Application of High Performance Computation for the Prediction of Urban Area Earthquake Disaster

APPLICATION OF HIGH PERFORMANCE COMPUTATION FOR THE PREDICTION OF URBAN AREA EARTHQUAKE DISASTER

Maddegedara Lalith*, Muneo Hori**
GCOE, Dept. of Civil Engineering, the University of Tokyo, Japan *
Earthquake Research Institute, the University of Tokyo, Japan **

ABSTRACT**:** With the aim of simulating earthquake disaster of a vast urban area, we enhanced the Integrated Earthquake Simulator (IES) with high performance computation (HPC). IES is a system to seamlessly simulate possible earthquake hazard, disaster, evacuation and recovery. The need of such system is indispensable. Businesses, local and state governments need reliable quantitative predictions of infrastructural damages due future earthquakes, to minimize damages and have sound recovery plans to minimize socioeconomic aftermaths. Only a system like IES can make reliable predictions taking the latest information and the current built-in-environment to the account; conventional predictions cannot take the current built-in-environment to the account. In order to simulate a series of scenario earthquakes within a reasonable short time, which is necessary for reliable quantitative predictions, IES should be enhanced to utilize HPC resources. Though IES has been enhanced with standard parallel computing techniques, some bottlenecks seriously limited it's scalability to double digit numbers of CPUs. We significantly improved its parallel computing performance by eradicating all the major bottlenecks. Details of the bottlenecks, remedies implemented in the modified IES and other performance enhancements are presented. Scalability of the modified IES is demonstrated with representative GIS tiles from Shinjuku district of Tokyo. It is shown that parallel computing performance of IES is improved significantly both in terms of problem size and run time. According to rough estimations, simulation of 2-3 million buildings in Tokyo needs ten to fifteen thousands of CPUs.

KEYWORDS: Integrated earthquake simulation, high performance computing

1 INTRODUCTION

Due to the continually increasing social and economic costs incurred by natural disasters, many countries are striving to adopt cutting edge technologies and latest information in disaster management to increase the social and economic security. Reliable prediction of a natural disaster, including the losses and reconstruction costs, is of great importance in disaster management; worldwide, there had been several fold increase in economic damages due to natural disasters. In the case of disasters of several decades recurrence interval, like earthquake and tsunamis, the predictions based on conventional statistical analysis of past events have a low reliability. Especially, when it comes to the estimation of damages to the built environment, on which most other factors like cost of reconstruction depend on, the changes within decades are so drastic that past event based predictions are of low reliability. Numerical tools capable of simulating the existing built environment taking the current conditions of structures, code of practice used for the structure designs, construction materials used, etc. into account are indispensable in disaster management.

Applications of such numerical tools for a large urban area like Tokyo are so computationally intensive that it requires the power of next generation super computers. While a single simulation of a large urban area itself requires the power of super computers, necessity of Monte-Carlo simulations to increase the reliability of the predictions further elevate the necessary amount of computational resources. Therefore, these numerical tools should be enhanced to utilize the power of next generation super computers like K-computer in Kobe, Japan.

A system, consisting of a collection of such numerical tools, called Integrated Earthquake Simulator (IES) is being developed (Ichimura et al. 2004, Hori et al. 2008). Its long term objectives are to simulate earthquake strong ground motion from source to site, damages to structures, emergency evacuation and short and long term social and economic recovery. Obviously, for disaster prediction in an urban area like Tokyo, where there are several millions of structures and 15 million people, IES is required to be enhanced with high performance computing (HPC) techniques. In this paper, we focus on the enhancements of seismic response analysis (SRA) module of IES with HPC techniques. Specifically, the objective of this work is to increase the parallel performance of IES so that Monte-Carlo simulations of millions of structures in an urban area like Tokyo can be conducted within a reasonably short time, on tens of thousands of CPUs.

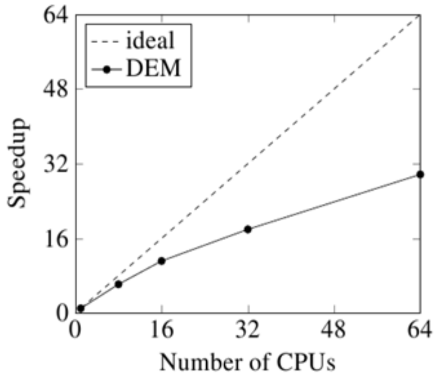

Figure 1 Speedup of the current IES parallel extension.

