EAGE

ECMOR XV

Mo P097

Organization of High-performance Computing on the Mobile Platforms to Calculate the Oil Recovery of Reservoir

T.S. Imankulov* (Al-Farabi Kazakh National University), D. Akhmed-Zaki (Al-Farabi Kazakh National University), B.S. Daribayev (Al-Farabi Kazakh National University) & O.N. Turar (Al-Farabi Kazakh National University)

SUMMARY

Recently variety of novel parallelization technologies and approaches is rapidly developing. Distribution of high-powered graphical processors and parallelization tools on mobile platforms required a detailed comparison of computational capabilities on practical problems. This paper considers numerical investigation of oil displacement process by polymer/surfactant flooding taking into account water salinity and temperature effects, study parallel algorithms for it on several platforms and describes the organization of high-performance computing by using GPU of mobile devices. Calculation times, efficiency and speed up of these algorithms on various platforms were compared in order to identify optimal technology or combination of technologies for effective and well-optimized developments of industrial scale simulator. Basic parallelization technologies being used: MPI and CUDA. The calculations are performed on a mobile devices Xiaomi MiPad with NVIDIA Tegra K1 and on a personal computers with NVIDIA GeForce and Tesla K20. Besides the comparison of different version's working time profound analysis of different platforms' using expedience presented considering of the received working time benefits. Standard and proven parallelization tools are being used by way of samples for justification competitiveness of new types of parallelization. The research output has led the authors to come to the main conclusion: mobile devices can be used as computers to solve problems in oil industry.

Introduction

At the moment, large-scale calculations for the simulation of complex physical processes are always attached to a huge computational systems and often not available remotely. Taking oil industry as an example specialists has no way of monitoring the simulation process directly at the well location not to mention adjusting any production or injection settings of simulation. The solution to this problem would be to create mobile interface for stream visualization of data of calculation performed on cluster. Without access to the network it is possible to use the mobile device or a distributed system of such devices to perform small simulations.

This paper synthesizes some basic results on fluid flow computation in porous media using mobile computing technologies. Mobile computing consists of preparing and visualization of data on CPU and performing calculations on the Graphic Processing Units (GPUs) of the device. The basic idea is to compare the productivity of application with its desktop alternative. Desktop application was tested on average GPU and on one of the most advanced PC GPUs at the moment.

Current studies justify usage of mobile devices in complex computational applications since they have enough resources to handle users growing demands. Several publications (Parmar *et al*. 2013; Phan *et al*. 2002) propose using mobile platforms in order to build distributed computational clusters for high performance scientific problem solutions. Downside of ideas provided in abovementioned papers that the issues related to cross platform execution and heterogeneity did not raised at all.

Heterogeneity in this scope means using of GPUs that are special-purpose programmable parallel architectures, primarily designed for real-time rasterization of complex graphics. Due to their highly parallel design and dedicated computational nature, GPUs have recently been used for scientific calculations (McClure *et al*. 2014; Rong *et al*. 2014). In these publications and in our work we used one of the most advanced technologies of GPGPU --- Compute Unified Device Architecture (CUDA) of NVIDIA corporation. The architecture of GPU in general and mobile GPUs in particular especially developed in order to be used in computational intensive applications. For example, they are widely used for image recognition problems, image processing, general-purpose computation (Rofouei *et al*. 2008; Singhal *et al*. 2010; Huang *et al*. 2013; Bordallo *et al*. 2009) and etc.

The model considered in given paper consider simulation of oil displacement process occurring within reservoir of oil-gas development field. As a test problem for the measurement we took the surfactant polymer flooding process. Our goal is to show that mobile devices has enough calculation power for industrial scale simulations using GPGPU on the example of the problem above. The problem of surfactant polymer flooding is chosen as bright and demonstrative example of calculations being performed using high performance heterogeneous distributed systems.

Mathematical Model of Surfactant/Polymer Flooding (SPF)

The mathematical model of two-phase flow in porous media has following assumptions:

- incompressible flow;
- gravitational forces and capillary effects is neglected;
- two-phase flow (water, oil) obeys Darcy's law.

