Proceedings of the International Symposium on Engineering Lessons Learned from
the 2011 Great East Japan Earthquake, March 1-4, 2012, Tokyo, Japan

A STUDY ON EFFECTIVENSS OF USING OFFICIALS FOR REDUCING PRE-EVACUATION TIME IN A LARGE AREA, BASED ON MULTI AGENT SIMULATIONS

Rithika Dulam[1], Maddegedara Lalith[2], Muneo Hori[3], Tsuyoshi Ichimura[4] and Seizo Tanaka[5]

1) Master Student, Earthquake Research Institute, University of Tokyo, Japan, rithika.d@gmail.com
2) Asst. Professor, Global Center of Excellence for Sustainable Urban Regeneration, University of Tokyo, Japan, lalith@eri.u-tokyo.ac.jp
3) Professor, Earthquake Research Institute, University of Tokyo, Japan, hori@eri.u-tokyo.ac.jp
4) Assoc. Professor, Earthquake Research Institute, University of Tokyo, Japan, ichimura@eri.u-tokyo.ac.jp
5) Asst. Professor, Earthquake Research Institute, University of Tokyo, Japan, stanaka@eri.u-tokyo.ac.jp

ABSTRACT: We developed a multi agent code enhanced with high performance computing (HPC) for simulating mass evacuations during catastrophic events like a tsunami. A numerical simulation that models each individual among the millions is the most viable method of predicting mass evacuation. We studied the effectiveness of reducing pre-evacuation time with the help of officials like policemen, whose duty is to personally encourage the residents to start evacuation immediately. The effectiveness of the official agents is studied by varying their densities and level of influence. It was found that even 0.5% officials can be effective in reducing fatalities in a tsunami.

Key Words: Multi agent simulation, Tsunami, Pre-evacuation time, Evacuation time, Parallel programming.

INTRODUCTION

Natural disasters cause extensive damage to life and property, leading to a situation of chaos. To reduce the impact of the disaster, prediction of mass evacuation for events, like tsunami, earthquake, etc., holds great importance to ensure public safety. Reducing the evacuation time is the key issue in tsunami disasters where the available time is less than 10 minutes for some areas. March 11, 2011 has seen a devastating picture of tsunami in Japan. Real experiments for large area evacuation are dangerous and numerical solutions are the only available solution at hand. There are a number of numerical models available but Multi agent system is suited for the current situation as it models the heterogeneity of human behavior quite well and helps us to produce possible trends of human activities to be prepared for a disaster. According to a survey conducted among the survivors of the 2011 Tohuku tsunami hit regions, between Yamamoto-cho and Minami-sanriku-cho, pre-evacuation time is more than 30 minutes. It has also been reported that comparatively less number of people heard warnings or messages from the local government or the local mitigation centers. This pre-evacuation time depends on many factors like perceived importance, uncertainty of the warning, finding family members, gathering valuables etc. Reducing this large pre-evacuation time is one of the most effective means of reducing total evacuation time. Especially for areas with short tsunami arrival time, reduction of pre-evacuation time is critically important.

Resorting to personalization would help reduce the evacuation time as it has been observed from the survey that many people ignore or do not hear the mass communication messages. Officials of the community, who have complete knowledge of their neighbourhood, could be employed to personally instruct the people to evacuate or inform them of the nearest and safest evacuation center, depending on the intensity of the disaster. We need to identify the density, distribution, functions performed, as well as the legal strength of enforcing the instruction. However, quantitative study of the effectiveness of these factors is difficult for a large urban area; no experiment is feasible for large area evacuation involving millions of people and small area experiments do not render a correct picture due to contagious and cascade effects of mass evacuation. This requires the HPC enhanced MAS with smart and intelligent agents so as to help the evacuees.

The current study of MAS is a part of much larger effort of developing a module to simulate actions taken by individuals and communities in an Integrated Earthquake Simulation (IES). IES is seamless simulation of earthquake wave propagation from source to the site seismic structure response of the buildings in a large urban area and the response of communities (Hori et al). The existing MAS code is developed for an earthquake simulation in crowded areas like shopping malls, train stations etc. Its development is based on the commonly known KISS (keep it simple and stupid) principle in multi agent simulations. Open space navigation needs sophisticated agents as human brain perceives closed and open environments quite differently. Also, there is a requirement for smarter agents with greater abilities like the officials to help the evacuees. For simulation of such smart and sophisticated agents in large number in a large domain, clearly, the current state is insufficient. Hence, a new KISS (Keep It Smart and Sophisticated) principle for the agent development along with HPC must be employed to satisfy the requirements.

