm can potentially construct better communication trees based on the wide variation of communication properties in a network.

- In the area of prediction-based optimizations the main challenge is creating an accurate model for collective communication, and [3] and [4] demonstrate the complexity of this issue. We will attempt to address this topic in a number of ways.
o Identify the source of discrepancy between communication performance and model predictions. This complex process involves investigating the hardware, the communication protocols, and the communication models.

o Design an alternative model for a particular network which allows for more accurate prediction of collective communication.

o Find a new formulation of existing communication model predictions which more accurately predict collective communication.

The first approach can be easier than the second approach. For example, the presented algorithms in [14] intuitively reduce communication over slow links. This is an optimization of an algorithm which is valid independent of the used platform. On the other hand, the second approach requires a profound understanding of the communication network – accurate predictions of the communication suggest that we fully understand how it performs on them used platform.

3. Summary of progress to date

Platforms

The main platforms available for my work can be divided into two categories:

- Development/Debugging: the HCL cluster and the CSI cluster csicluster.
- Benchmarks/Experiments: on the Grid’5000 infrastructure, which provided sufficient network heterogeneity to test and verify our work on heterogeneous communication models; the shamrock cluster at IBM Dublin, which allows experiments using heterogeneous media as well

1. Introduction into the area

My first task was to get familiar with collective communication in MPI and how heterogeneous communication performance models can be used to predict collective communication. In particular:

- Benchmarks were made on various collectives on the HCL cluster and the Grid’5000 infrastructure.
- The tools MPIBlib/CPM [23][24] were explored and tested. They are developed by our lab and implement functionality in benchmarking and communication performance models respectively. They will be used in my future work as well.

2. Model-based collective communication trees – modifications and experimental results

A communication tree is a central part of the collective communication algorithms we use as described in section 1. In the used implementation, each collective algorithm creates a communication tree. After the communication tree is generated, the collective communication is scheduled as a sequence of point-to-point operations along the tree.
An algorithm by J. Traeff [2] exists for the collective operations irregular scatter and gather for homogeneous networks. Irregular means that each process can send or receive a message of different size and content to or from the root. Homogeneous means that Traeff does not consider the possibility of the links between processes having different network properties. A natural extension of this algorithm was implemented which takes into account the properties of the links besides the message sizes with the help of communication models.

The communication models were also used to extend binomial tree algorithms. The binomial tree algorithms are simpler than arbitrary algorithms on trees, because in this case the tree structure is predefined by the definition of a binomial tree. Only the mapping “nodes-to-processes” needs to be established. For these algorithms, a number of variations was implemented. Since these variations are orthogonal, they build a whole set of new variations which is the product of variation in each dimension. The various orthogonal dimensions can be summarized as follows:

\[
\text{\textless MODEL\textgreater} \times \text{\textless OPERATION\textgreater} \times \text{\textless TRAVERSAL ORDER\textgreater} \times \text{\textless MAPPING STRATEGY\textgreater}
\]

- Models: Hockney, LMO, PLogP
- Operations: Scatter(v), Gather(v), Reduce, Broadcast
- Traversal orders: depth-first, breadth-first, an order following the depth of each binomial subtree
- Mapping strategies: minimum-first, maximum-first in relation to the prediction function

The operations are coupled with an order of traversing a tree and a strategy used when traversing the tree in the order specified.

We will give an example: the operation \texttt{Hockney_Scatter_dfs_binomial_min} uses parameters from the Hockney model for each link between any two nodes, which means every link has a latency \( \alpha \) and a bandwidth \( \beta \). The traversal order is depth-first traversal. The chosen strategy is minimum-first.

Starting from a fixed root and a predefined binomial tree structure, the mapping of processes to the nodes of the tree will be dynamically done. The node which has minimum Hockney-estimated communication time to the root for the given message size will be attached as a child of the root. Then the same will be done with the rest of the nodes recursively since the approach is depth-first.

