Han B, Lu XZ, Xu Z, Li Y, Seismic damage simulation for urban buildings based on high-performance GPU computing, Proc. 14th Int. Conference on Computing in Civil and Building Engineering, 2012.06. Moscow, Russia :368-369.

Seismic damage simulation for urban buildings based on high-performance GPU computing

Bo Han, Xinzheng Lu & Zhen Xu

Department of civil engineering, China Key Laboratory of Civil Engineering Safety and Durability of China Education Ministry, Tsinghua University

Yi Li

Key Laboratory of Urban Security and Disaster Engineering of Ministry of Education, Beijing University of Technology

Abstract

Refined models have been an important development trend of urban regional seismic damage prediction. However, the application of refined models has been limited due to their high computational cost if implemented on traditional Central Processing Unit (CPU) platforms. In recent years, Graphics Processing Unit (GPU) technology has been developed and applied rapidly due to its powerful parallel computing capability and low cost. In this paper, a simulation method for urban regional seismic damage is developed based on GPU-CPU cooperative computing technology, which reduces the computing time significantly. The seismic damage simulation is then applied to a medium-sized city using the proposed method to illustrate the unique advantages of GPU technology in large-scale regional seismic damage simulations.

Keywords: urban regional seismic damage simulation, GPU, parallel computing

1 Introduction

China is subjected to some of the most serious seismic disaster threats in the world. The scientific prediction of seismic damage is an extremely important task to provide early warning of potential seismic risks (Qiao & Yan 2005). Currently, many researchers are trying to improve the traditional seismic damage prediction methods to make them more accurate, efficient and realistic. The development and application of refined models has been the main trend of urban regional seismic damage prediction (Hu & Xu 1995). However, the computing platform is an important constraint of this approach. When the results must be obtained in very limited time (e.g., for some emergency response purpose), a very powerful computer platform is necessary (Hori & Ichimura 2008). Currently, most simulations using refined models are based on super computer systems, which are very expensive to maintain. Thus, the wide use of such systems is very difficult. Hence, urban regional seismic damage simulations based on refined models are "feasible" rather than "available" methods now.

In recent years, the rapid development of Graphics Processing Unit (GPU) technology has resulted in a new vision for refined model simulation (Wu & Liu 2009). Traditional methods are based on Central Processing Unit (CPU) platforms, and the development of the CPU follows Moore's Law. Although the performance of a single CPU core is increasingly powerful, the application of CPU platforms in urban regional seismic damage simulation is limited due to the high cost of each CPU core. In contrast, for a GPU, though the performance of a single GPU core is relatively weak, there are many more cores than in a CPU. Thus, the total floating computing capacity of a GPU is much

higher than that of a CPU, while its cost is a small percentage of the cost of a traditional CPU platform. After NVIDIA Corporation developed the Compute Unified Device Architecture (CUDA), the programming difficulty of GPU general computing has been significantly reduced. Now, GPU computing has played an increasingly important role in biology, electromagnetism, geography and so on (Zhang et al. 2009).

Despite its advantages, GPU is not a perfect solution for all problems. Because the computing capacity of each GPU core is relatively weak and the data exchange between different GPU cores is very time consuming (Li et al. 2009), computing tasks with the following characteristics can take full advantage of GPUs and will be the most suitable for GPU computing:

(1) The computing task can be divided into many subtasks;

(2) Each subtask has a moderate computing workload and can be individually implemented on a single GPU core;

(3) The data exchange between different subtasks is limited.

Fortunately, urban regional seismic damage simulation has all of these features. Although there are thousands of buildings in an urban area, if each building is treated as a subtask and simulated with macromodels (e.g., concentrated-mass story model) that are able to meet the accuracy requirement, the computing workload of each subtask (i.e., the degree of freedoms (DOFs) of each building) is sufficiently small for a single GPU core. Furthermore, there is little interaction between different buildings during an earthquake, resulting in little data exchange between different GPU cores. Therefore, the computing capacity of GPU cores can be maximized with these features. Because there are hundreds of cores in a GPU and each GPU core can be used for the simulation of one building at a time, only several task assignment rounds are required to complete the simulation for a city with thousands of buildings using GPU computing. Thus, the computational efficiency should be very high.