Though the current SRA module of IES is equipped with standard parallel computing techniques, a serious boost in parallel performance is necessary to simulate a millions of structures in a single simulation. As shown in figure 1, the scalability of current IES is too low to attain the intended ultimate goal. Speedup is a common measure of scalability of a parallel program defined as the run time ratios between single CPU and many CPUs. The speedup curve in figure 1 is obtained with 10,000 buildings and a nonlinear Discrete Element Method (DEM) code. At the best case, it seems to produce 50 CPUs output with 256 CPUs, which is far lower than the scalability required for simulating a large urban area. The objective of this work is to significantly improve the parallel performance of IES SRA module.

The identified major parallel performance bottlenecks in IES are the use of temporary files to exchange data between IES and independent SRA executable, unbalanced work load assigned to CPUs, large number of unnecessary inter-processor communications and the large number of file input/output (I/O) operations involved in output data saving. These bottlenecks not only affect the parallel scalability but also seriously limit the number of building to several tens of thousands.

In implementing countermeasures for bottlenecks, priority is given to elimination of temporary files, since extensive use of temporary files in parallel environment can harm computer systems. The use of temporary files is totally eliminated either with library or system-V shared memory segment and semaphore mechanisms. All-worker model with several orders of magnitude less number of message passing is implemented instead of master-slave model in the current version. Static load balancing based on the previously recorded run time information is introduced to assign nearly equal workloads to all the CPUs. The output saving time is drastically reduced with advanced MPI-IO operations which allow collective file IO operations of large number of CPUs. These modifications improved the parallel performance to the near ideal.

The rest of the paper is organized as follows. Section two presents the details of the bottlenecks and countermeasures. The section three presents a demonstration simulation and discusses scalability of the modified IES. Some concluding remarks are given in the section four. In this paper, the term current IES refers to that with performance bottlenecks and the outcome of this work is referred as the modified IES.

2 PERFORMANCE BOTTLENECKS AND REMEDIES

By the design, IES SRA module has the potential to reach near ideal parallel scalability. Currently, IES uses a series of simple to moderately advanced non-linear SRA methods like multi degrees of freedoms (MDOF), discrete element method (DEM), one component model (OCM), nonlinear fiber element model, etc. All these seismic response analysis models are implemented as serial programs. Parallel computing resources are used to execute these serial computing codes in large numbers. The main tasks of parallel SRA module are generating input data for a suitable SRA method according to the type of each structure, assign the structures to all the available CPUs such that resources are optimally used and manage the terabytes of output data efficiently. In this setting, near ideal parallel scalability is reachable with large number of CPUs, since relatively a small number of inter-processor communications is necessary to manage input/output data and find the optimal usage of computing resources. In the rest of this section, the identified bottlenecks of the current parallel SRA module and the remedies which improved the parallel performance to the near ideal are explained.

2.1 Inter-process communication (IPC) with temporary files

The use of temporary files in parallel environment is not only degenerate the scalability, but also could damage the file system. Some of the SRA programs in IES are developed with FORTRAN; mostly complying with Fortran 77 standard. On the other hand, IES is developed with C++; object oriented design is the best for a system like IES, under which various numerical tools are gathered. In mixed language programming, the most simple is to make the parent program invoke the child programs with *system*( ) command, exchanging data with temporary files. This has been the case with current IES; it is the most used inter-process communication (IPC) method in research community. Temporary files based IPC is a serious bottleneck in parallel computing. File access is time consuming and at the minimum it needs 4-6 million file operations for a simulating 2-3 millions of structures in a city like Tokyo. Also, it can heavily stress the file system when all the CPUs start to simulate two or three storied residential buildings, the most abundant in any city.

The temporary file based IPC bottleneck is eliminated by calling the FORTRAN based SRA programs as libraries. In mixed mode programming, converting child programs to static or dynamic libraries and linking to main program is the standard technique. It neither involves temporary files nor sacrifices the performance. All the major SRA programs are modified and linked to IES as libraries. This process was time consuming and error prone since converting some of the codes was difficult due to the non-standard compiler options used by their developers. Under such situations, a less error prone and less time consuming approach is adopting shared memory and semaphore mechanisms of SystemV IPC (commonly abbreviated SysV IPC) (Daniel et al. 2000, Stevens 1999) for inter-process communication and called the SRA programs as independent executable with *system*() command. As the name implies, shared memory mechanism allow multiple processes to share a common segment of memory to exchange a large volume of data. Semaphores can be used to prevent multiple processes simultaneously accessing a shared memory segment. In addition to attaching as libraries, the major SRA programs are attached to IES using these IPC mechanisms. As it is shown in the next section, SysV IPC does not degenerate the parallel performance.