In section 2, the methodology of the current model which includes the behavior of the agents, the environment and the generation of the officials is discussed. Section 3 explains about the steps involved in the verification of the model. Section 4 deals about the effectiveness of the officials in reducing the preparation time. Section 5 concludes the paper along with the possible future work.

METHODOLOGY

The behavior of the agents is improved for smooth navigation in open environments. The algorithms developed are discussed below. The key issue tackled is to test the effectiveness of the officials in reducing the pre-evacuation or preparation time of the evacuees. The functions of the officials are also discussed in this section.

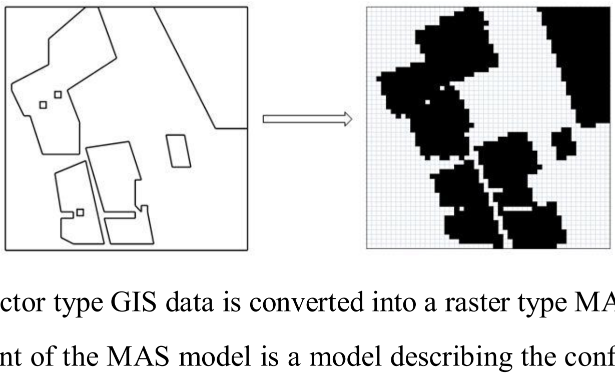


Figure 1: Vector type GIS data is converted into a raster type MAS environment

The environment of the MAS model is a model describing the configuration of public space. The GIS data is automatically converted into the MAS domain data. It is a structured raster type domain having a grid size of 1m^2, consisting of obstacle cells and open cells. The agents are allowed to move around in the open cells avoiding the obstacles. The raster environment used for the current simulation is shown in Fig. 1.

Improved behavior of the Agents

The agent in the model has three basic functions, namely See(), Think() and Move(), which represent the human's most important features. The functions, which are self explanatory, are the vital aspects of the simulation.

The vision and the decision making are improved for agents to navigate smoothly in wide open areas. As mentioned, the former code is designed for simulating evacuation in a crowded environment like shopping malls, train stations, etc. The area influencing the decision, which is the visual distance in the current model, was restricted to a very small radius of 1.5m, since larger radius is insignificant in crowded environments. The vision radius is increased to 30m so that agents can identify various features of their surrounding and navigate smoothly in urban areas. A good and compatible vision and decision making model assures a smooth agent motion, a key for human modeling.

Agent vision model

Visual Perception is an important function of the agent model as the Think() function just analyzes the information generated by the vision. For the current simulations, the vision radius is set to 30m since further increase does not show a significant improvement of agents' movements. An agent scans the 360^o vision field at $d\theta$ intervals, similar to radar, and collects the grid cells in the line of sight. The value of $d\theta$ is set such that arc length with visibility radius is 50% of the grid size (dx) (i.e. visibility radius $\times$ $d\theta$ = 50% of the grid size). For 30m visibility radius, $d\theta$ is set to 1^o. Currently, the agent vision is only horizontal, hence the elevation of the cell is not considered. Fig. 2 shows the vision of the agent and how it perceives the environment.

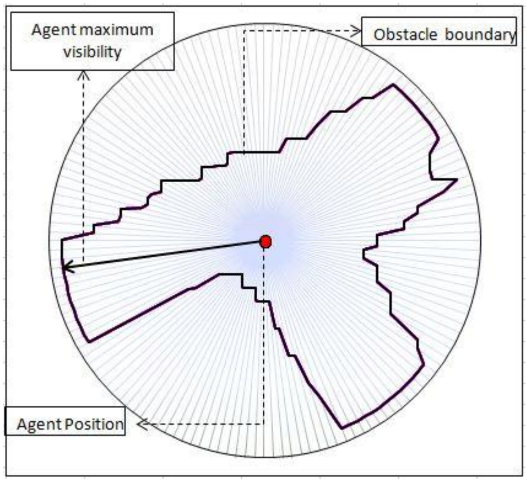


Figure 2: Vision Perception of an agent

Decision making model

The pivot of the agent model is the decision making function. In function Think(), it analyzes the data collected through the vision model and takes a decision, depending on its surroundings and the final destination. The agent scans through the view data, collects all the available paths and chooses the closest path to the final destination vector.

