Computer simulation implementations and optimization of the right atrium of the heart based on GPU

Zhao Chunxi

Jilin Agricultural University, Changchun, 1301180, China

Keywords: GPU; accelerated computing; heart; computer simulation

Abstract. For heart right atrium electrophysiology simulation computation huge, time-consuming problem, a method based on high-performance graphics processing unit (GPU) to achieve parallel computing and optimization. First, consider the differences between right atrial heart cell center and the edge of the constructed one-dimensional non-homogeneous cardiac tissue model right atrium; operator split method enables solution of the model calculation task with parallelism. Presents three parallel strategies based on the specific solver process, and which takes the shortest thread blocks from the policy settings were further optimized data storage mode switching frequencies and so on. The results show that: for simulation 500 cells CUDA program execution time than the serial program fell by 60%, after further optimized CUDA program execution time can be reduced by 84%; the greater the right atrial cardiac tissue, GPU acceleration effect more obviously. The results verify the GPU acceleration solution method can significantly increase the heart rate right atrium solver model, reduce the actual execution time.

1. Introduction

In recent years, with the development of graphics processing units (Graphic Process Unit, GPU), the use of the processor on the graphics card to speed up the completion of a general purpose computing model solver speed attentions. Within GPU often contain hundreds of stream processors, can provide an ideal parallel architecture to handle thousands of concurrent threads. In addition, GPU global memory and shared memory bandwidth is very large, so read and write speed, high processing efficiency. These characteristics are applied to the heart of a computer model of GPU solver possible. However, unlike in the past need not be concerned about the underlying hardware of the computer programming mode in order to improve GPU acceleration effect, the programmer often require different treatment for different tasks and the amount of objects from the size of thread blocks, CPU host and GPU devices many aspects of communication, the entire data on the GPU memory allocation and so after considering specific solutions.

For computer simulation of the heart, the researchers presented the relevant GPU accelerated heart anatomical model solver methods; main object of study is the heart of the mechanical activity rather than electrical activity. Sato et al reported the feasibility of GPU accelerated ventricular tissue electrical activity simulation, but did not discuss the optimization program, and single-cell model used in the first generation, equations and parameters involved in solving the needs relative to the second-generation action potential dynamics model is much simpler. GPU has yet to see reported in the heart of the right atrium simulation applications, so this paper heart right atrium to the second generation single-cell model established on the basis of one-dimensional non-homogeneous organization, proposed and compare different strategies for solving parallel rate influence, and GPU-accelerated implementation of the model simulation and optimization methods are discussed.

Computer graphics text configured for NVIDIA GeForceGTX550Ti. It consists of more than four processors; each processor contains 48 stream processors. Implement the entire program based Compute Unified Device Architecture CUDA 4.2 (The Compute Unified Device Architecture).

2. Calculation and Optimization

2.1 Parallelization strategies Model Calculation

Right atrial heart tissue made up of many cells connected to the conductive coupling from one another through the gap. Organization of membrane potential changes follows the following equation:

$$\nabla \Big| (\sigma V) = \mathrm{A_m}(\mathrm{Cm}\frac{\partial V}{\partial t} + \mathrm{I_{ion}}) \qquad (1)$$

For the solution of the model calculation with parallelism, firstly, the operator splitting method using equation (1) into a description of the electrical activity of single-cell behavior of ordinary differential equations (2) and intercellular electrical tension expansion step of partial differential equations (3) ;

$$\frac{dV}{dt} = -\frac{1}{\mathrm{C_m}}\mathrm{I_{ion}} \qquad (2)$$

$$\frac{\partial V}{\partial t} = \frac{1}{\mathrm{A_m C_m}}\nabla \bullet (\sigma V) \qquad (3)$$

Operator splitting so that when solving action potentials can be calculated for each cell, and then consider the interaction between the cells electrically coupled. Since the solution of the single cell model the actual count is the most time-consuming, especially for the case of more than the number of cells, thus solving operator split so different single cell can give different threads simultaneously, so that solving tasks with a parallelizability.

This paper constructs a one-dimensional non-homogeneous tissue heart right atrium. Since the center to the edge of the differences in cell morphology, so each cell size and cell membrane capacitance exponential calculation coefficient:

$$F = 1.07 \times 29.0\frac{1.0n}{N} \Big/ 30.0(1.0 + 0.7745\exp(-(29.0 \times \frac{1.0n}{N} - 24.5) \Big/ 1.95))) \qquad (4)$$

Where: n-th cell corresponding coefficient F stands; N represents the total number of cells.