2.2 Large number of file I/O operations in output data handling

When used to simulate a large urban area, SRA module produces a large volume of output data; in average 8.5GB binary data per 10,000 structures. Since visualization is one of the easiest means of comprehending such a large volume of data, IES should organize and save the output data in a ready-to-visualize format. In the current SRA module, this step involves an extremely large number of temporary files, not only hindering the parallel performance but also posing a serious threat to the file system. While the use temporary files must be eliminated, methods to minimize the parallel performance degeneration should be sought.

In order to minimize the reduce the parallel performance degeneration, output handling modules are redesigned to save the large volume output to a small number of files, each not exceeding the maximum input file size of the visualization software. In order to minimize post-processing time, the output files should be formatted for the visualization software. However, with POSIX type I/O it’s difficult to save data in ready to visualize format. To explain why, it needs to explain how the data is distributed over the CPUs. As will be explained in the section 2.3, buildings from one GIS tile are distributed over many CPUs to attain good load balancing. With POSIX IO, saving data of one GIS tile, scattered over many CPUs, to a single file can only be done with sequential file access; either sending all the data to master CPU to re-organize and save or each CPU opens and saves data to one file sequentially. Utilizing parallel I/O operations, one can attain better performance than either of these options.

We utilized the MPI-IO functionalities comes with the MPI-2 standard to write the SRA output in ready to visualize format, thereby achieving higher parallel performance and reducing the visualization time. Compared to POSIX IO, MPI-IO can deliver much higher IO performance in parallel environment, provided supporting hardware and parallel file system are available. MPI-IO provides four levels of file access; independent or collective access of contiguous or noncontiguous data. We utilized the level 3 access function *MPI_File_write_all*( ) which allow non-contiguous collective access to a file, to write SRA output data to a ready to visualize format.

2.3 Unbalanced work load assigned to CPUs

Though the SRA module of IES is embarrassingly parallel, proper load balancing is important to achieve a good scalability. Current IES SRA module does not assign equal amount of work to each CPU leading to large run time difference among CPUs. Due to the dependence of location and the magnitude of the earthquake, the exact run time of non-linear SRA models for each structure cannot be predicted. Therefore, some form of dynamic load balancing is necessary to attain a perfect load balance. Possibly, hybrid solution like static load balancing for the first 95% and switching to dynamic load balancing for the remaining would be the best since this involves less number of inter-processor communication. Though a hybrid of static and dynamic load balancing is a good choice for this problem, only static load balancing is implemented since dynamic load balancing requires changes to the core of IES.

In the simple static load balancer, first all the building shape and previous run time data are shared among all the CPUs. The shared shape data contain a shapes and previous run times of group of buildings from one or more GIS tile. Each data set is compressed to reduce to size of the message. CPUs pick a subset of data so that each has equal amount of run time, calculated from the previous run time data. The use of static load balancer is an acceptable solution since the run time difference of a given building due different input SGM data would not be greater than 5%. The examples given in the next section provide parallel performance of this static load balancer.

2.4 More on load balancing

Achieving a perfect load balancing with static or static-cum-dynamic load balancing may not necessarily reduce the total run time. To achieve the best load balance in the modified IES, a building is considered as the smallest computation unit or the grain size in partitioning the computational load. As a result buildings from one GIS tile are scattered over almost all the CPUs. This leads to longer time for collective IO operations since the number of participating CPUs is large and each CPU contribute relatively small amount of data. One potential solution is to form several CPU sub groups and make only the masters of those subgroups to perform the collective IO. However, this method does not perform well when the data size from each CPU is greater than 100kB (Latham et al. 2004), which is the case with IES data.

Another option is to group the buildings from a GIS tile to several sets with approximately equal total run time and use these sets as the smallest computational grain size. When this larger grain sizes have large total run time, buildings from one GIS tile is scattered over less number of CPUs effectively reducing collective IO time. An intermediate grain size should be chosen to get the optimal behavior since larger grain sizes increase the load imbalance. The major advantage of this method is possibility to further reduce file saving time with concurrent collective IO of several CPU subgroups. Since buildings are scattered over smaller number of CPUs, it could be possible to finds disjoint sets of CPUs which has data from different GIS tiles. These CPU sub groups can concurrently save corresponding data with collective IO on subgroup communicators. When using over 200 CPUs, this may reduce the total run time.