For the aforementioned reasons, GPU/CPU cooperative computing provides a good hardware platform for urban regional seismic damage simulation. However, there are still some problems that must be addressed for the application of this technique. This paper conducts a preliminary study of urban seismic damage simulation using GPUs to illustrate the implementation procedure and advantages of this technology.

2 Program architecture

For the program architecture, “smaller subtasks-less data exchange” should be satisfied to achieve more efficient CPU/GPU cooperative computing. The details of the design of program architecture are provided below.

2.1 Principles

(1) The CPU is used for data reading and the assignment of computing tasks due to its powerful logic computing capacity.

(2) The GPU is used for the nonlinear time-history computing of individual buildings with its powerful parallel computing capacity.

(3) The communication between the CPU and GPU is reduced to prevent communication delays.

2.2 CPU computing tasks

(1) Read the ground motion data and building parameter data and store them in the host memory.

(2) Allocate the space in the host memory inside of the CPU and the global memory inside of the GPU for the data exchange between the CPU and GPU.

(3) Copy the data between the two memories inside of the CPU and GPU.

(4) Invoke the global function in CUDA to call the GPU for calculation and manage the GPU resource.
(5) Output results.

2.3 GPU computing tasks

(1) Allocate the space in the graphic memory for temporary data.
(2) Read the data for each building, perform nonlinear time-history computing, and write the result to the graphic memory that has been allocated by the CPU for data exchange.

2.4 CPU-GPU communication mode

(1) Define the GPU's parameters in the main function running on the CPU, such as <<<grid, block, thread>>>.
(2) Use "cudaMemcpy()" to copy data between the CPU and GPU.

3 Mechanical model of the buildings

Most buildings in the city are ordinary multistory buildings. Thus, in this research, they are model with a multi-degree-of-freedom concentrated-mass model (i.e., the multistory shear model in Figures 1 and 2). The multistory shear model is not optimal for all of the buildings; some other macromodels, such as the beam model, are also appropriate for some high-rise buildings. However, this paper focuses on the demonstration of the feasibility and efficiency of the proposed GPU simulation; further research will be required for more detailed modeling.

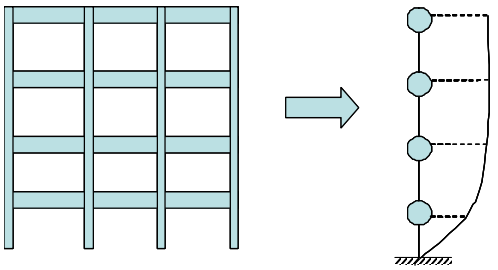

Figure 1. Diagram of the multistory shear model for a building

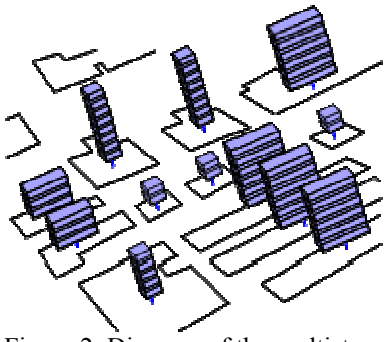

Figure 2. Diagram of the multistory shear model for a city

The bilinear hysteretic model and Lu-Qu hysteretic model (Lu et al. 2009) are used as the hysteretic models to represent the interstory behavior. The computing required for the bilinear hysteretic model is easy, so it is used to compare the computational efficiency. More complex interstory behavior can be considered in the Lu-Qu hysteretic model, so this model is used for the seismic damage simulation of buildings in a real city.