The electrical activity of the cell itself using the second generation of action potential kinetic model description, different current flows through the cell membrane by the respective channel gating factor adjustment, gating factor in turn was a function of membrane potential. Gating factor s relations with the membrane potential V is as follows:

$$\frac{ds}{dt} = F(V, s) \qquad (5)$$

Therefore, the solution unicellular operator first need to calculate parameters, namely the use of formula (4) to get the actual membrane capacitance and cell size of each cell, and then solving ordinary differential equations (5) to give the gating factor, and then calculate the corresponding ion current, final based on the total ion current, solving ordinary differential equations (2) to give the membrane potential of individual cells. Here, all ordinary differential equations are solved using Euler time step to take 0.03ms. Since the membrane potential to solve a differential equation involving more than 20, often large amount of computation, so this part of the calculation of this article all by the GPU. According solving process, designed three core functions in order to complete the variable initialization (kernels 1), capacitors, cell size and other parameters and solution (Kernel 2) gating factor to calculate the ion current and membrane potential (Kernel 3). Among them, the core function of the number 1, 2 and 3 need to register, respectively 19, 32 and 25, each of the multiple processor threads 256,1024 and 1024, respectively.

After obtaining the membrane potential of individual cells, according to the operator splitting of thought, considering the role of intercellular coupling of potential, solving partial differential equation (3). Here, the results of single-cell potential as an initial value calculated using the three-point difference method for solving the equation (3). Space step taken 0.1mm. σ according to formula (4) changes exponentially from the center to the edge. Organizational boundary-free flux boundary condition for processing, namely:

$$(\nabla V)\big|_{x=0} = (\nabla V)\big|_{x=l_s} = 0 \qquad (6)$$

Calculation of intercellular coupling portion employs two strategies in parallel strategy by the CPU 1, 2 in parallel strategy done by GPU. Thus, a policy 2, except for three kernel solver single cell design, but also increase the kernel Calculated for diffusion and boundary conditions. Kernel 4 used several registers 26, each multiprocessor number of threads in 1024. Visible, Strategy 1 and Strategy 2 is the task of solving the entire subdivision into several modules to different kernel functions were completed.

The impact on the operation speed of comparing different division of tasks, in the policy 3 kernel will integrate a number of functions by the policy 2 4 reduced to two. In addition to a kernel initialization task to complete, the remaining computing tasks are completed in a kernel function. Two kernel functions required number register 19 and 32, respectively, each multi-processor thread 256 and 1024, respectively.

2.2 Optimization of CUDA program

To improve performance, we use NVIDIA Performance Analyzer Visual Profiler program for analysis. Based on the results, from the thread block size, data exchange, and memory allocation tripartite face CUDA program optimization. Select the text of three parallel strategies shortest execution time optimization strategy as a starting point. NVIDIA GeForce GTX550Ti CUDA computing capability to meet the specifications, so each multiprocessor can accommodate up to 1536 threads, 8 threads block. To make a thread in working condition fills the entire multiprocessor; the total number of threads in the block should be greater than the number of threads that can accommodate multiprocessor divided by the total number of thread blocks (192). The total number of multi-processor registers and the number of registers for each thread needs to be seen; can accommodate 1024 threads on a multiprocessor. Therefore, for computing tasks of this article, the number of threads is designed to 512, a total of two threads block.

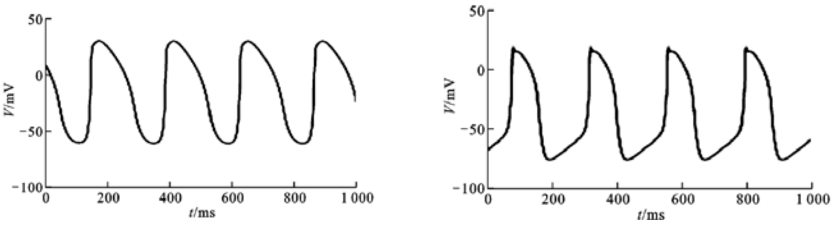
If each time step execution of the operation, will copy the data from the memory to the main memory, then a lot of time consumed in the frequent exchange of data. And once as much done on GPU computing and scratchpad to be a number of computing tasks to complete before data back to main memory, so you can reduce the frequency of data exchange. GPU shared memory is limited by the capacity of paper 500 time steps is calculated as a group to arrange a data exchange, the frequency of data exchange between host and GPU in this way reduced.

Due to the large global memory capacity GPU device and can be accessed all the threads, it is often used as the main memory data, but it is slow to read and write is reduced efficiency. By adjusting the storage structure, used in the calculation of higher frequency data (such as membrane potential, membrane capacitance, etc.) into the on-chip shared memory fast access to speed up access; try to save local variables in registers or shared memory to improve the access speed.

3. Results and discussion

3.1 Tissue action potential

To ensure the accuracy of the text GPU solution method, first calculate the right atrium of the heart tissue model of action potential. Figures 1a and 1b are action potentials center cells and edge cells.



(a) Center Cell (b) Border cells

Figure 1. The heart and right atrial tissue model of action potential

As can be seen, cells after each resting potential repolarization automatic uplift gradually reach the threshold and trigger another exciting feature of self-discipline. Compared with the edge of the cell, the center cell depolarization rate is slower, the action potential duration long. Right atrial tissue heart center and the edge of the maximum diastolic potential cells were -57mV, -76mV, rising rates are extremely 7.9V / s, 12.5V / s. These results and existing research, and the resulting calculation is consistent with the CPU simulation results described in this paper GPU solver accuracy.

