

GPGPU calculations of gas thermodynamic quantities

Igor Mračka, Peter Somora and Tibor Žáčik

Department of Applied Mathematics

Mathematical Institute

Slovak Academy of Sciences

Štefánikova 49, 814 73 Bratislava, Slovakia

mracka@mat.savba.sk, somora@mat.savba.sk, zacik@mat.savba.sk

Abstract—Computational processors NVIDIA Tesla GPU based on the new Fermi generation of CUDA architecture are intended to perform massively parallel calculations applicable to various parts of the scientific and technical research, including the area of fluid dynamics modeling, in particular the simulation of real gas flow. In this paper we show that a significant acceleration of simulation calculations can be achieved even without the parallelization of the solution of involved differential equations by parallel pre-calculation of thermodynamic quantities using GPGPU.

Index Terms—gas state equation, AGA8, gas thermodynamic quantities, CFD, GPGPU, CUDA, FERMI, NVIDIA Tesla.

I. INTRODUCTION

IT IS well-known that the computing power of processors keeps increasing by the (slightly modified) Moore's law [1], even though the focus of processor development has shifted from boosting the frequency to increasing the number of cores in processors. Main stream desktop processors contain four to eight cores. This new line of multi-core processors has changed the course of programming from serial towards parallel algorithms. Hand in hand with the increase of computer performance the demand arises for the increase of computation precision especially in scientific or technical applications [2]. However, in a large part of scientific problems the dependency between computational difficulty and calculation precision is "stronger" than linear and the computation time increases quadratically (or even faster) with increasing precision. In such cases the computer performance of common desktop multi-core CPUs is not sufficient. Fortunately, devices with a new architecture of General Purpose computation on Graphics Processing Units (GPGPU), usually termed as GPGPU processors, focused on massive parallel computations and capable of performing fast calculation in double precision is becoming available right now.

The new class of GPGPU massively parallel processors containing hundreds of cores is able to simultaneously process thousands of computational threads (a survey of general-purpose computations on graphics hardware is presented in [3]). Although this brings obvious advantages to scientific and technical computations, the developers must also deal with limitations of the new architecture (e.g., new memory classification, thread distribution between streaming multiprocessors,

etc.). An important aspect of the utilization of the GPGPU processors is the existence of development tools for the optimization and debugging of massively parallel algorithms. Nowadays, two probably biggest producers of GPGPU processors are AMD and NVIDIA. The new Radeon GPU processor series of AMD is based on the FireStream architecture [4] and contains more cores than the NVIDIA Tesla GPU processor based on the CUDA architecture codenamed FERMI ([5], [6]). Since the raw computational power and capabilities in double precision calculations of both class of processors are comparable, the main decision parameter for utilization could be the support of programming languages and development tools.

The NVIDIA CUDA architecture supports C, C++ and FORTRAN programming languages and several development environments (APIs) – CUDA C/C++, OpenCL, Direct Compute (and recently announced Microsoft C++ AMP) – thus ensuring a rising popularity among the scientific and technical community (see [7] and [8]). The spread of NVIDIA CUDA popularity is documented by a long series of papers and reports of scientific teams and institutions [2], [9], and has also reached the area of fluid dynamics modeling [10], [11], [12], [13], as proven by the increase of interest of scientific research centers in applying the NVIDIA Research Center program.

This paper is focused on the acceleration of (both steady-state and transient) simulation calculations of gas flow using approximations of values of thermodynamic quantities involved in the calculations. Several methods exist for the evaluation of values of thermodynamic quantities, each with its own range of application, accuracy and degree of computational difficulty. We show that, by pre-calculating the values of approximation matrices, it is possible to maintain a constant access time of the values of thermodynamic quantities during simulation independently of the method used. The short access time implies significant acceleration of simulation. Applying a more complex state equation of gas results in the increase of the time of calculation of each element of the approximation matrix. Increasing the approximation precision results in the increase of the matrix dimension and hence in a quadratic increase of the number of matrix elements to be evaluated. With this in mind, the utilization of massively parallel computations on GPGPU processors appears to be the ideal solution

for the evaluation of approximation matrices. Moreover, the parallelization itself is rather straightforward (regardless of the limitations caused by the transfer to GPU) without the need for any major changes in the implementation of the original algorithms. For the comparison of approximation matrices evaluation times the ratio between the time of GPGPU parallel evaluation and the time of one CPU core serial evaluation was taken. When considering parallel evaluation on multicore CPUs instead of serial evaluation, the resulting acceleration ratio will be correspondingly smaller, but not quite objective, since apart from the number of CPU cores the result also depends on the individual hardware configuration (activation/deactivation of hyper-threading technology, etc.) and the actual CPU usage of background running processes.