If implicit dynamic computing is used, the GPU computing time will increase significantly due to the additional logical operations induced by the nonlinear iterations; the convergence problem may occur by using implicit dynamic algorithm. Thus, in this paper, the central difference method (Chopra 1995) is used to solve the dynamic equations. The formula of the central difference method is

$$(\frac{m}{\Delta t^2}+\frac{c}{2\Delta t})u_{n+1}=P_n-(k-\frac{2m}{\Delta t^2})u_n-(\frac{m}{\Delta t^2}-\frac{c}{2\Delta t})u_{n-1} \quad (1)$$

In this formula, classical Rayleigh damping is used for the damping matrix.

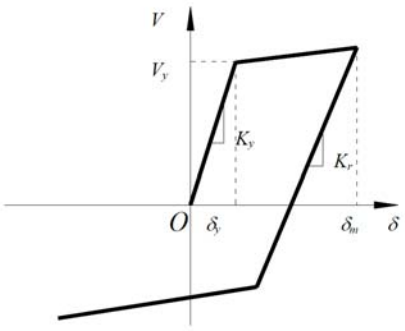

Figure 3. Diagram of the bilinear hysteretic model

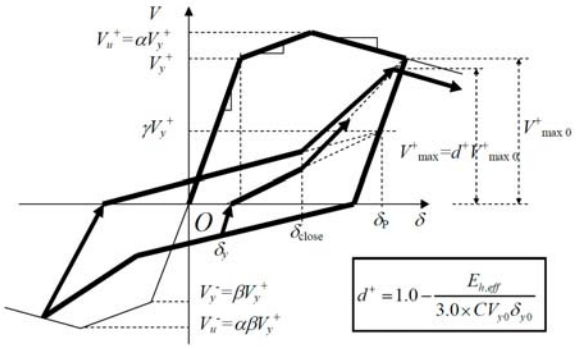

Figure 4. Diagram of the Lu-Qu hysteretic model

4 Implementation and Validation

4.1 Implementation of parallelization

The parallelization of the program is achieved with the following steps:

(1) Program the kernel function for the dynamic computing of a single building using CUDA C. Because every building is treated as an independent parallel task, the entire time-history computing should run inside of one kernel function. The kernel function's process should be as follows:
   a) Read the data of the single building assigned to this thread from the global memory;
   b) Perform the time-history computation;
   c) Write the results to the global memory.

(2) Program the host functions (running on the CPU), such as data inputting, copying and outputting.

(3) Program the main function, which manages the GPU resource and calls the kernel function for computing.

4.2 Program performance benchmark

To benchmark the performance of the parallelization and the corresponding parameters, the performance comparison between the GPU and CPU platforms is shown below.

4.2.1 Data for the benchmark

(1) 1 000 six-story buildings with the same parameters.

(2) Computational time and steps: 40 s, 8 000 steps.

(3) Interstory hysteretic model: bilinear hysteretic model (Figure 3).

(4) Only output the result of 1 building to reduce the bottleneck effect caused by the hard disk's writing speed.

4.2.2 Platforms

CPU： Intel Core i3 CPU 530 @2.93GHz

Memory： DDR3 4G 1333MHz

GPU ： NVIDIA GeForce GTX 460

Graphic memory: GDDR5 1GB 950MHz

The two platforms have similar prices (approximately 200 US Dollars), so they can be used to compare their performance-to-price ratio.

In the test, the parallel computing parameters of the program are as follows: grid=1, block=4, GPU_ Architecture=sm_10 (NVIDIA 2011).

4.2.3 Results of the benchmark

The relationship between the quantity of buildings and computational time was compared, and the results are shown in Figure 5.