3 ILLUSTRATIVE EXAMPLE

The modified parallel extension of IES SRA module involves message passing only at the very beginning and the very end of a simulation; at the beginning to share some configuration files and building shapes and run time data of each GIS tile and at the end to reorganize and save output data with MPI collective IO functions. The modified IES should have a fairly high parallel scalability as long as there are large numbers of buildings and output is saved to small number of large files. To test the scalability of the modified parallel extension, we simulated all the buildings the GIS tile 09ld171, in which 14,000 buildings are located. The objective of this simulation is to demonstrate IES SRA module, check the scalability against number of CPUs, find out the remaining bottlenecks and roughly estimate the necessary computational resources for a given number of buildings.

3.1 Details of the computation model and the computer environment

Depending on the type of the building, IES can generate input data for any of the available SRA model specified by the user. For this demonstration simulation, structural skeleton of the buildings are automatically generated from the GIS data since the actual structural data was not available to the authors. Dimensions and the spacing of the beams and columns are decided based on the Japanese code of practice for building design. Figure 2 shows a sample of auto generated building shape and the structural skeleton from GIS data. IES provides freedom to assign a preferred SRA model to each building, out of several available models like fiber element model, OCM, DEM, MDOF, etc. However, for this demonstration simulation, all the buildings are simulated with the fiber element model. Fiber element model is simple and computationally light, but still allows significant insight to the seismic response of both the structural members and the entire structure (Spacone et al. part I- II, 1996). All the structures are excited by the same strong ground motion data observed during the 1995 Kobe earthquake. Therefore, this simulation does not reflect the naturally observed complex behavior due to the effects of complicated underground soil and rock structures. IES can to capture these complex behaviors either using observed SGM data at the neighborhood of each building or with synthetically generated high resolution SGM data, with the companion multi-scale SGM simulation module of IES (Ichimura etal 2004, Hori etal 2008).

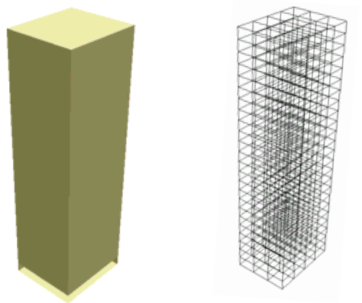

Figure 2 Outer shape of a building and its automatically generated beam column skeleton.

All the simulations are conducted in a commodity Linux cluster made of connecting 8 workstation nodes with a Gigabit switching hub. Each workstation has a 32GB DDR2 600MHz memory and two Quad-Core AMD Opteron 2379 HE processors. Therefore, the current simulation does not reflect the performance advantage of utilizing MPI-IO collective operations; this cluster has neither supporting hardware nor a supporting file system for MPI-IO (Latham et al. 2004).

3.2 Scalability

In order to check the scalability of the modified SRA parallel module, we conducted several simulations, with different number of CPUs. As is seen in the graph of runtime versus number of CPUs in $\log_2$ scale, shown in figure 3, the modified parallel module has almost the ideal scalability. Compared to the performance curve shown in the figure 1, the modifications have significantly improved the parallel performance. When the fiber element model is called as an independent executable with SysV

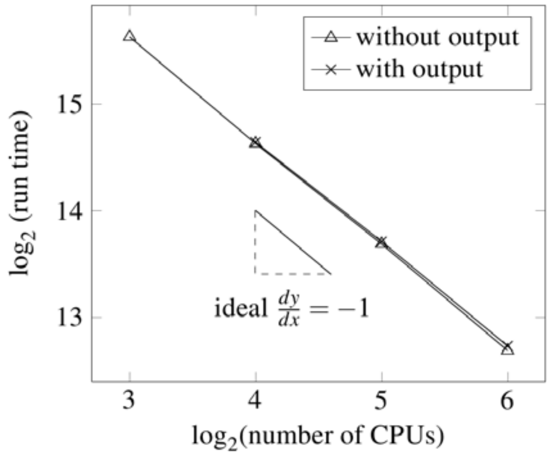

Figure 3 Runtime vs. number of CPUs of the modified IES, in $\log_2$ scale

Table1 Runtimes of 14,000 structures

No. of CPUs	Run time excluding output saving /(s)	Time for MPI collective IO /(s)
16	25820	214
32	13754	218
64	7008	189

IPC resources, it only took additional 30 seconds with 64 CPUs. This indicates that there is no serious penalty of calling a third party executable with SysV IPC resources in parallel environment. According to Table 1 the run time for 14,000 structures is around two hours with 64 CPUs. Also, it has taken around 200 seconds to reorganize and save the 12GB of output data with collective MPI-IO operations. As it was mentioned, the cluster used for the simulation had neither hardware nor a file system supporting collective MPI-IO operations. With supporting hardware and software, nearly 10 times reduction of file IO time can be achieved. Figure 4 shows some snapshot of displacement of buildings.