II. THERMODYNAMIC QUANTITIES IN SIMULATION CALCULATIONS

The following system of partial differential equations (representing the conservation laws of mass, momentum and energy, respectively, [14]) describes the one-dimensional model of the turbulent flow of a mixture of gas with constant composition through a pipeline of constant inner diameter.

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho v}{\partial x} = 0,$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial}{\partial x} (\rho v^2 + P) + \frac{\lambda}{2D} \rho v |v| + \rho g \frac{\partial z}{\partial x} = 0,$$

$$\frac{\partial}{\partial t} \left[\rho \left(\frac{v^2}{2} + h - \frac{P}{\rho} + gz \right) \right] + \frac{\partial}{\partial x} \left[\rho v \left(\frac{v^2}{2} + h + gz \right) \right] + \alpha (T - T_w) \frac{\pi D}{S} = 0,$$

where

- x – space coordinate,
- t – time coordinate,
- P – gas pressure,
- T – gas temperature,
- ρ – gas density,
- v – gas velocity,
- h – specific enthalpy,
- λ – resistance coefficient,
- D – pipeline diameter,
- S – pipeline cross-section area,
- z – pipeline altitude at given point,
- g – standard gravity acceleration,
- T_w – pipeline wall temperature.

Since in practical situations only the measurements of pressure P , temperature T and mass flow G are usually available, the gas velocity v is calculated using the following formula

$$v = \frac{G}{\rho S}.$$

The density is expressed using the state equation for the real gas

$$\rho = \frac{P}{ZRT},$$

where

- R – gas constant for given gas composition,
- Z – compressibility factor.

The change of enthalpy h occurring in the energy conservation law is given by the thermodynamic equation

$$dh = c_p dT - c_p \mu dP,$$

where

- c_p – specific heat capacity at constant pressure,
- μ – Joule-Thomson coefficient.

In general, the state equation expresses the relation between the quantities P , T and ρ , and is usually represented by a semi-empirical equation with the coefficients to be determined experimentally for some gas mixture, pressure and temperature ranges. Such approximated formulas are usually given the name of its authors, for example: Van der Waals, Redlich-Kwong, Peng-Robinson, Lee-Kesler, BWR (Benedict-Webb-Rubin) state equations. There are also various modifications of the original equations: Soave modification of the Redlich-Kwong equation or Elliott-Suresh-Donohue modification of the Peng-Robinson equation, etc. (see [15]). In some cases the equations get names after its institutions of origin, such as AGA (American Gas Association), see [16], or GERG (European Gas Research Group), see [17].

Specific heat capacity c_p , enthalpy h , and the Joule-Thomson coefficient μ are thermodynamic quantities depending on the gas state at a given point in space and time and are functions of pressure and temperature (for given gas composition): $c_p = c_p(P, T)$, $h = h(P, T)$, $\mu = \mu(P, T)$, and $Z = Z(P, T)$. All thermodynamic quantities mentioned above (as functions of pressure and temperature) can be derived from the actual state equation.

By solving the above equations for a steady state – assuming the time-derivations equal zero – after some simplifications we obtain the following formula representing steady-state hydrodynamic conditions in one pipeline:

$$P_1^2 - aP_0^2 = bG|G|,$$

where P_0 and P_1 denote the pressures at the beginning and the end of pipeline, respectively, and the constants a and b are determined by

$$a = \exp \left(\frac{2g}{Z_{av}RT_{av}} \right),$$

and

$$b = \frac{\lambda L}{DS^2} \cdot Z_{av}RT_{av} \cdot \frac{a-1}{\ln a},$$

where Z_{av} and T_{av} are average values of compressibility and temperature along the pipeline, respectively. For steady state simulations, both the average temperature calculation and the distribution of temperature along the pipeline network involve

the use of specific heat capacity at constant pressure and the Joule-Thomson coefficient.

For both transient and steady state simulations, if the pipeline network contains compressor stations, the model of compressor involves the use of compressibility and the Poisson coefficient.

It follows that regardless of the type of simulation (transient or steady-state) the evaluation of thermodynamic quantities (compressibility, specific heat capacities, enthalpy, Joule-Thomson coefficient, Poisson coefficient) occurs quite frequently. For example, in the case of steady-state optimization simulations the time needed to acquire compressibility and other thermodynamic quantities from state equations of Redlich-Kwong or Peng-Robinson occupies approximately half the time of the overall calculation. If, in particular, simulating the flow of natural gas, then instead of general state equations one can use the AGA8 version of state equation the coefficients of which have been derived directly for gasses with components and concentrations typical for natural gas [16], [18]. By using AGA8 the resulting gas properties, e.g., compressibility, can be determined much more accurately, but the compressibility calculation algorithm is much more calculation-demanding than in the case of Redlich-Kwong or Peng-Robinson. In fact, the evaluation of thermodynamic quantities in steady-state simulations has been consuming more than 98 % of the overall calculation time.