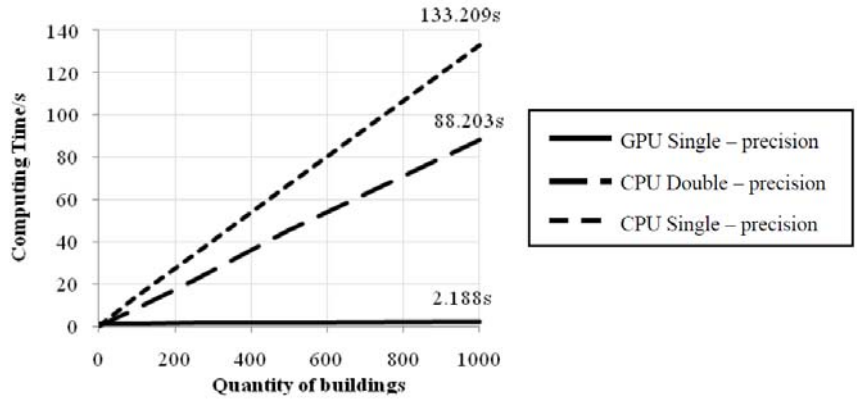

Figure 5. Relationship between the quantity of buildings and the GPU and CPU computing times

The computing time of the GPU is much shorter than that of the CPU. Because the shape of the GPU computing time curve is unclear in Figure 5, it is enlarged in Figure 6.

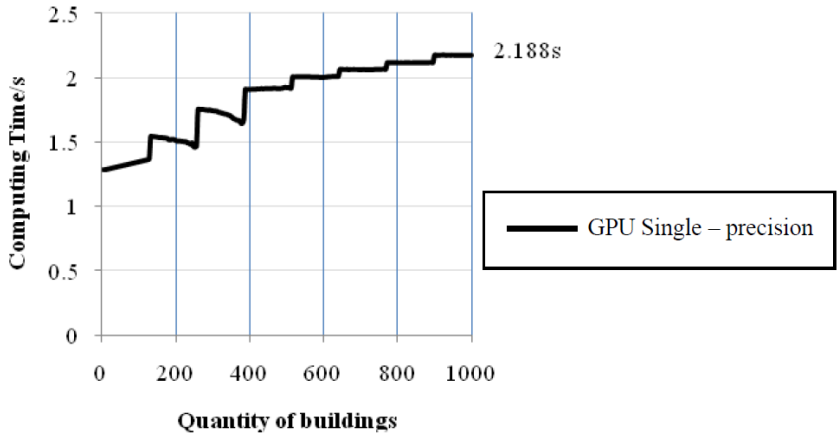

Figure 6. Relationship between the quantity of buildings and the GPU computing time

As Figures 5 and 6 show, the shape of the CPU program has a linear time curve, indicating that the computing time is proportional to the quantity of buildings. This result is consistent with the serial computing theory, the concept of which is “Single thread, Single task”. In contrast, the shape of the GPU program’s time curve is polyline, and the turning points are always at the points at which the quantity of buildings is a multiple of 128 because when using CUDA for parallel computing, every 32 threads in a same block constitute a “warp”. Inside of a warp, the parallel performance is very high,

but there is latency between two warps. In this test, the number of blocks is 4. Therefore, when the number of threads is larger than 4×32=128, there will be one more warp to compute. Thus, the computing time will elongate due to the scheduling latency. However, this effect will be reduced as the quantity of threads increases. When the number of threads is smaller than 384, the effect is very obvious; however, when the number of threads is larger than 384, the effect is not as obvious because when the quantity of threads is very large, CUDA can optimize the schedule of the threads; thus, the latency is hidden in the computing time.

Comparing the GPU and CPU computing times shows that the GPU is perfect for extensive parallel nonlinear time-history computing, as shown in Figure 7.