Mass conservation equations and velocities for each phase can be written as follows:

$$m\frac{\partial S_w}{\partial t} + div(\boldsymbol{v_w}) = q_1 \quad (1)$$

$$m\frac{\partial S_o}{\partial t} + div(\boldsymbol{v_o}) = q_2 \quad (2)$$

$$S_w + S_o = 1$$

$$\boldsymbol{v_i} = -K_0\frac{f_i(s)}{\mu_i}\nabla P, \;\; i = w, o \quad (3)$$

where m – porosity, S_w, S_o–water and oil saturations, , q_1, q_1 - source or sink, $\vec{v}_w, \vec{v}_o$ – velocity of the water and oil phases, $f_i(s), \mu_i$ – relative permeability and viscosity for phase i, K_0 –permeability tensor.
Polymer, surfactant, salt and heat transport equations are given by (Babalyan *et al*. 1983):

$$m\frac{\partial}{\partial t}(c_p s_w) + \frac{\partial a_p}{\partial t} + div(\boldsymbol{v_w} c_p) = div(mD_{pw} s_w \nabla c_p) \quad (4)$$

$$m\frac{\partial}{\partial t}(c_{sw} s_w + c_{so} s_o) + \frac{\partial a_{surf}}{\partial t} + div(\boldsymbol{v_w} c_{sw} + \boldsymbol{v_o} c_{so}) = div(mD_{sw} s_w \nabla c_{sw} + mD_{so} s_o \nabla c_{so}) \quad (5)$$

$$m\frac{\partial}{\partial t}(c_s s_w) + div(\boldsymbol{v_w} c_s) = 0 \quad (6)$$

$$\frac{\partial}{\partial t}[(1-m)C_r\rho_r + m(C_w s_w \rho_w + C_o s_o \rho_o)T] + div(\rho_w C_w \boldsymbol{v_w}) + div(\rho_o C_o \boldsymbol{v_o}) = div\big[\big((1-m)\lambda_0 + m(\lambda_1 s_w + \lambda_2 s_o)\big)\nabla T\big] \quad (7)$$

where c_p, c_s – polymer and salt concentrations in aqueous phase, c_{sw}, c_{so} – surfactant concentration in water and oleic phases, a_p, a_{surf} – polymer and surfactant adsorption functions, D_{pw}, D_{sw}, D_{so} – polymer and surfactant diffusion coefficients, C_w, C_o, C_r – specific heat of water, oil and rock, ρ_w, ρ_o, ρ_r – density of water, oil and rock, $\lambda_0, \lambda_1, \lambda_2$ – coefficients of thermal conductivity.
Initial conditions:

$$s_w|_{t=0} = s_{w0}, c_{pw}\big|_{t=0} = c_{p0}, \ a_p\big|_{t=0} = a_{p0}, c_s|_{t=0} = c_{s0}, T|_{t=0} = T_p$$
$$c_{sw}|_{t=0} = c_{sw0}, \ c_{so}|_{t=0} = c_{so0}, \ a_{surf0}\big|_{t=0} = a_{surf0} \quad (8)$$

Boundary conditions:

$$\frac{\partial s_w}{\partial n}\Big|_{\partial\Omega} = 0; \quad \frac{\partial P}{\partial n}\Big|_{\partial\Omega} = 0; \quad \frac{\partial T}{\partial n}\Big|_{\partial\Omega} = 0;$$
$$\frac{\partial c_{pw}}{\partial n}\Big|_{\partial\Omega} = 0; \quad \frac{\partial c_{sw}}{\partial n}\Big|_{\partial\Omega} = 0; \quad \frac{\partial c_s}{\partial n}\Big|_{\partial\Omega} = 0; \quad (9)$$

We used the following viscosity dependence on injected reagent concentrations and temperature: (Flory *et al*. 1953):

$$\mu_a = \mu_w\big[1 + (\gamma_1 c_p + \gamma_2 c_p^2 + \gamma_3 c_{sw} + \gamma_4 c_{sw}^2)c_s^{\gamma_5} - \gamma_6(T - T_p)\big] \quad (10)$$

$$\mu_o = \mu o_o\big[1 - \gamma_7(T - T_p)\big] \quad (11)$$

where $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6, \gamma_7$ – constants.μo_o – initial viscosity of oelic phase, T_p – reservoir temperature. The imbibition relative permeability curve for water/oil flow is given by

$$f_w(S_w) = S_w^{3.5}; \ f_o(S_w) = (1 - S_w)^{3.5}$$

The adsorbed concentration of polymer is a function of polymer concentration, given by:

$$a = \frac{bc_p}{1 + bc_p}$$

where b – Langmuir constant.
The numerical formulation of equations (1)-(9) based on finite difference method. The IMPES method is very used for solving such problems. But, in this case, we solve the equations explicitly. The explicit schemes are naturally parallelizable in these systems: a geometrical domain decomposition divides the original domain into subdomains such that the hyperbolic problems can be solved in parallel using the GPUs. Numerical realization and results of numerical experiments was proposed by the authors in (Danayev *et al*. 2015; Akhmed-Zaki *et al*. 2015).