Moreover, the overall calculation time has increased to such an extent, that it has started to pose a problem in practical use.

III. APROXIMATION MATRICES

The solution of the problem could be the pre-calculation of thermodynamic quantities for selected values of pressure and temperature. The values for other pressures and temperatures can be obtained by interpolating the pre-calculated values.

Let (P_{\min}, P_{\max}) and (T_{\min}, T_{\max}) be intervals of pressures and temperatures respectively covering the values of pressure and temperature occurring in the calculations. For any concrete class of problems such intervals can be found.

Divide the intervals (P_{\min}, P_{\max}) and (T_{\min}, T_{\max}) into a given number of subintervals

$$P_{\min} = P_1 < P_2 < \dots < P_M = P_{\max}, \quad M \gg 1,$$

$$T_{\min} = T_1 < T_2 < \dots < T_N = T_{\max}, \quad N \gg 1.$$

Denote

$$\Delta P = P_2 - P_1, \quad \Delta T = T_2 - T_1.$$

By calculating the values of an arbitrary thermodynamic quantity in the points (P_i, T_j) one can obtain a matrix of numbers, which shall be called the approximation matrix of the selected quantity. The i -th row of the approximation matrix contains the values for the fixed pressure P_i and similarly, the j -th column contains the values for fixed temperature T_j .

The approximation matrix for compressibility

$$A_Z = \begin{pmatrix} Z(P_1, T_1) & Z(P_1, T_2) & \dots & Z(P_1, T_N) \\ Z(P_2, T_1) & Z(P_2, T_2) & & \\ \vdots & & \ddots & \vdots \\ Z(P_M, T_1) & Z(P_M, T_2) & \dots & Z(P_M, T_N) \end{pmatrix}$$

shall be called the compressibility matrix (and similarly for matrices of other thermodynamic quantities).

The request for a not pre-calculated value of compressibility is realized by means of bilinear interpolation (using the four closest grid points of the matrix) which is continuous and not computationally demanding.

That means, for the value of compressibility at (P, T) one has to find integers i, j and real numbers x, y , ($0 \leq x < 1$, $0 \leq y < 1$) such that

$$P = i \cdot \Delta P + x \quad \text{and} \quad T = j \cdot \Delta T + y.$$

Then for the (2×2) submatrix

$$A_{ij} = \begin{pmatrix} Z(P_i, T_j) & Z(P_i, T_{j+1}) \\ Z(P_{i+1}, T_j) & Z(P_{i+1}, T_{j+1}) \end{pmatrix},$$

the bilinear interpolation reads

$$Z(P, T) \doteq (1-x \quad x) \cdot A_{ij} \cdot \begin{pmatrix} 1-y \\ y \end{pmatrix}$$

(see Fig. 1).

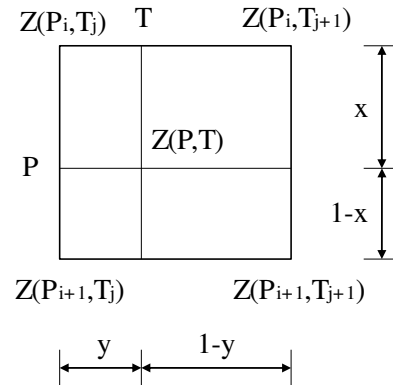


Fig. 1. Bilinear interpolation of $Z(P, T)$ from the four closest points of compressibility matrix.

It is obvious, that the increase of the number of grid points, $M \times N$, results in the increase of accuracy of the interpolated compressibility values. In the case of optimization calculations it is usually sufficient for the neighboring pressure grid points to be ca. 5 kPa apart and the neighboring temperature points to be ca. 0.1 °C apart. Thus, when covering the pressure interval from 0 to 10 MPa and temperature interval from -20 to 80 °C, the resulting matrix is of dimension $2,000 \times 1,000$ hence requiring 2,000,000 calculations of compressibility (see Fig. 2). A serial calculation of the compressibility matrix according to AGA8 can take tens of seconds on a common CPU. When solving an assignment with fixed gas composition

the matrix can be pre-calculated and the calculation obtains the required compressibility values by interpolation.

