Computers and Electrical Engineering 51 (2016) 168–183

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Characterization and modeling of multicast communication in cache-coherent manycore processors☆

Sergi Abadal[a,*], Raúl Martínez[b,1], Josep Solé-Pareta[a], Eduard Alarcón[a], Albert Cabellos-Aparicio[a]

[a] NaNoNetworking Center in Catalonia (N3Cat), Universitat Politècnica de Catalunya, Barcelona, Spain
[b] Oracle Labs, Oracle Corporation, Vancouver, BC, Canada

ARTICLE INFO

Article history:
Received 10 May 2015
Revised 23 December 2015
Accepted 23 December 2015
Available online 21 January 2016

Keywords:
Manycore processors
Multicast
Broadcast
On-chip traffic analysis
Network-on-chip
Scalability

ABSTRACT

The scalability of Network-on-Chip (NoC) designs has become a rising concern as we enter the manycore era. Multicast support represents a particular yet relevant case within this context, mainly due to the poor performance of NoCs in the presence of this type of traffic. Multicast techniques are typically evaluated using synthetic traffic or within a full system, which is either simplistic or costly, given the lack of realistic traffic models that distinguish between unicast and multicast flows. To bridge this gap, this paper presents a trace-based multicast traffic characterization, which explores the scaling trends of aspects such as the multicast intensity or the spatiotemporal injection distribution for different coherence schemes. This analysis is the basis upon which the concept of *multicast source prediction* is proposed, and upon which a multicast traffic model is built. Both aspects pave the way for the development and accurate evaluation of advanced NoCs in the context of manycore computing.

1. Introduction

In the ever-changing world of microprocessor design, multicore architectures are currently the dominant trend for both conventional and high-performance computing. Chip Multiprocessors (CMPs) resulting from the interconnection of several processing cores were conceived to overcome the complexity and power scalability hurdles of processors with a single CPU; however, the scalability concerns have now migrated to facets such as memory management, programmability or the limits of parallelism as the core count increases.

Inherent parallelism limits aside, these scalability concerns are generally dependent on the architecture or programming model of choice. A long-running debate has brought up strong arguments for the adoption of two widely-known models in manycore CMPs: shared memory and message passing. Shared memory provides remarkable programmability and compatibility with legacy code. However, its scalability is arguably limited by performance and architectural complexity issues related to data consistency. On the contrary, message passing offers unique validation and hand-tuned performance benefits, which come at the cost of placing an increasingly heavy burden upon the programmer. The differences between these

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. M. Daneshtalab.

* Corresponding author. Tel.: +34 662086489 .
E-mail address: abadal@ac.upc.edu (S. Abadal).

[1] Raúl Martínez was working for INTEL at the INTEL Barcelona Research Center when the main ideas of the paper were developed.

http://dx.doi.org/10.1016/j.compeleceng.2015.12.018
0045-7906/

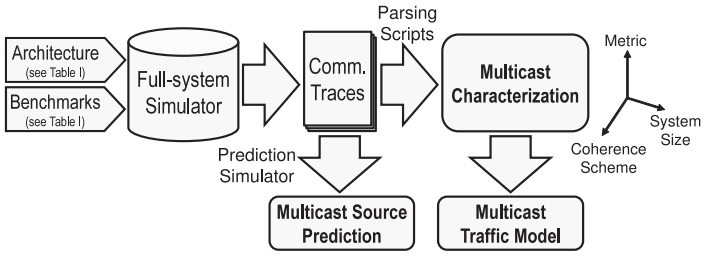

Fig. 1. Simulation-based multicast traffic characterization and modeling.

two extremes contrast with one common point: most of the scalability issues are tightly coupled to on-chip communication limitations. Due to this, the research focus in multiprocessors has gradually shifted from how cores compute to how cores communicate.

The limited scalability of conventional Networks-on-Chip (NoCs) in the presence of global and multicast traffic represents an important constraint to multiprocessor architects and programmers. In shared-memory multiprocessors, cache coherence is the main source of on-chip communication and is generally maintained through directory-based protocols that limit the use of multicast to the invalidation of cache blocks on a shared write. This reduces the injection of multicast traffic, yet at the cost of non-scalable area and energy overheads required to track the sharers of the data. In message passing systems, where communication is explicitly set by the programmer, the use of collective communication routines such as `MPI_Bcast` or `MPI_Allgather` is often avoided. This, however, may lower the maximum achievable performance and increase the complexity of parallel programming over message passing even further. Also, the lack of proper multicast support hampers the development of novel programming models and manycore systems that may be multicast-intensive [1].

Given that multicast and broadcast may become a critical factor guiding the design of future manycore CMPs, there is a need to understand how the characteristics of such traffic will scale with the number of cores. Providing accurate multicast traffic characterization in different scenarios would be useful for the early-stage design and evaluation of NoCs in general and multicast mechanisms in particular. However, to the best of the authors' knowledge, no tools are available for the analysis and modeling of multicast traffic. Different works have characterized or modeled inter-processor communication in moderately-sized shared memory processors (see [2] and references therein), as well as in message passing clusters or supercomputers [3–5]. However, none of them has analyzed how collective communication scales for a wide set of representative applications and architectures at the CMP scale. Also, existing traffic models do not differentiate between unicast and multicast flows and only offer data for a given system size.

In this paper, we aim to address these issues in one of the contending programming models (shared memory) by performing a scalability-oriented multicast traffic characterization. Our contribution and methodology is summarized in Fig. 1. We first analyze traces of widely known SPLASH-2 and PARSEC benchmark applications running over a set of existing cache coherence alternatives and for different processor sizes. Since scarce data is available beyond 64 cores, we additionally perform an exploratory extension of the study up to 512 cores whenever possible. Although most of these applications were not build to scale to thousands of processors [6,7], the analysis of its multicast traffic patterns may be still useful to extrapolate the behavior of future scalable shared-memory applications. We later employ the results of the characterization process to (1) propose and evaluate the potential of *multicast source prediction*, and (2) create a synthetic traffic generator that faithfully models multicast communication and that can be used to generate realistic mixed traffic profiles. With this, we aim to trigger further research in the area of multicast support for manycore systems.

The paper is a direct extension of previous work by the authors [2]. The original contribution is augmented as follows:

- The analysis now includes results regarding Token Coherence [9], as well as directory-based coherence with imprecise tracking up to 512-core CMPs.
- The concept of *multicast source prediction* is presented and evaluated, placing emphasis on its potential applicability at the NoC design level.
- The characterization results are used to implement a multicast traffic model and propose an appropriate synthetic traffic generator.

The remainder of this paper is as follows. In Section 2, we provide background on cache coherence and on-chip networking which may further motivate this work and be useful to understand its results. Within the same section, we present related work in NoC traffic analysis and modeling. Section 3 details the characterization methodology used to obtain multicast communication traces and to analyze them. The results of the multicast traffic characterization are presented in Section 4, which are later used in Section 6 to create and validate a multicast traffic model. Section 7 concludes the paper.

2. Background and related work

This section seeks to further motivate this work by explaining why it is necessary to analyze unicast and multicast traffic separately (Section 2.1); by providing background on the relation between cache coherence and the characteristics of the