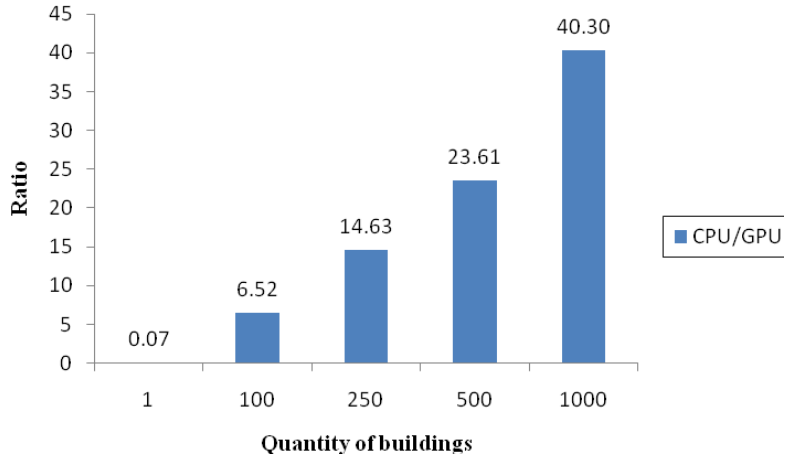

Figure 7. Comparison of the CPU/GPU computing time ratio

When a single building is computed, the CPU computing time is 7% of the GPU computing time because the performance of a single CPU core is much better than that of a single GPU core. However, as the number of buildings increases, the CPU computing time increases significantly, while the GPU computer time increases slightly due to the parallelization. Thus, when 1 000 buildings are computed, the GPU computing time can be 1/40 of the CPU computing time.

Table 1 compares the CPU and GPU results. When t=40.000 s (i.e., the last time point), there is approximately 0.1% error between them. This accumulated error is due to the difference of the CPU and GPU's floating point computing rules. The error is in an acceptable range and can thus be ignored.

Table 1 Results of the computations using a CPU and GPU (t=40.000 s)

Displacement (10^{-2} m)	1F	2F	3F	4F	5F	6F
GPU	0.522 84	0.709 95	0.247 54	0.196 42	0.196 33	0.196 28
CPU	0.522 72	0.709 70	0.247 29	0.196 16	0.196 07	0.196 02
Deviation	0.02%	0.04%	0.10%	0.13%	0.13%	0.13%

5 Application

As an example, the seismic damage prediction of a medium-sized city is implemented using the proposed method. The simulation steps are as follows:

(1) Convert the information of the city's buildings (area, height, number of floors, coordinates and structural type) from the GIS database to the input file of the program.
(2) Use the Lu-Qu hysteretic model for computing; the parameters can be estimated by the methods proposed by Xu et al. (Xu et al. 2008)
(3) Run the time-history computing and obtain the displacements and collapse data of the building.
(4) Convert the MDOF model to the 3D model of urban buildings and map the displacements to the 3D model. Generate the vertices and triangular patches and obtain the dynamic display results.

Figures 8 and 9 show the effects of modeling in the system at Step (1).

Figure 8. General plan view of the city

Figure 9. General oblique view of the city

There are 7 449 buildings, 172 064 vertices and 314 220 triangular patches in the model. The nonlinear simulation of the city is implemented by the methods suggested in this paper. The ground motion of the Northridge record (a 40-s strong motion record) is chosen for computing. The nonlinear time-history computing of the 7 449 buildings is accomplished in 216 s, 194 s of which is used for data output and only 22 s of which is used for nonlinear computing. The program meets the requirements of efficient low-cost seismic damage prediction. Figure 10 shows the damage distribution predicted by the program. The red buildings indicate the buildings that may collapse, while the gray buildings are those that will not collapse. If the collapse effects algorithm is introduced (i.e., the collapsed floors are removed), the seismic damage simulation can be more realistic, as shown in Figure 11.

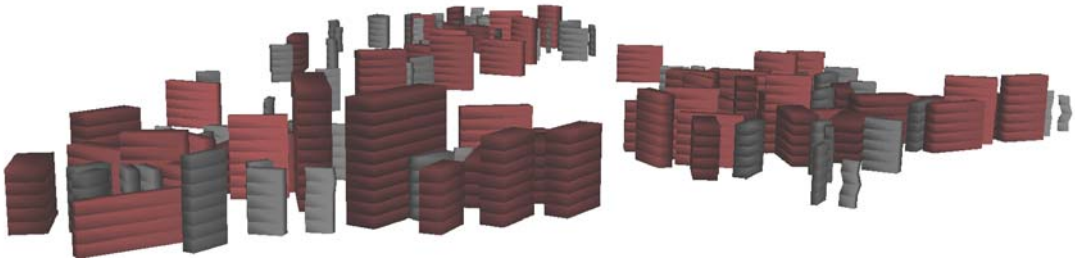

Figure 10. Partial oblique view of the dynamic scene (the displacements scaling ratio: 10)