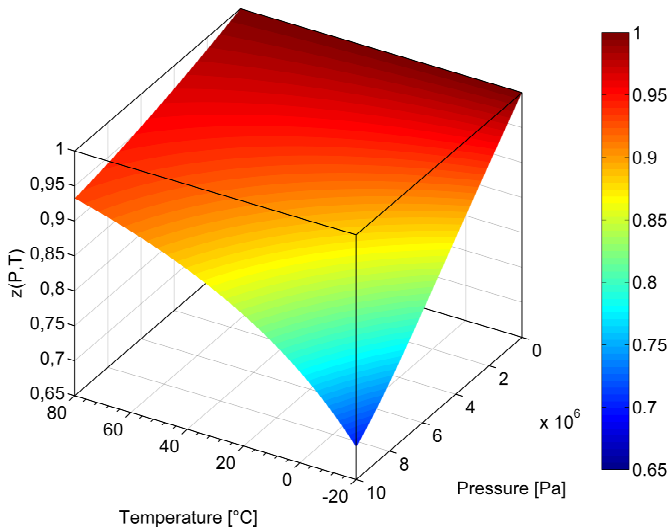


Fig. 2. Visualization of compressibility calculated from AGA8 state equation as function of pressure and temperature.

By using the approximation matrices we have achieved (even if slightly compromising the accuracy) a constant access time to the values of thermodynamic quantities. The time needed to calculate compressibility by bilinear interpolation is smaller than the time needed to directly evaluate compressibility from the more simple equations of Redlich-Kwong and Peng-Robinson. Hence the approximation matrix approach is suitable for any algorithm of thermodynamic values acquisition. From the time-consumption point of view, the use of approximation matrices is the same regardless of the state equation used in the simulation calculations.

However, the use of approximation matrices is relevant up to the point of gas composition change. Every change of gas composition requires the values of the matrices to be re-calculated. For example, when solving a number of optimization tasks with each task having its own gas composition, the continuous re-calculation of approximation matrices is from the time-consumption point of view unacceptable.

IV. GPU-BASED CALCULATIONS

The evaluation of matrices of thermodynamic quantities consists of a large number (of order 10^6) of mutually independent calculations. Among the inputs for the algorithm of evaluation are the following: a vector of pressures and temperatures and a matrix of constants derived from the properties of each component of the overall mixture of gas (the matrix of thermodynamic quantities is evaluated for a given or constant gas composition). The output of the algorithm is a matrix of values of a selected thermodynamic quantity (compressibility, enthalpy, thermal capacity, etc.). Even though

the algorithm itself is relatively small (less than 8 KB after compilation of AGA8 algorithm), it contains a large number of standard operations—multiplications, divisions and evaluations of analytic functions (powers, square roots, logarithms) – that have to be performed in double precision. For example one evaluation of compressibility of a mixture of natural gas with twelve components based on the AGA8 state equation requires approximately five hundred calculations of analytic functions (200 powers, 300 square roots) and approximately ten thousand standard operations (+, −, *, /).

One can clearly see that the problem of evaluation of state matrices is a typical case for massive parallelization within processors supporting the parallel processing of a large number (more than a thousand) of computation threads. Current processors in desktop PCs now usually contain four cores and allow the simultaneous processing of four to eight threads (for example, Intel Core i7 950, see [19]).

This was our main stimulus for deciding to use the massive parallel processors Tesla GPU based on the Fermi architecture (see [5]) that are supporting the CUDA C/C++ interface environment (see [7]) stemming from C++ and hence allowing a smooth transition of C++ algorithms onto the GPGPU platform while at the same time satisfying the necessary condition of performing mathematical operations (and analytic functions evaluation) in double precision. Compared to ordinary CPUs this hardware allows the parallel processing of a far larger number of computation threads simultaneously. More precisely, the Fermi architecture Tesla GPU processors consist of fourteen to sixteen streaming multiprocessors (SM) each containing thirty two cores, hence capable of processing at least 448 computation threads parallelly. Each SM unit has its own exclusive L1 cache and a shared memory (see [20], [6]) enabling it to optimize parallel processing by reorganizing the calculations into blocks such that the computation threads running on the same SM unit are sharing partial results. Another important part of the Tesla GPU processors is the Giga Thread planner distributing the blocks of threads to subordinate SM planners thus ensuring a uniform utilization of all SM units in the GPU processor. For full utilization of SM units in thread processing it is possible to define blocks containing up to 1024 computation threads. The overall number of blocks that can be processed in one request of parallel evaluation on a GPGPU processor (i.e., CUDA kernel) is “limited” by the size of the so called three-dimensional grid of blocks that is, 65535^3 , allowing the evaluation of the whole state matrix in a single request.

