Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **8**, No. 6, pp. 971-991

DOI: 10.4208/aamm.2015.m1138
December 2016

Toward Cost-Effective Reservoir Simulation Solvers on GPUs

Zheng Li[1,*], Shuhong Wu[2], Jinchao Xu[3] and Chensong Zhang[4]

[1] *Kunming University of Science and Technology, Kunming 650093, China*
[2] *Research Institute of Petroleum Exploration and Development, CNPC, Beijing 100083, China*
[3] *Department of Mathematics, Penn State University, University Park, PA 16802, USA*
[4] *LSEC & NCMIS, Academy of Mathematics and Systems Science, Beijing 100190, China*

Received 5 June 2015; Accepted (in revised version) 13 October 2015

Abstract. In this paper, we focus on graphical processing unit (GPU) and discuss how its architecture affects the choice of algorithm and implementation of fully-implicit petroleum reservoir simulation. In order to obtain satisfactory performance on new many-core architectures such as GPUs, the simulator developers must know a great deal on the specific hardware and spend a lot of time on fine tuning the code. Porting a large petroleum reservoir simulator to emerging hardware architectures is expensive and risky. We analyze major components of an in-house reservoir simulator and investigate how to port them to GPUs in a cost-effective way. Preliminary numerical experiments show that our GPU-based simulator is robust and effective. More importantly, these numerical results clearly identify the main bottlenecks to obtain ideal speedup on GPUs and possibly other many-core architectures.

AMS subject classifications: 65M10, 78A48

Key words: GPUs, reservoir simulation, fully-implicit method.

1 Introduction

Nowadays, computers are equipped with multicore CPUs and coprocessors which have tens or even thousands of light cores, such as Intel Many Integrated Core (MIC) coprocessors and graphic processing units (GPUs). This kind of heterogeneous architecture provides opportunities for development of more powerful and more energy efficient supercomputers. In the Top500 (spring 2015) list of supercomputers, there are 17 clusters

*Corresponding author.
Email: lizhxtu@126.com (Z. Li), wush@petrochina.com.cn (S. Wu), xu@math.psu.edu (J. Xu), zhangcs@lsec.cc.ac.cn (C. Zhang)

accelerated by Nvidia or AMD GPUs and 11 by Intel Xeon Phi (MICs) chips in the top 100 systems. Although, the GPUs still dominates over the MIC coprocessors in terms of number of systems in the Top500 list for the moment, it is still too early to tell which coprocessor(s) will eventually win the race to the exascale era and beyond.

Reservoir simulation plays an important role in designing efficient development processes and improving oil recovery ratio. High-resolution reservoir models can better characterize the reservoir heterogeneity and describe complex water encroachment [19, 38]. The demand for more accurate computer simulations has led to larger and, in turn, more heterogeneous, reservoir models. Such models entail larger and more difficult linear systems. For fully implicit reservoir simulation, the solution of Jacobian systems often accounts for the majority of simulation time. Hence simulation can be greatly accelerated by faster linear solvers. Reservoir property calculation and matrix assembling, which although accounts for the majority part in source code, is cheap in terms of computational cost and is embarrassingly parallelizable. Since the parallel scalability depends strongly on what algorithm is used in application, speedup itself should not be over emphasized. State-of-the-art sequential solvers often utilize multilevel and multistage algorithms that offer fast convergence, especially for large models. It is challenging to port these algorithms to many-core architectures without compromising the rate of convergence or robustness. Many simple preconditioning methods, such as weighted Jacobi, polynomial preconditioning and multi-color versions of the incomplete factorization methods, are naturally parallel and exhibit strong scalability. Exposing fine-grained parallelism often involves either a careful redesign of existing algorithms or developing novel algorithms in which sequential dependencies on data are reduced or eliminated [21]. In both cases, reusing the legacy code is very difficult if not completely impossible.

A lot of research effort has been devoted to creating fine-scale reservoir models and efficient reservoir simulators on high-performance computers; see [26] and references therein for details. Many researchers have investigated reservoir simulation on many-core architectures; most of them are on GPUs. Klie et al. [29] developed a GPU-based GMRES solver with multiple preconditioners and achieved an overall computational speedup of 2x compared to conventional CPU multicore simulations. Yu et al. [53] developed GPU-based block ILU preconditioners and showed a 6x speedup compared with their CPU implementation for the industry standard SPE10 benchmark [15]. Appleyard et al. [1] developed a massively parallel nested factorization for two-stage preconditioner in combination with a GMRES solver and also assembled matrix on GPU. An overall speedup factor of 10x was achieved using a GPU over execution on a single CPU core. Tchelepi and Zhou [44], based on the Appleyard's work, developed a massively parallel nested factorization preconditioner on multiple GPUs and achieved 19x speedup for double-precision computations compared to sequential CPU code for the SPE10 problem. Fung et al. [24] used a heterogeneous approach in which their simulator ran on CPUs while the linear solver ran automatically chosen device (CPU or GPU). Esler et al. [21] developed a GPU based reservoir simulator and they claim that it costs less than two minutes on one Nvidia K40 card to solve the full SPE10 benchmark.