Chapter 3
Design and Optimizations of Lattice Boltzmann Methods for Massively Parallel GPU-Based Clusters

Enrico Calore
University of Ferrara, Italy & National Institute for Nuclear Physics, Italy

Alessandro Gabbana
University of Ferrara, Italy & National Institute for Nuclear Physics, Italy

Sebastiano Fabio Schifano
University of Ferrara, Italy & National Institute for Nuclear Physics, Italy

Raffaele Tripiccione
University of Ferrara, Italy & National Institute for Nuclear Physics, Italy

ABSTRACT

GPUs deliver higher performance than traditional processors, offering remarkable energy efficiency, and are quickly becoming very popular processors for HPC applications. Still, writing efficient and scalable programs for GPUs is not an easy task as codes must adapt to increasingly parallel architecture features. In this chapter, the authors describe in full detail design and implementation strategies for lattice Boltzmann (LB) codes able to meet these goals. Most of the discussion uses a state-of-the-art thermal lattice Boltzmann method in 2D, but all lessons learned in this particular case can be immediately extended to most LB and other scientific applications. The authors describe the structure of the code, discussing in detail several key design choices that were guided by theoretical models of performance and experimental benchmarks, having in mind both single-GPU codes and massively parallel implementations on commodity clusters of GPUs. The authors then present and analyze performances on several recent GPU architectures, including data on energy optimization.

DOI: 10.4018/978-1-5225-4760-0.ch003
INTRODUCTION AND BACKGROUND

The Lattice Boltzmann Method (LBM) is widely used in computational fluid-dynamics, to describe fluid flows. This class of applications, discrete in time and momenta, and living on a discrete and regular grid of points, offers a large amount of available parallelism, making LBM an ideal target for recent multi- and many-core processor-based clusters. LBM is an interesting simulation tool, not only in basic sciences but also for engineering and industrial applications. It is also popular in the energy sector, and is widely used in the oil&gas industry to model porous, multi-phase or otherwise complex flows, in order to better understand the dynamics of oil and shale-gas reservoirs and to maximize their yield. Recent developments have also started to tackle relativistic fluid dynamics, further extending the potential application domain not only to astrophysics problems or elementary particle physics applications, but also to the study of exotic materials, graphene for instance, were the peculiar behavior of the electrons moving in the material lattice can be treated within a formalism similar to relativistic hydrodynamics.

High Performance Computing (HPC) has seen in recent years an increasingly large role played by Graphics Processing Units (GPUs), offering significantly larger performance than traditional processors. In GPUs many slim processing units perform in parallel thousands of operations on a correspondingly large number of operands. This architectural structure nicely adapts to algorithms with a large amount of available parallelism, as in these cases it is possible to identify and concurrently schedule many operations on data items that have no dependencies among them. This is very often the case for so-called stencil codes typically used to model systems defined on regular lattices. They process data elements associated to each lattice site applying some regular sequence of mathematical operations to data belonging to a fixed pattern of neighboring cells. General implementation and optimization of stencils on GPUs has been extensively studied by many authors, (Holewinski, Pouchet, & Sadayappan, 2012), (Maruyama & Aoki, 2014), (Vizitiu, Itu, Lazar, & Suciu, 2014), (Vizitiu, Itu, Niţă, & Suciu, 2014). This approach is appropriate for several computational Grand Challenge applications, such as Lattice QCD (LQCD) and LBM, for which a large effort has gone in the past in porting and optimizing codes and libraries for both custom and commodity HPC computing systems (Bernard et al., 2002; Bilardi, Pietracaprina, Pucci, Schifano & Tripiccione, 2005). More recently many efforts have been focused on GPUs (Bernaschi, Fatica, Melchionna, Succi, & Kaxiras, 2010; Bonati et al., 2017; Bonati, Cossu, D’Elia, & Incardona, 2012; Pedro Valero-Lara, 2014; Pedro Valero-Lara et al., 2015; Januszewski & Kostur, 2014; Tölke, 2008). These efforts have allowed to obtain significant performance levels, at least on one or just a small number of GPUs (Bailey, Myre, Walsh, Lilja, ...
Code Clone Detection and Analysis in Open Source Applications
www.igi-global.com/chapter/code-clone-detection-and-analysis-in-open-source-applications/192915?camid=4v1a

Exceptions in Ontologies: A Theoretical Model for Deducing Properties from Topological Axioms
www.igi-global.com/chapter/exceptions-ontologies-theoretical-model-deducing/62435?camid=4v1a