Main advantages of Tesla GPU processors [5] (compared to standard Intel Core i7 950 3.07 GHz PC CPU [19]):

- 14–16 SM units with 32 cores each = 448–512 total cores (PC CPU: 4–8 cores)
- massive raw power in floating point calculations – up to 1 TFLOPS (PC CPU: 70 GFLOP)
- large ECC internal memory of 3–6 GB unconstrained by operating system requests
- large throughput between SM units and global memory – up to 150 GB/s (PC CPU: 20 GB/s)

- large number of parallel computation threads possible to process – in so called data parallel algorithms (PC CPU: 4–8 threads)
- several parallel programming languages / environments support – CUDA C/C++, OpenCL, FORTRAN, Direct-Compute
- development tools for debugging and optimization of parallel algorithms – NVIDIA Nsight, NVIDIA Compute Visual Profiler

A notable disadvantage of the GPGPU processors compared to standard PC processors is a limited amount of resources such as constant memory, relatively small cache and a shared memory in SM units. The primary bottleneck in GPU calculations appears to be the throughput of the PCI-Express bus. For example, with SM unit frequency 1.147 GHz and fourteen SM units per processor the system is able to process up to 512×10^9 standard operations in double precision per second. If every evaluation of a state quantity would require the transfer of the complete task assignment (i.e., pressure, temperature and gas composition determined constants) occupying about two hundred bytes of memory, then such transfer through the PCI-Express bus (with speed ca. 4 GB/s) would take approximately 1/20 of a microsecond, nevertheless within that time the GPGPU processor could perform up to 25,000 standard operations in double precision! It follows that it is extremely important to minimize the amount of data transferred between the host PC and the GPGPU device.

Main disadvantages of Tesla GPU processors:

- limited memory resources of SM units – 64 KB for L1 cache and shared memory,
- low throughput of the PCI-Express bus (4–6 GB/s) – unsuitable for simple calculations where the ratio of the number of operations or against the number of bytes of task assignment and task output for one calculation is less than 100 : 1.

The problem of thermodynamic quantities matrix evaluation consists of a sufficient amount of independent calculations and is suitable for massive parallelization. Computational algorithm (AGA8) of state quantities is not particularly demanding with respect to the operational and constant memory so the resources of GPGPU processor are sufficient and no special algorithm tuning for GPGPU is needed. For example, the result matrix containing $2,000 \times 1,000$ values occupies 16 MB of memory which takes only ca 2 milliseconds to transfer from GPU to host PC. So the problem is an ideal candidate for maximal potential utilization of the massively parallel Tesla GPU processor, with the possible task runtime acceleration factor reaching 60 compared to serial evaluation of the matrix on a standard reference PC (Intel Core i7 950 3.07 GHz CPU).

Compared to the serial evaluation, the parallelization of the calculations on standard four core CPUs accelerates the task runtime approximately three to five times – the exact number being heavily dependent on the CPU configuration (activation/deactivation of hyper-threading technology, etc.) and the actual CPU usage of background running processes.

Such parallel CPU-based acceleration of the approximation matrices calculations is in no way sufficient for our objectives.

The transformation of approximation matrices calculation from CPU to GPU only requires the replacement of the matrix grid point evaluation cycle with the CUDA api function (CUDA kernel [21]) providing the distribution of individual evaluations to the GPU cores.

V. RESULTS

The simulation calculations were performed using a model of the Slovak transit gas pipeline network consisting of approximately two thousand three hundred kilometers of pipelines, four compressor stations each containing more than twenty compressors of various types, fourteen pressure regulator valves and circa four hundred block valves. As the state equation AGA8 was used, with the pressure range 8 to 10 MPa and temperature range -10 to $+60$ °C.

Without the use of approximation matrices the steady-state simulation runtime was 22.50 s. With the use of approximation matrices the same simulation runtime was 0.45 s (not including the time for the pre-calculation of approximation matrices) which represents a 50-fold acceleration. The simulation runtime is not affected by the change of state matrix dimensions. For each new gas composition the pre-calculation of six approximation matrices dimensioned $2,000 \times 1,000$ was required (each for one quantity: compressibility, enthalpy, Joule-Thomson coefficient, viscosity, Poisson coefficient, specific heat capacities at constant pressure). This takes 21.45 s when calculating on CPU. Most of the time, 17.50 s, is consumed by the compressibility matrix calculation.

The overall calculation time combining the steady-state simulation time with the approximation matrices pre-calculation time was $0.45 + 21.45 = 21.90$ s representing a 1.03-fold acceleration compared to the case not using the approximation matrices. Hence the approximation matrices evaluation consumes the majority of calculation time where the compressibility matrix evaluation takes about 80 % of the overall calculation time. It follows that for the sake of acceleration of calculations with varying gas composition the acceleration of approximation matrices evaluation is critical (especially for compressibility matrix).

The transient simulation calculation (four hours of transient gas transport) without the approximation matrices has taken 248.50 s, while with the use of approximation matrices the time taken was 38.90 s. The number of compressibility evaluations was approximately $20 \cdot 10^6$.