We repeat this until the binomial tree has a process number assigned to each tree node. After the tree has a complete node-process mapping, we perform a scatter operation along the entire tree until all processes have received their personalized message.

After implementing these variations of collectives through different communication trees, we performed a number of experiments both on the HCL cluster and on Grid’5000. As a first observation, we found that our modifications do not improve the performance on a homogeneous network as was the network of the HCL cluster. However, for a heterogeneous network of computers as Grid’5000, a model-based approach for constructing a tree is particularly beneficial. We ran various multi-site communication experiments for the collective operations scatter and gather. We found that the most complex algorithm we used – the modified Traeff algorithm – is not necessarily better than some of the simpler modified binomial tree algorithms. In particular the
binomial depth-first minimum-based irregular scatter/gather operations using the simple Hockney model performed excellent. We found that this particular algorithm clusters together the nodes with faster communication links in a binomial communication tree. In most cases this means that the physically closer machines (e.g. inside a cluster) will also be clustered together inside a communication tree. In this way there is only one message propagation across different sites in the case two sites are involved. If our modified algorithm is included in the MPI middleware, it would be very useful for complex multi-site runs. Collective communication for such a setup would be optimized without the need of a manual configuration of a large meta-computer. Currently, most improvements of such multi-site MPI experiments are based on the fact that users explicitly know and specify the properties of the meta-computer. The communication middleware itself does not currently support automatic optimizations.

![Graph](image.png)

**Figure 6: Impact of model-based algorithms on irregular scatter and gather across sites on Grid’5000 infrastructure**

We published and presented our findings in [5].

3. Hierarchical extension of the communication performance models.

The reason hierarchy is introduced is usually to simplify things. In communication models, the hierarchy of a communication network (like Grid’5000) can be introduced for different simplifications, for example:

- Reduce the complexity of model parameter estimation. Measuring the model parameters for N processes can take N^2 time, and a hierarchy can significantly reduce this overhead by classifying the links into very few different categories.

- For model predictions, hierarchy can be used for sophisticated predictions. As an example, a graph can be used to represent the hierarchy of a network. A number of attributes can be stored, and each attribute can represent local properties of the network. An accurate prediction for the communication between two distant nodes can therefore take into account many parameters along the way connecting the two nodes in the hierarchical representation.

So far, we have implemented the first approach which greatly increases the efficiency of estimating model parameters on Grid’5000. This was important particularly for experiments launching as many
processes per node as the cores available. We created just a few categories of links. E.g. for a setup of 8 nodes per cluster on different sites, we assume that each intra-node communication on one site is equivalent. Therefore we can perform a single experiment per cluster instead of “each-with-each-other” which is 36.

4. Linear interpolation instead of models for non-linear MPI point-to-point times.

An important challenge during the estimation of model parameters was the observation that for Grid’5000 cross-site MPI point-to-point communication is not linear in respect to the message size (see Figure 7). This means that a precise estimation is impossible with a linear model like the Hockney model. For example, the PLogP model allows for piecewise linear model estimation. Instead, we tried a more simple approach which does not use a model and runs a number of benchmarks for a lot of message sizes. In this case, the prediction function simply looks up the results and linearly interpolates the runtime based entirely on the measurements. The advantage of this approach is that it can be accurate even when the execution time is any non-linear function. The disadvantage is that to be precise, a very large number of benchmarks have to be executed for a link. However, this approach is still feasible if such a benchmark is coupled with a hierarchical approach as described in the previous section. Experiments showed that the communication time of collectives can be predicted more precisely, particularly non-linearities like the step function in the communication time at certain message sizes.

![Figure 7: Typical MPI point-to-point benchmark across sites](image)

5. Improving the bandwidth of cross-site optical fibre connections.

During various experiments on collective operations, we noticed a phenomenon of cross-site optical fibre links on Grid’5000: using multiple MPI point-to-point communications along these links in parallel proved more efficient than using a single connection. As an example, transferring a 1 MB message using MPI across two sites – one in Bordeaux, one in Grenoble - is faster if the message is split into 16 fragments and each fragment is sent with parallel MPI calls.

