Future Generation Computer Systems 61 (2016) 66–82

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Extending τ-Lop to model concurrent MPI communications in multicore clusters

Juan-Antonio Rico-Gallego [a,*], Juan-Carlos Díaz-Martín [a], Alexey L. Lastovetsky [b]

[a] University of Extremadura, Avd. Universidad s/n, 10003, Cáceres, Spain
[b] University College Dublin, Belfield, Dublin 4, Ireland

HIGHLIGHTS

- We present an extension of the τ-Lop performance model for multicore clusters.
- The τ-Lop goal is to help in the design and optimization of parallel algorithms.
- It is applied to collective algorithms in mainstream MPI implementations.
- The τ-Lop model is compared to other well known and established models.
- A methodology is described for the measure of the parameters of the model.

ARTICLE INFO

Article history:
Received 30 September 2015
Received in revised form
27 January 2016
Accepted 27 February 2016
Available online 11 March 2016

MSC:
00-01
99-00

Keywords:
Parallel performance models
Parallel algorithms
Message passing interface
Performance analysis
Multicore clusters

ABSTRACT

Achieving optimal performance of MPI applications on current multi-core architectures, composed of multiple shared communication channels and deep memory hierarchies, is not trivial. Formal analysis using *parallel performance models* allows one to depict the underlying behavior of the algorithms and their communication complexities, with the aims of estimating their cost and improving their performance.

LogGP model was initially conceived to predict the cost of algorithms in mono-processor clusters based on point-to-point transmissions with network latency and bandwidth based parameters. It remains as the representative model, with multiple extensions for handling high performance networks, covering particular contention cases, channels hierarchies or protocol costs. These very specific branches lead LogGP to partially lose its initial abstract modeling purpose.

More recent $\log_n$P represents a point-to-point transmission as a sequence of *implicit transfers* or data movements. Nevertheless, similar to LogGP, it models an algorithm in a parallel architecture as a sequence of message transmissions, an approach inefficient to model algorithms more advanced than simple tree-based one, as we will show in this work.

In this paper, τ–Lop model is extended to multi-core clusters and compared to previous models. It demonstrates the ability to predict the cost of advanced algorithms and mechanisms used by mainstream MPI implementations, such as MPICH or Open MPI, with high accuracy. τ–Lop is based on the concept of *concurrent transfers*, and applies it to meaningfully represent the behavior of parallel algorithms in complex platforms with hierarchical shared communication channels, taking into account the effects of contention and deployment of processes on the processors. In addition, an exhaustive and reproducible methodology for measuring the parameters of the model is described.

1. Introduction

Modern high performance computing systems are composed of nodes with a significant number of cores per node and deep memory hierarchies, connected by high performance networks. Scientific applications face the challenge of obtaining as much performance as possible from such complex platforms. MPI [1] is the de facto standard that defines the communications interface used by this type of applications. The MPI execution model is based on processes communicating by message passing, including point-to-point and collective operations, which involve a group of processes. Frequently used [2], collective operations are a key issue in achieving performance and scalability in parallel applications.

* Corresponding author.
E-mail addresses: jarico@unex.es (J.-A. Rico-Gallego), juancarl@unex.es (J.-C. Díaz-Martín), alexey.lastovetsky@ucd.ie (A.L. Lastovetsky).

http://dx.doi.org/10.1016/j.future.2016.02.021
0167-739X/

A collective operation can be implemented by algorithms from a preset menu, with one of them chosen at runtime based on the value of parameters such as message length or group size. In general, an algorithm is executed as a sequence of steps, and defines a point-to-point communication graph between the processes of the group. Algorithms are designed with the goals of minimizing the number of such steps, optimizing the use of the underlying communication channels available in the system, or fitting a particular network technology or topology. Besides, achieving these goals in modern complex multi-core clusters requires to consider the hierarchy of communication channels (imposed by the difference in channels performance), the implications of process distribution over the cores of the platform (virtual topology), and the contention effects due to the use of shared communication resources.

Parallel performance models provide a theoretical framework for analytic representation of the communications and their associated costs, based on system parameters. Important features including advanced transmission mechanisms (such as RDMA and OS bypass), middleware related costs, or network technology, should be captured by an accurate and scalable cost estimation model.