Fig. 3 depicts the area of the compressibility matrix used in the steady-state simulation optimization calculations with the assignment being the maximization of mass flow through the gas pipeline network. The color indicates the number of evaluations of gas properties for an area with dimensions $\Delta P = 5 \text{ kPa}$ and $\Delta T = 0.1$ °C uniformly covering the used cartesian product $[0 \text{ MPa}, 8 \text{ MPa}] \times [-10 \text{ °C}, 60 \text{ °C}]$. The overall task time was 387 s with more than $2.27 \cdot 10^9$ evaluations of compressibility for given P and T .

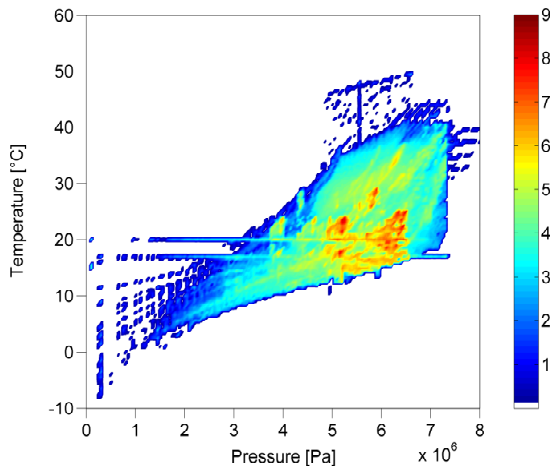


Fig. 3. Example of using of compressibility matrix in maximal gas pipeline network mass flow calculation. (Color scale indicates the exponent in the power of 10.)

It follows from Fig. 3 that the area used in the simulation occupies a significant part of the area covered by the approximation matrices.

The GPGPU acceleration of the approximation matrix evaluation was demonstrated using the calculation of the compressibility matrix based on the AGA8 algorithm, since the compressibility matrix evaluation consumes most of the overall calculation time.

The state matrix evaluation time depends on the matrix dimensions. Table 1 shows the comparison of compressibility matrix calculation times for CPU (serial calculation on Intel Corei7 950) and GPGPU processor (parallel calculation on NVIDIA Tesla C2050 GPU) and Table 2 shows the comparison of their acceleration factors. For additional comparison the results achieved on NVIDIA GTX 570 (with more cores than Tesla C2050, but with limited FERMI architecture) [22] are also stated. The results achieved on CPU are not satisfactory in the case of varying gas composition. Evaluation of the approximation matrix consists of a large number of independent calculations parameterized by pressures and temperatures. Such task is typically suitable for massive parallelization and hence ideal for GPGPU processing. By using massively parallel calculations on GPGPU processor we were able to cut down the evaluation time of state matrices (of size $2,000 \times 1,000$) from tens of seconds (18 s) to under one second (0.3 s), providing a satisfactory level of simulation precision.

In matrices with the number of compressibility evaluations under 2500 the calculation times on CPU are comparable to the selected GPGPU device. We have focused on cases where the compressibility evaluation count exceeded 10^4 (matrices with dimension 100×100 and more), as illustrated by the calculation times (Fig. 4) and calculation acceleration factors (Fig. 5) for the compressibility matrix evaluation. As one can see from Fig. 5, in the range of evaluations from 10^5 to 10^6 the acceleration factor of the GPGPU processor over CPU increases sharply. This corresponds to the area of evaluation

TABLE I
COMPARISON OF COMPRESSIBILITY MATRIX CALCULATION TIMES USING AGA8 ALGORITHM.

$M \times N$	Corei7 950 (serial)	Tesla C2050 (parallel)	GTX 570 (parallel)
	[s]	[s]	[s]
500×250	1.14	0.05	0.08
$1,000 \times 500$	4.39	0.1	0.18
$2,000 \times 1,000$	17.88	0.32	0.55
$4,000 \times 2,000$	70.26	1.16	1.84

TABLE II
COMPARISON OF ACCELERATION FACTORS OF COMPRESSIBILITY MATRIX CALCULATION USING AGA8 ALGORITHM.

$M \times N$	$T_{\text{Tesla}}/T_{\text{CPU}}$	$T_{\text{GTX}}/T_{\text{CPU}}$
500×250	22×	14×
$1,000 \times 500$	42×	25×
$2,000 \times 1,000$	56×	33×
$4,000 \times 2,000$	60×	38×

counts where the parallel calculation is too small to utilize the full potential of the GPGPU processors. Beyond the value of $5 \cdot 10^6$ evaluation counts one can see the slowing down (stabilization) of the acceleration factor increase corresponding to the area of evaluation counts where the potential of the GPGPU device is assumed to be fully utilized.

Concluding, the GPGPU parallelization is optimal beyond some value of evaluation counts. The upper bound for the evaluation counts is provided by the maximal allowed calculation time or by memory limits allocated for results. In the case of the evaluation of compressibility matrix with $2 \cdot 10^6$ elements (regarded as sufficiently large to provide satisfyingly precise results) the potential of the GPGPU device is still not fully used.