Algorithm implementation using desktop GPU

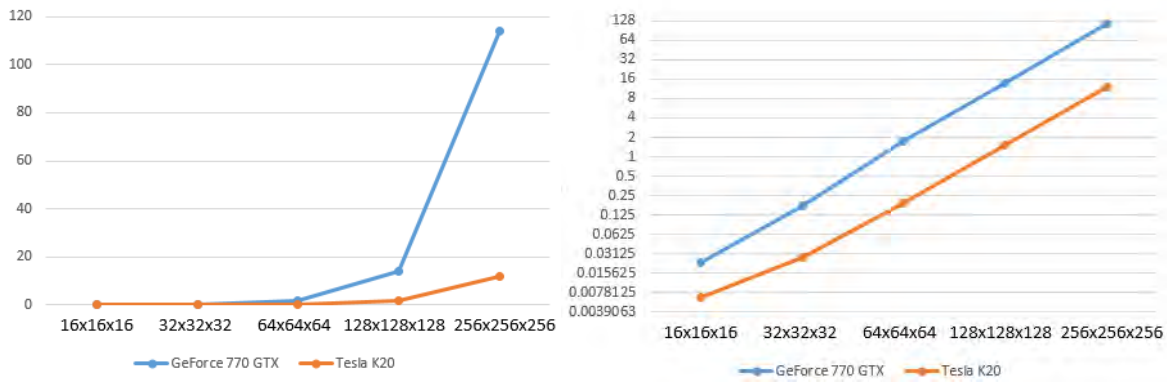
The mobile version of application was implemented using CUDA technology of parallelization using NVIDIA graphics cards. The only mobile processor implementing this technology available at the moment is NVIDIA Tegra K1 with Kepler microarchitecture. So we decided to use desktop GPUs with the same microarchitecture. The first one to be tested is a NVIDIA GeForce GTX 770 which is average one in terms of performance. GeForce series is widely used for computation intensive applications. Even some scientific calculations are implemented on graphics cards with GPU of that series (Nyland *et al*. 2007).

Second GPU used for testing of the application is NVIDIA Tesla K20 which is one of the most powerful GPUs at the moment with Kepler microarchitecture.

Our goal is to compare convergence times of finite differential algorithm implementation. The tests are performed on different grid sizes and with different threading settings. The algorithm uses variables with double precision and due to great number of arrays application occupies large space in RAM.

	Tesla K20(GK110)	GeForce GTX 770(GK104)	Tegra K1 (GK20A)
Cores	2496	1536	192
Memory bandwidth [GB/s]	208	224	17
Memory (max) [GB]	5	2	1
Compute Capability	3.5	3.0	3.0
Theoretical performance single (GFLOPS)	3524	3213	365
Theoretical performance double (GFLOPS)	1175	134	290

Table 1 *Description of the NVIDIA Kepler architecture video cards.*



(a) (b)

Figure 1 *Computation time comparison for desktop GPU devices: (a) classic representation, (b) vertical axis representing convergence times presented as logarithmic ruler.*

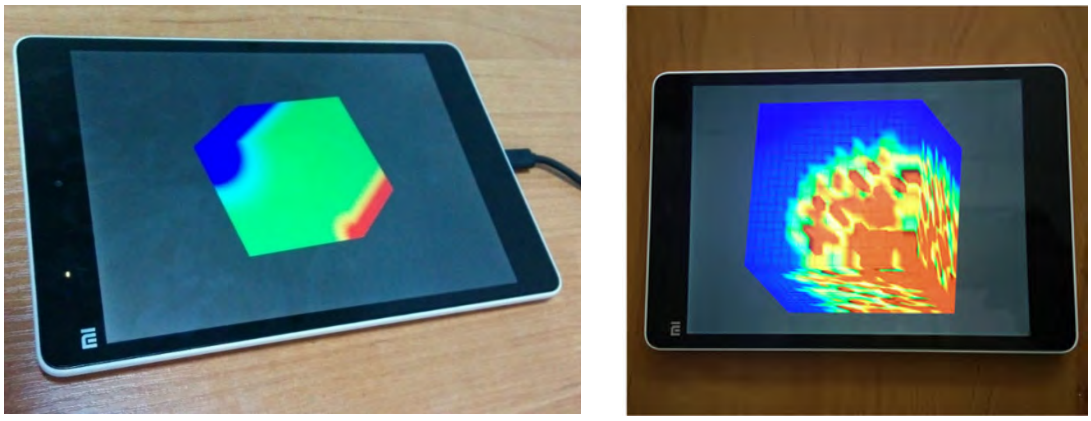
Analyzing Table 1 we can see that mobile device has even bigger double performance value than average desktop GPU which is greatly focused on single precision operations. Also we may expect that Tesla GPU will show tenfold faster time measurement.