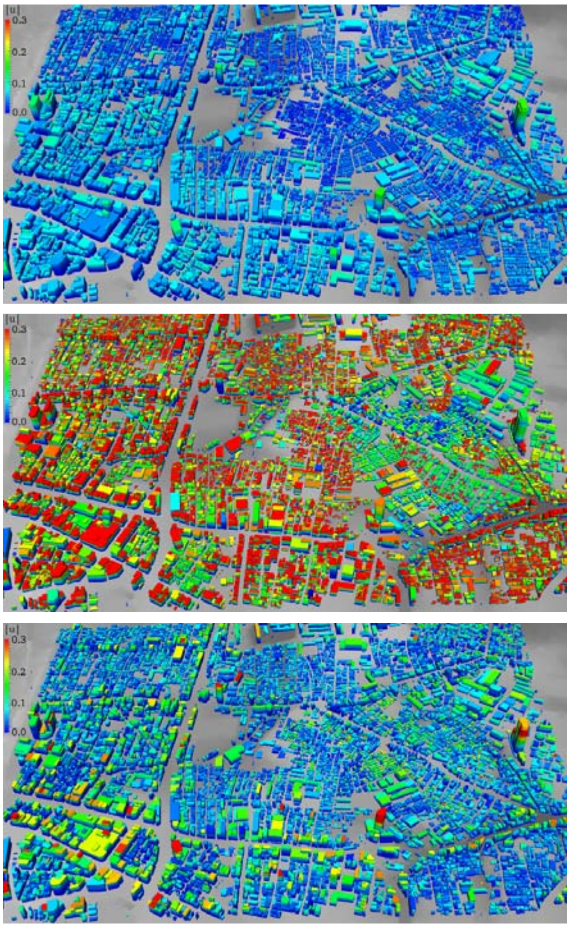

Figure 4 Some snapshots of displacement time history of buildings in GIS tile 09ld171, simulated with fiber element model.

4 CONCLUDING REMARKS

Parallel performance of seismic response analysis module of IES is significantly improved eradicating all the bottlenecks in the former model. The extensive use of temporary files, large number of inter-processor communications, unbalanced workloads and poorly designed output data handling are the identified performance bottlenecks. With a relatively small simulation of 14,000 structures, it was demonstrated that the modified parallel module has almost the linear scalability with respect to the number of CPUs. This near ideal scalability is unchanged even with 12GB of output data saving is included. It was observed that using SysV IPC resources for data exchanging between independent executable does not degrade the parallel performance. According to the demonstration problem presented here, 10,000 15,000 CPUs are needed to simulate whole Tokyo area. In future, increasing the computation grain size used for load balancing and concurrent data saving by multiple groups of CPUs are to be implemented to further improve the parallel performance when using large number of CPUs.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge support from the Japan Ministry of Education, Culture, Sport, Science, and Technology (MEXT) for establishing Global Center of Excellence in the University of Tokyo. This work is done under the Global Center of Excellence for Sustainable Urban Regeneration.

REFERENCES

Tsuyoshi Ichimura, Muneo Hori, Kenjiro Terada and Takahiro Yamakawa, 2004, On Integrated Earthquake Simulator Prototype: Combination Of Numerical Simulation And Geographical Information System, *Journal of Structural Mechanics and Earthquake Engineering,* 9(4): 1-12.

Muneo Hori and Tsuyoshi Ichimura, 2008, Current state of integrated earthquake simulation for earthquake hazard and disaster, *J. of Seismology*, 12(2): 307-321.

Enrico Spacone, Filip C. Filippou and Fabio F. Taucer, 1996, Fiber beam-column model for non-linear Analysis of R/C frames: Part I. Formulation, *Earthquake Engineering and Structural Dynamics*, 25:711-725.

Enrico Spacone, Filip C. Filippou and Fabio F. Taucer, 1996, Fiber beam-column model for non-linear Analysis of R/C frames: Part II. Applications, *Earthquake Engineering and Structural Dynamics*, 25:727-742.

Daniel P. Bovet and Marco Cesati, 2000, Understanding the Linux Kernel, O'Reilly, ISBN: 0-596-00002-2.

W. Richard Stevens, 1999, UNIX Network Programming, Volume2: Interprocess Communications, ISBN 0-13-081081-9.

Rob Latham, Rob Ross, and Rajeev Thakur, 2004, The impact of file systems on MPI-IO scalability, *Lecture Notes in Computer Science*, 3241:87-96.

Kwangho Cha, Hyeyoung Cho, and Sungho Kim, 2007, Performance Analysis of the Subgroup Method for Collective I/O, *Int. J. of Electrical and Electronics Engineering*, 1(8).