An increase of the matrix dimension to $4,000 \times 2,000$ (acceleration factor increases) would cause a slight increase of the overall calculation time (to 1.16 s), but would quadratically increase the amount of memory for pre-calculated matrices (to 64 MB for each) which is not desirable in our case. If one would assume a two-time increase in precision and four-time increase in memory consumption compared to the current “optimal” case, then the achieved full utilization of the potential of the GPGPU device is represented by an acceleration factor of 60 (for NVIDIA Tesla C2050 GPU).

VI. CONCLUSION

The focus of the paper rests on the problem of performance optimization in gas transit simulations, i.e., the calculation of thermodynamic quantities based on a given state equation of real gas. We solved this problem by pre-calculating the thermodynamic quantities matrices which provides independence from computational difficulty of the used gas state equation. The evaluation of a thermodynamic quantity in simulation calculations for given gas composition is performed as the bilinear interpolation from nearest grid points of the respective

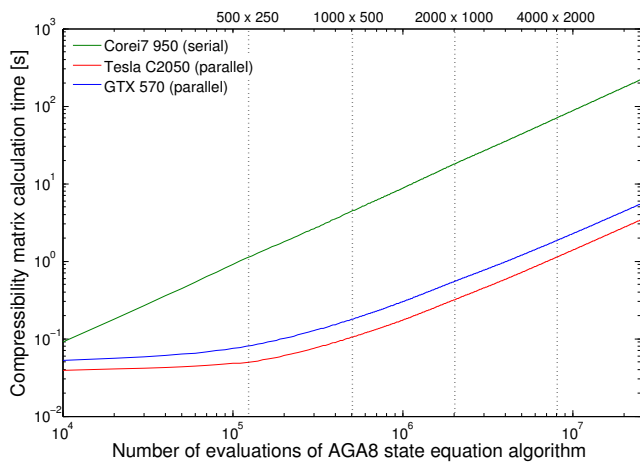


Fig. 4. Compressibility matrix calculation times on CPU and GPU in correspondence with compressibility evaluation counts through direct AGA8 algorithm calculation. (lower value is better)

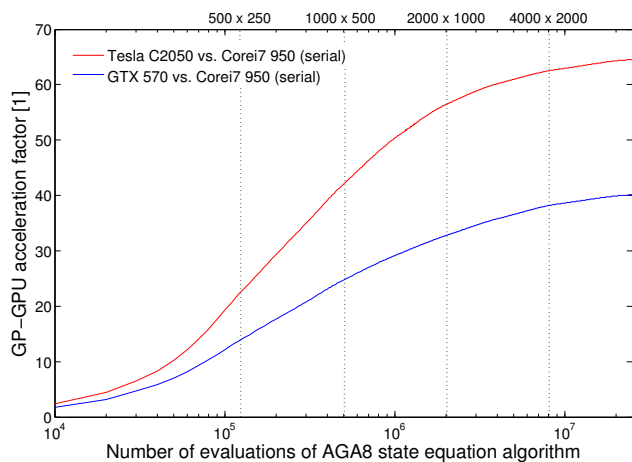


Fig. 5. Acceleration factor of GPU over CPU for compressibility matrix calculation in correspondence with compressibility evaluation counts through direct AGA8 algorithm calculation. (higher value is better)

pre-calculated matrices in much faster time than in the case of precise calculation of the given quantity, but with some degree of inaccuracy caused by small dimensions of the thermodynamic quantity matrix. In many cases the inaccuracy of the used real gas state equation is larger than the inaccuracy caused by bilinear interpolation even for small thermodynamic quantity matrices (e.g., 200×200). However, when considering more precise (and more computationally demanding) state equations, e.g., AGA8, GERG, the inaccuracy caused by the small dimensions of the approximation matrices becomes the limiting factor. Hence, choosing a more accurate state equation is not sufficient in order to achieve more precise simulations, but one also has to increase the dimensions of the approximation matrices. Since the computational difficulty increases quadratically with the size of the matrix, one has

to find a way to shorten the time required for the pre-calculation of the approximation matrices. A very suitable way appears to be the use of the new massively parallel GPGPU processors (NVIDIA Tesla C2050 GPU). We were able to shorten the time of pre-calculation of the compressibility matrix ($4,000 \times 2,000$) from 70 s in serial evaluation to 1.16 s in parallel evaluation (i.e., accelerated more than 60 times). Besides the evaluation of the thermodynamic quantities matrices there are other interesting problems in fluid dynamics (e.g., gas transport optimization, leak detection, etc.) that can benefit from the enormous potential of GPGPU. Additional promising predictions of the GPGPU hardware manufacturers state that the GPGPU potential shall increase several times compared to its current status (NVIDIA CEO on its GPU Technology Conference in 2010 presented a roadmap with the new CUDA architecture codenamed Maxwell, to be introduced in 2013, with performance about 10 times better than the current FERMI architecture [23]). From our point of view, all these facts forecast a bright future for the utilization of GPGPU technology in scientific and technical applications.