Double precision operations are the major part of any scientific usage of GPU. Figure 1 demonstrates real measurements of application convergence time on different grid sizes. Note that on Figure 1b vertical axis representing computing time implemented as logarithmic ruler. On the figure and according numerical values in Table 2 we can see that whole computation time on GeForce (blue line)

always 9-10 times above the results of Tesla (red line). These results fit to theoretical performance of the devices.

Mobile application using CUDA technology

Mobile implementation of algorithm described above has been applied as APK program that was tested on NVIDIA Shield and Xiaomi MiPad tablets. These tables are the earliest ones with processor Tegra K1 (Kelion, 2014) which has same compute capability as GeForce GTX 770. Figure 2 demonstrates running application with visualization performed using OpenGL ES 2.0.



(a) (b)

Figure 2 *Demonstration of Finite differential application working on mobile device: (a) with constant permeability and (b) variable permeability.*

Let's compare convergence time of mobile application with desktop implementations. Table 2 shows running times of all application launches. Symbol '-' denotes that the launch was not successful due to the lack of the graphical memory. Indeed considering maximal graphic memory from Table 1 application sometimes cannot allocate memory since each node of the grid requires 168 bits of RAM for all variables.

Grid size	Application convergence time, sec		
	GeForce GTX 770	Tesla K20	Tegra K1
16x16x16	0.023	0.006657	0.042022
32x32x32	0.171	0.027161	0.137662
64x64x64	1.776	0.186602	0.764796
128x128x128	13.814	1.51414	-
256x256x256	114.124	12.1033	-
256x256x512	-	25.4145	-
128x128x64	6.633	0.74981	3.405675

Table 2 *Application convergence time with different grid sizes.*

Graphical demonstration of these results are shown in Figure 3. All launches for the same grid size make single iteration and equal amount of inner iterations when calculating Pressure. So considering the fact that all applications implement same math functions and skipping some low-level technical features we can claim that for same grid size different application launches perform equal amount of operations. Based on this we suppose that convergence times of launches on different platforms with same grid size have to be linearly proportional to theoretical double performances of the devices.

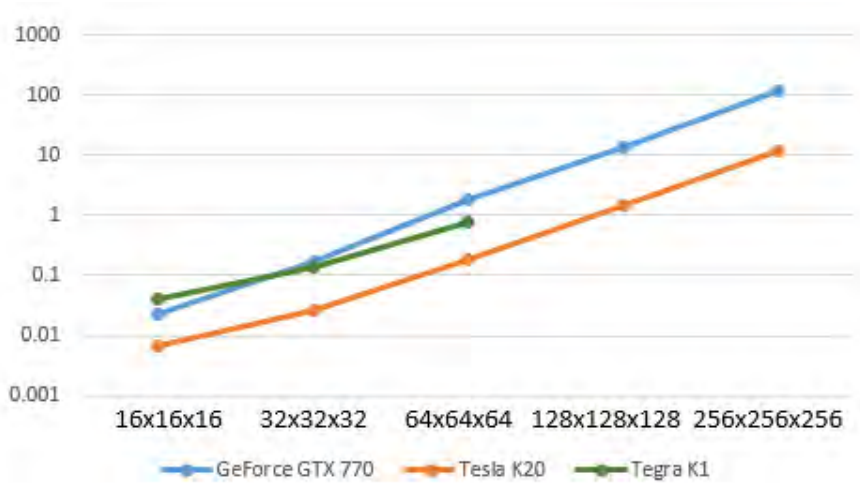


Figure 3 *Convergence time comparison for both mobile and desktop versions of the application.*

Mobile processor has more than two times higher theoretical double performance in GFLOPS. Despite the fact that this difference is unnoticeable at small numbers as the grid size increases the actual ratio of convergence times between GeForce and Tegra K1 become closer to theoretical expectations. Maximal grid size that we succeeded to run on all platforms 128x128x64 is shown in last row of Table 3. As we can see mobile processor performs calculations almost two times faster than desktop GPU GeForce GTX 770.