3.2 Execution time and speedup

To assess the effect of GPU solver, the paper use NVIDIA Visual Profiler measuring actual execution time of three strategies.

Figure 2 shows the simulation 1.6s, 200 cells in the case. The results show that the execution policy 3 shortest, longest desired policy 1. This is because, compared with other policies, the policy of partial differential equation solver by the CPU, potentially reducing the parallel degree programs, while increasing the frequency of the data exchange between the host and GPU devices. Strategy 3 with respect to the policy 2, after the integration of multiple kernels can reduce memory between function calls and access frequently time-consuming, thus improving the overall efficiency. Thus, as much as possible to increase the number of tasks and reduces the degree of parallelism kernels will help improve processing speed.

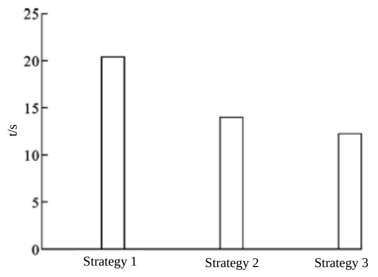
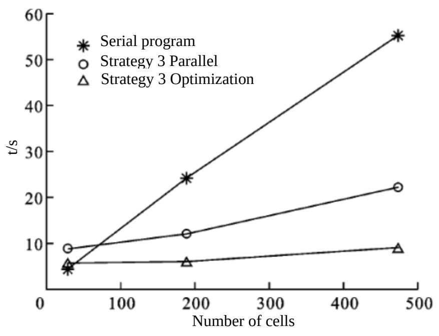
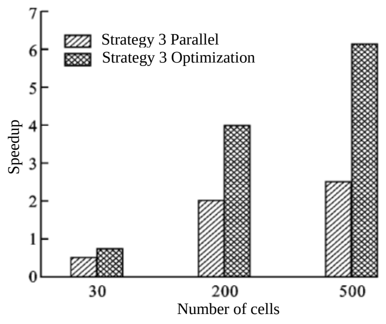


Figure 2. Three parallel execution time strategy

In three parallel strategy, since the execution time policy 3 shortest, most efficient, and therefore the policy paper 3 is optimized. Figure 3a shows the number of cells in different tissue models serial (marked *), policy 3CUDA parallel (marked ○) and Strategy 3 optimization (marked Δ) actual execution time of the program. 3b is a policy CUDA 3 parallel speedup program before and after optimization. Figure 3a reflect the 200 and 500 cells, tissues, CUDA parallel program execution time is far less than the serial program, and larger the organization, the more the number of cells, the more significant acceleration, optimized execution time can be further shortened . For example, 500 cells, tissue simulation, with respect to the serial program, CUDA parallel program execution time before optimizing a 60% reduction, optimized execution time is reduced by 84%. However, 30 cells, before CUDA parallel program execution time optimization 8.92s, but longer than the execution time of the serial program is optimized to reduce the execution time of 5.86s, but still greater than the execution time of the serial program . This may be due, in a small number of cells, small model organization, the parallel computing savings time is less than data exchange between CPU and GPU time-consuming, resulting in prolonged overall execution time. This shows that GPU accelerated simulation method for small-scale operation not only can not play to their strengths, but will reduce the efficiency. Can also be seen to Figure 3b, the larger the organization, the greater the speedup, the effect is significant after optimization.



(a) Execution Time



(b) Speedup

Figure 3. Effect of different cell numbers and programming strategies of execution time and speedup

4. Summary

In this paper, the right method to use GPU acceleration of cardiac electrical activity of atrial tissue simulation explored. The results show that: the operator splitting, the heart of right atrial tissue model solver the task itself may have to use parallelism to achieve accelerated GPU possible. The single-cell proliferation to solve and part of all to try to increase the GPU parallelism, reduce the frequency of the bulk transfer of data exchange through, and take full advantage of the shared memory and registers and other measures, can accelerate the effect is obvious. Parallelization strategies and optimization method proposed is a more complex simulation of right atrial tissue heart lay the future on the basis of a personal computer.

Reference

[1] JAYSHREE P, JITENDRA K, MADHURA B A. GPGPU processing in CUDA architecture [J]. Advanced Computing: An International Journal, 2012,3 (1): 105-105.

[2] MACEDONIA M. The GPU enters computing's mainstream [J]. Computer, 2013,36 (10): 106-108.

[3] HOANGTRONG T M, WILLIAMS G S B, LEDERER J W, et al. GPU-enabled stochastic spatiotemporal model of rat ventricular myocyte calcium dynamics [J]. Biophysical Journal, 2011,100 (3): 557.

[4] YU Rongdong, ZHANG Yin, ZHANG Sanyuan, et al. GPU accelerated simulation of cardiac activities [J]. Journal of Computers, 2010,5 (11): 1700-1704.