ACKNOWLEDGMENT

This work was supported by VEGA grant 2/0124/10.

REFERENCES

- [1] Moore's Law Made real by Intel Innovations <http://www.intel.com/technology/mooreslaw/>
- [2] High Performance Computing – Supercomputing with Tesla GPUs, www.nvidia.com/tesla
- [3] J. D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krger, A. E. Lefohn, T. J. Purcell, "A Survey of General-Purpose Computation on Graphics Hardware," *Computer Graphics Forum*, Vol. 26, No. 1, 2007, pp. 80–113.
- [4] GPU Computing: Past, Present and Future with ATI Stream Technology, http://developer.amd.com/gpu_assets/GPU_Computing_-_Past_Present_And_Future_with_ATI_Stream_Technology.pdf
- [5] Tesla C2050/C2070 GPU computing processor, http://www.nvidia.com/docs/IO/43395/NV_DS_Tesla_C2050_C2070_jul10_lores.pdf
- [6] Next-Generation GPU Architecture – 'Fermi', http://www.lunarc.lu.se/Documents/nvidia-workshop/files/presentation/45_Fermi.pdf
- [7] NVIDIA CUDA Compute Unified Device Architecture 2.0 Programming Guide, 2008.
- [8] Khronos Group, The OpenCL Specification, Version 1.0, 2009.
- [9] W. W. Hwu, "GPU Computing Gems," Morgan Kaufmann Publishers (is an imprint of Elsevier), Burlington, MA, USA, 2010.
- [10] N. Goodnight, CUDA/OpenGL Fluid Simulation, NVIDIA Corporation, 2007.
- [11] J. M. Cohen, M. J. Molemaker, "A Fast Double Precision CFD Code Using CUDA," *Proceedings of Parallel CFD*, 2009.
- [12] E. Phillips, Y. Zhang, R. Davis, J. Owens, "CUDA Implementation of a Navier-Stokes Solver on Multi-GPU Desktop Platforms for Incompressible Flows," 47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition, No. AIAA 2009-565, Orlando, FL, USA, January 2009.
- [13] J. C. Thibault, I. Senocak, "CUDA Implementation of a Navier-Stokes Solver on Multi-GPU Desktop Platforms for Incompressible Flows," 47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition, No. AIAA 2009-758, Orlando, FL, USA, January 2009.
- [14] L. D. Landau, E. M. Lifshitz, "Fluid Mechanics (Volume 6 of Course of Theoretical Physics Second English Edition), Pergamon Press, Maxwell House, Fairview Prak, Elmsford, NY, USA, 1987.

- [15] Y. S. Wei and R. J. Sadus, "Equation of State for the Calculation of Fluid-Phase Equilibria," *AICHE Journal Review*, vol. 46, No. 1, January 2000, p. 169–196.
- [16] M. Farzaneh-Gord, A. Khamforoush, S. Hashemi, H. P. Namin, "Computing Thermal Properties of Natural Gas by Utilizing AGA8 Equation of State," *International Journal of Chemical Engineering and Applications*, vol. 1, No. 1, June 2010.
- [17] O. Kunz, R. Klimeck, W. Wagner, M. Jaeschke, "The GERG-2004 Wide-Range Equation of State for Natural Gases and Other Mixtures," GERG technical monograph, Publishing House of the Association of German Engineers, Germany, 2007.
- [18] K. E. Starling, J. L. Savidge, "Compressibility Factor of Natural Gas and Other Related Hydrocarbon Gases," *Report No. 8 Software of American Gas Association(AGA)*, 3rd printing, November 2003.
- [19] Intel Core i7-900 Desktop Processor Extreme Edition Series and Intel Core i7-900 Desktop Processor Seriesm, <http://download.intel.com/design/processor/datashts/320834.pdf>
- [20] D. B. Kirk, W. W. Hwu, "Programming Masively Parallel Processors," Morgan Kaufmann Publishers (is an imprint of Elsevier), Burlington, MA, USA, 2010, pp. 8.
- [21] CUDA Programming Model Overview, 2008. <http://www.sdsc.edu/us/training/assets/docs/NVIDIA-02-BasicsOfCUDA.pdf>
- [22] NVIDIA Geforce GTX 570 GPU Datasheet, <http://www.nvidia.com/docs/IO/102043/GTX-570-Web-Datasheet-Final.pdf>
- [23] NVIDIA GPU Technology Conference 2010 Jen-Hsun Huang Keynote <http://blogs.nvidia.com/2010/09/gpu-technology-conference-liveblog-jen-hsun-huang-keynote/>