Conclusion

Paper describes testing of industrial scale process simulation on mobile device and comparison of its effectiveness with desktop application. Difference of the proposed method from already existing systems that use mobile devices as computational units like BOINC (boinc.berkeley.edu) is that we use GPGPU on mobile platform to perform complex computations. As it was demonstrated computational power of mobile device exceeded average desktop GPU. Considering availability of mobile technologies this computational power can be considered as a great advantage of its usage as computational hardware.

Disadvantage of such usage is low graphic memory that makes impossible to calculate large complex problems with described method on the single device. As a solution we can outline some distributed system of mobile devices. There are studies which present such systems implemented by MPI [16]. Further direction of our studies is implementing such system performing each node computations on GPU. So it will be combination of 2 different parallelization paradigms the CUDA and the MPI on mobile devices.

References

Ahmed-Zaki, D.Zh., Mukhambetzhanov, S.T. and Imankulov, T.S. [2015] Design of i-Fields System Component: Computer Model of Oil-Recovery by Polymer Flooding. *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics* (ICINCO 2015), **2**, 510-517.

Babalyan, G.A., Levy, B.I., Tumasyan, A.B. and Khalimov, E.M. [1983] *Oilfield development using surfactants.* Nedra, Moscow.

Bordallo, M., Hannuksela, J., Silven, O. and Vehvil¨ainen, M. [2009] Graphics hardware accelerated panoramabuilder for mobile phones. *Proc. SPIE Multimedia on Mobile Devices*, doi: 10.1117/12.816511.

Danaev, N.T., Mukhambetzhanov, S.T., Ahmed-Zaki, D.Zh. and Imankulov, T.S. [2015] Mathematical modeling of oil recovery by polymer/surfactant flooding. *Communications in computer and information science*. **549**, 1-12.

Flory, P.J. [1953] *Principles of polymer chemistry.* Cornell University Press.

Huang, M. and Lai, C. [2013] Accelerating Applications using GPUs on Embedded Systems and Mobile Devices. *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC),* 1031-1038.

Kelion, L. [2014] *CES 2014: Nvidia Tegra K1 offers leap in graphics power*. BBC.

McClure, J.E., Prins, J.F. and Miller, C.T. [2014] A novel heterogeneous algorithm to simulate multiphase flow in porous media on multicore CPU-GPU systems. *COMPUTER PHYSICS COMMUNICATIONS,* **185**(7), 1865-1874.

Nyland, L., Harris, M. and Prins, J. [2007] *Fast N-Body Simulation with CUDA.* GPU Gems 3. Addison Wesley.

Parmar, K., Jani, N., Shrivastav, P. and Patel, M. [2013] Mobile Grid Computing: Facts or Fantasy? *INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY SCIENCES AND ENGINEERING*, **4**(1).

Phan, T., Huang, L. and Dulan, C. [2002] Challenge: Integrating Mobile Wireless Devices into the Computational Grid. *Proceedings of the 8th annual international conference on Mobile computing and networking (MOBICOM), ACM*. doi:10.1145/570645.570679.

Rofouei, M., Stathopoulos, T., Ryffel, S., Kaiser, W. and Sarrafzadeh M. [2008] Energy-Aware High Performance Computing with Graphic Processing Unit. *Proceedings of the 2008 conference on Power aware computing and systems.*

Rong, F.M., Zhang, W.H., Shi, B.C. and Guo, Z.L. [2014] Numerical study of heat transfer enhancement in a pipe filled with porous media by axisymmetric TLB model based on GPU. *International Journal of Heat and Mass Transfer,* **70,** 1040-1049.

Singhal, N., Park, I.K. and Cho, S. [2010] Implementation and optimization of image processing algorithms on handheld GPU. *17th IEEE International Conference on Image Processing (ICIP),* 4481-4484.

Virtejanu, I. and Nitu, C. [2013] Programming Distributed Application for Mobile Platforms using MPI. *U.P.B. Sci. Bull.*, **75**(4), 143-148.