The type of parameters divides the known models in two broad groups: hardware and software. Hardware models, such as *Hockney* [3] or *LogP* [4], represent communication costs with hardware related parameters, such as network latency or bandwidth. They were initially created for homogeneous mono-processor clusters, and the increasing complexity of modern platforms limits their accuracy. In addition, they show weakness in representing different mechanisms provided by the software such as communication protocols, or in estimating the impact of the communication middleware on the communication cost. These issues are partially addressed by adding new parameters for such protocols in *LogGPS* [5] or hierarchical communications in *LogGPH* [6]. By contrast, software models, such as $\log_n$P [7,8], abstract the hardware complexities by the adoption of parameters related to the communication middleware, with the drawback of a possible loss of network technology details.

τ-Lop [9] is a software parametrized parallel model aimed to represent parallel algorithms and accurately and scalably predict their cost. It provides a simple and powerful abstraction by considering a message transmission as a sequence of transfers. The transfers reflect data movements between hardware or software agents, and are modeled based on the underlying architecture characteristics. The decomposition of a message transmission into a sequence of transfers allows for an incremental analysis of the algorithm, from a high level representation to low platform-specific details. For instance, a point-to-point message transmission could be used as the building block to model a collective operation, but deeper insight can be obtained by further considering the transmission as a sequence of data transfers.

The decomposition of a point-to-point message transmission into transfers is not a new concept [10]. τ-Lop goes beyond in natively representing the concurrency in the access to the channel when it is shared by parallel transfers. This contention effect has a significant impact on the algorithm performance, so that its consideration in the model results in a substantial improvement in the accuracy of the predicted cost.

The contribution of this paper is the extension of τ-Lop, which was initially developed to model the behavior of collective algorithms in shared-memory compute nodes, to modern multi-core clusters.

Furthermore, the extended τ-Lop model addresses the influence on the cost of the deployment of processes over the system processors. In a multi-core cluster with hierarchical organization of communication channels, the mapping of processes can improve or aggravate the contention effect. τ-Lop takes into account the way the virtual topology defined by the algorithm is mapped onto the physical topology of the machine. This ability provides a more realistic representation of the algorithm, leading to a better cost prediction.

The rest of the paper is structured as follows. Section 2 revisits the state-of-the-art models, with emphasis on LogGPH and $m\log_n P$, later used in cost evaluations. Section 3 describes the τ-Lop model and its extensions to cover multi-core clusters. Section 4 discusses its application to key algorithms of MPI collectives in multi-core clusters and demonstrates its potential as an analytical inference model for prediction of the communication cost. Section 5 addresses the communication modeling of a whole application, a parallel matrix multiplication kernel on a multi-core cluster. Section 6 gives details of the experimental approach used to estimate the parameters of the model, and Section 7 concludes the paper. An Appendix is included to describe the approach to report the error measurements.

2. Related work

Most of the current parallel performance models derive from Hockney [3] and Culler et al. LogP [4]. Both are linear models, in the sense that they represent the point-to-point communication cost by a linear function of the size of the message. Hockney represents the cost as $T = \alpha + m\,\beta$, where m is the size of the message, α is the network latency, and β is the inverse of the network bandwidth. The execution time of a parallel algorithm in the cluster is formulated in terms of these parameters. Together with benchmarks measurements, this simple model has been used for comparison of different MPI collective algorithms in a given system, for ranges of message sizes and numbers of involved processes [11,12].

The more advanced LogP cost model splits the network latency in the network delay (L) and the network overhead (o). The overhead is the time invested by the processor in sending and receiving a message. This separation of processor and network contribution allows one to model the computation and communication overlap. LogP also includes the minimum time interval between message transmissions as g.

Alexandrov et al. *LogGP* model [13] extends LogP with an additional parameter, G, that captures the network bandwidth for long messages. It represents the cost of a single point-to-point message transmission as:

$$T_{LogGP} = L + 2\,o + (m-1)\,G.$$

Several models drawn from LogGP contribute with a variety of add-ons to represent different aspects of the communication in complex HPC platforms. LogGPS [5] covers aspects of synchronization, providing the rendezvous cost as a new parameter, LogGPH [6] supports representation for hierarchical architectures through a different set of parameter values for each communication channel, LoGPC [14] adds contention time in mesh networks, and LogGPO [15] accurately captures the overlap of communication and computation in non-blocking transmissions.

Linear models have proven their ability to model point-to-point message transmissions, as shown by Al-Tawil et al. in [16], and upon them, the message passing algorithms underlying the collective primitives of the MPI standard on mono-processor clusters [17,18]. Analytical modeling of a broad range of these algorithms using the Hockney model can be found in [12,19] and [11]. An optimal broadcast algorithm is developed in [20], and in [21] for heterogeneous networks, the *gather* collective is addressed in [22] and [23], algorithms for *reduction* operations are developed in [24], and for the *Alltoall* collective in [25].