The movement in raster environment is quite difficult, as identifying the features like the shape of buildings, junctions or roads is complicated. The agent can only recognize the open cells and obstacle cells; it is difficult to prevent the agents moving towards dead ends or to block them when moving towards already explored areas. There are many possibilities that an agent loops around in an area for a long time due to the complexity of the urban environment. Such agents are identified and A* algorithm is used to find the path to move out of the looping area. Once, an agent moves out of the trapped region, the former Think () function resumes.

Officials and their functions

The officials are smart agents with greater abilities, like the police, rescue officers, etc. They help people evacuate to safer areas. The main function of an the official is Guide() in which an official asks an agent to start evacuation immediately, using his influence power. The official searches in his visibility, for agents who have not started their evacuation and moves towards them. When the target agent is within the officials influence radius, the official informs him about the situation, the coming tsunami and suggests him to start evacuation immediately. This influence radius is considered as the conversational distance. Various influence radius, 2.5m. 5.0m, 10.0m, 20.0m, 30.0m have been tested and it was found that there is no significant difference in 2.5m and 10.0m; the extra 7.5m travelled by the official to reach the agent is less significant. However, a value of 2.5m has been used for the simulation. In Fig. 3, the movement of the official is shown through the sequence of snapshots from left to right and top to down. We can see the official, the red cone, moving to each of the agents in his visibility and asking them to evacuate. As soon as the agent is in the influence radius of the official, the agent starts to evacuate.

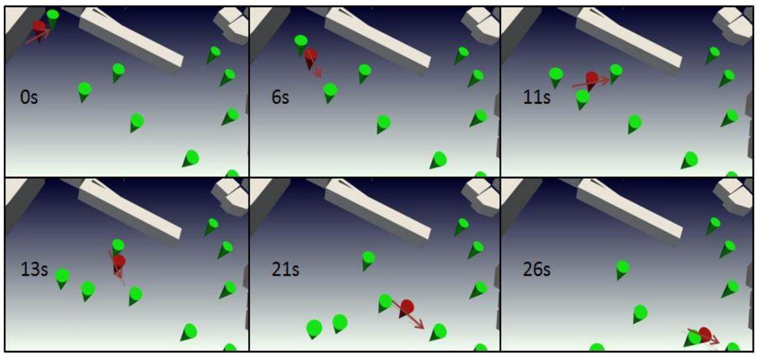


Figure 3: Movement of an official

An object oriented language, C++ has been used to code the current MAS program. C++ is quite suitable for programming MAS, as each component can be coded as an object. The model takes full advantage of the key features of C++ like inheritance, polymorphism, data encapsulation etc.

PARALLEL PERFORMANCE

As suggested previously, for rendering a correct picture of the effects of mass evacuation, simulation of large numbers of people in a huge urban area is required. Simulating 30 minutes period of evacuation with 5000 people, in 1 km^2 urban environment, requires around 5 hours on a single CPU. Hence, the unavoidable requirement of parallel processing is clearly seen. A distributed computing model with Message Passing Interface (MPI) is used in the current program. To balance the work load, the domain is partitioned such that each processor core is allotted approximately equal number of agents. When an agent moves across the partition boundary, he is permanently moved to the corresponding CPU. Once significant load imbalance is observed due to this agent movement among CPUs, the domain is repartitioned to reassign balanced loads to CPUs.

Improving the Performance

Existing MAS based crowd simulations have achieved limited scalability on small number of CPUs. With several strategies, we attained super linear scalability up to 128 CPU cores with 500,000 agents. The main strategies used are: 1) virtual CPU topologies; 2) reducing the frequency of communications for ghost layer updates; 3) hiding the communications and minimizing the data volume; 4) minimize data exchange in repartitioning.

Virtual CPU topologies

We use a 2D-tree algorithm to decompose the domain into square regions with nearly equal number of agents. Although this partitioning strategy does not minimize the communication data volume, the simple geometry makes it easier to handle agent movements between partitions. A major disadvantage of 2D-tree based partitioning is that we cannot take the advantage of communication topology of underlying hardware; the communication patterns between partitions are too irregular. However, the distributed graph topology interface of MPI 2.2 standard provides a more user friendly interface, to better match the communication pattern of the partitions.