Apart from being unexpected, this phenomenon strongly affected our benchmarks and the correctness of our predictions. For example, some linear communication was surprisingly fast compared to some of the binomial tree communication. We explain this with the fact that any binomial tree communication introduces sequentiality in the way messages are propagated along the binomial tree, other than non-blocking linear communication. This observation was surprising for MPI, since on a local area network such behaviour is not observed. However, similar improvements have been observed in the domain of distributed computing [21].
We further researched this topic and confirmed that this “split and send in parallel” approach significantly improves the bandwidth for large messages across sites on Grid’5000. We have no explanation of the underlying reasons for that but we suspect the reason is either in local cluster properties (bonding of many cables allowing parallel message transfer until a packet leaves a site) or in properties of the optical fibre connecting the various Grid’5000 sites. We found that the approach is not beneficial on a local site or cluster level.

We published and presented our results in [6].

4. Plan for the next 6 months

A model of serial and parallel components under resource sharing conditions

In [6] we observed an interesting phenomenon: in a complex communication network packets can traverse a series of different components, e.g. including Ethernet and optic fibre. If we transfer packets simultaneously, i.e. the resources are shared among many connections, different hardware components might behave differently. Either these packets are transferred in parallel, or they are transferred in serial. A combination of these seems to be the case in the very complex Grid’5000 infrastructure. To better understand if our assumption is correct, we performed a number of experiments on the shamrock cluster at IBM Dublin. Since the cluster connects Ethernet clusters with an optic fibre link, it has some of the components of Grid’5000. Indeed, we verified with various TCP and MPI benchmarks that Ethernet switches can receive packets in parallel, while the optic fibre connection is serial. While this observation is not new, we believe there is no communication model which describes serial and parallel components under resource sharing conditions, their properties and the properties of the communication chain they build.

The next 6 months will further explore if a simple and accurate model can be created to describe the behaviour of hierarchical networks under resource sharing conditions. Our experiments on

Figure 8: Comparison of MPI point-to-point communication and the modified version for messages in the range 100 Kbytes-1 MB and 1 MB-10MB
Grid'5000 and the shamrock cluster suggest that parallelization occurs at the endpoints of a complex communication chain, and serialization occurs in the middle phase. A similar direction of research is also explored recently by following papers:

- [7] explores independently how a LogP-based model can be created for such resource-sharing conditions and for small messages on Infiniband clusters.
- [8] paper describes how contention can be modelled for such resource-sharing conditions on Infiniband clusters.

We believe our work differs in following aspects:

- Currently, our model targets transfer of large data volumes. This is a simplification, because latency-related factors are not important in this case
- Our model is very simple – there are simple serial and parallel terms behind each component, and they are combined as needed
- Our model targets more complex chains of communication components than previously described. They might include but are not limited to Infiniband, Ethernet, optic fiber etc.

There is a dynamics in the network e.g. when the number of communicating channels changes during communication. This should be considered and the idea of presenting a dynamic communication as a set of static states might be useful in this case.

**Risks of the proposed research plan:**

- It is not clear if the model parameters can be accurately estimated
- It is not clear if the proposed model will be accurate in predicting communication
- A number of assumptions is made in our model which hold for the experiments done so far but needs to be checked each time:
  - Fairness – every communicating channels steals an equal piece of the available bandwidth
  - The assigned bandwidth will remain stable throughout the communication as long as large data volumes are transferred

5. **References**

11. http://www.top500.org/ - Top500 website
15. Edgar Gabriel, Michael Resch, Thomas Beisel and Rainer Keller. Distributed computing in a heterogeneous computing environment
23. http://hcl.ucd.ie/project/cpm - CPM project