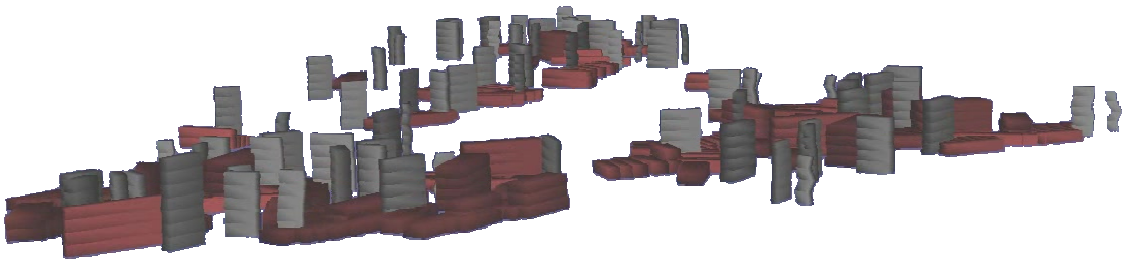

Figure 11. Simulation results of collapse

6 Conclusion

In this study, GPU-CPU cooperative computing for the seismic damage simulation of urban buildings is developed. Seismic damage simulation for urban buildings based on high-performance GPU computing reveals that the efficiency using a GPU can be 40 times greater than that when using a CPU at a similar price, meeting the requirements for high-performance computing. The damage simulation for a real medium-sized city shows the wide application prospects of the proposed method for seismic damage prediction. Future work, including improving the efficiency and generality of the program, optimizing the mechanical model, and enhancing the rendering and scene display effects, will create an efficient, low-cost urban regional seismic damage simulation technique.

Acknowledgements:

The authors are grateful for the financial support received from the National Nature Science Foundation of China (No. 51178249, 90815025), the Tsinghua University Research Funds (No. 2010THZ01) and the Program for New Century Excellent Talents in University (NCET-10-0528).

References

QIAO, Y.L. and YAN, W.M.,2005. Review of predictive method of seismic damages to buildings. *Industrial Construction*, 2005, (6), 1-6.

HU,B.S. and XU, X.W.,1995. Look forward to developmental direction of predicting earthquake damage to buildings. *World Information On Earthquake Engineering,* 1995, (03), 33-38.

HORI, M. and ICHIMURA, T., 2008. Current state of integrated earthquake simulation for earthquake hazard and disaster. *Journal of Seismology*, 2008, (2), 307~321

WU, E.H. and LIU, Y.Q., 2004. General Purpose Computation on GPU. *Journal of Computer Aided Design & Computer Graphics*, 2004, 16(5), 601-612.

ZHANG, C.H., LIU, J.Q. and XU, Q.J., 2009. Analysis and application of the GPU parallel computing technology. *Information Technology*, 2009, (11), 86-89.

LI, B., ZHAO H.C. and ZHANG, M.F., 2009. Parallel Programming For High-Performance Computing on CUDA. *Microcomputer Applications*, 2009, (09), 55-57+64+69.

LU, X.Z., YE, L.P., LIAO, Z.W. et al, 2009. *Elasto-plastic analysis of buildings against earthquake—theory, model and implementation on ABAQUS, MARC AND SAP2000*. Beijing, China Architecture & Building Press.

CHOPRA, A.K., 1995. *Dynamics of Structures*, New Jersey , Prentice-Hall.

NVIDIA, 2011. *NVIDIA CUDA Programming Guide*, Available online: http://developer.download.nvidia.com/compute/DevZone/docs/html/C/doc/CUDA_C_Programming_Guide.pdf, Last accessed: November 2011.

XU, F., CHEN, X.P., REN, A.Z. and LU, X.Z., 2008. Earthquake Disaster Simulation for an Urban Area, with GIS, CAD, FEA, and VR Integration. *Tsinghua Science and Technology*, 2008, (S1), 311-316.