Reduction of the frequency of ghost layer updates

This is the main strategy for going beyond linear scalability, which is made possible by maintaining wide ghost regions (overlapping between two partitions). The human agents needs the ability to see far to detect obstacles, other agents and slow moving agent groups. This requires maintaining a wide ghost region, which should be larger than the visibility distance of 30 to 100m. Surely, this requirement significantly increases the communication overhead. However, in human crowd simulations, this wide ghost boundary can be used to gain high scalability. To explain, the terminology introduced in Fig. 4; a partition assigned to a CPU is further subdivided as shown. To avoid slow moving crowds an agent needs to see the other agents at far end of his visibility. Therefore, a small error in the position of agents at the far edge of receive region does not affect the agents inside a partition. This allows skipping several ghost layer updates and maintaining the continuity by executing the agents in the ghost layer in a certain order along with the inner agents and boundary agents. The number of steps to be skipped should be selected to minimize the propagation of any error into a partition from the agents at the far edge of receive region. Depending on the distance of agent's visibility, ghost updating at every 5-10 time step is sufficient (time interval is 0.2s). This significantly increases the scalability parallel extension of evacuation module.

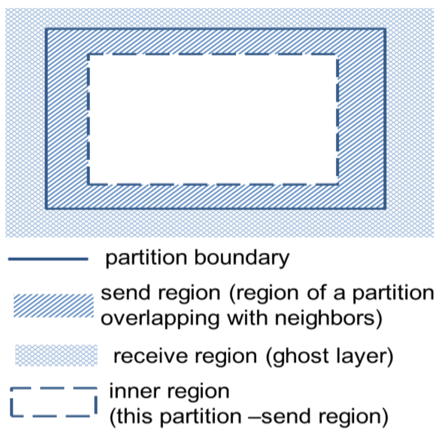

partition boundary
send region (region of a partition overlapping with neighbors)
receive region (ghost layer)
inner region (this partition –send region)

Figure 4: Partitioning of the agents in each CPU

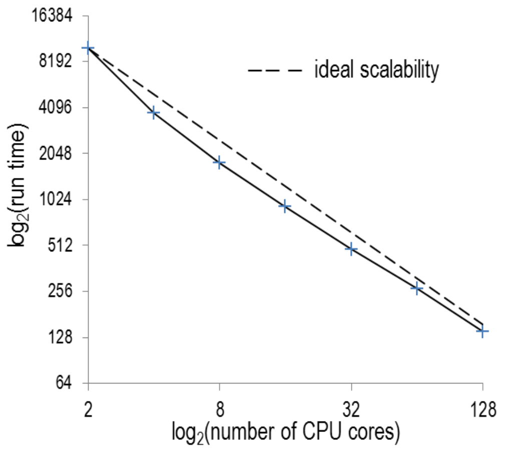


Figure 5: Scalability of the program up to 128 CPUs

Hiding communications and minimizing data volume exchanged

Compared with particle physics type simulations, of similar category, like SPH or N-body, sophisticated and smart agents involve 10 to 20 times data. Further, sophisticated agents have dynamically growing variables like memory of past experiences. To deal with this large data volume, only the recent updates of dynamic data and essential data for ghost updating is exchanged during ghost update. Further, the ghost update communication is hidden by executing the agents in a certain order.

Minimizing data exchange in repartitioning

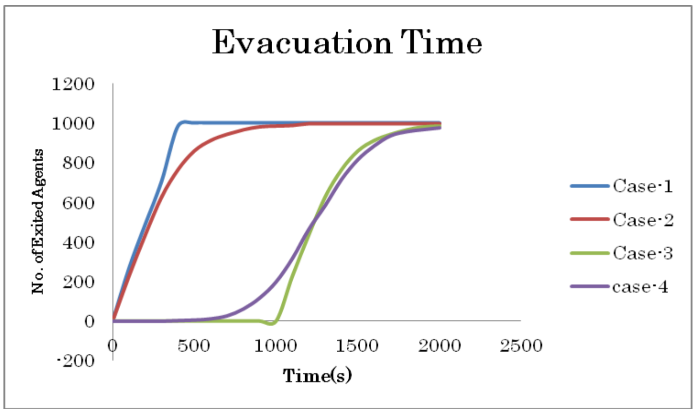
Migration of agents from a partition to another brings load imbalance. When significant load imbalance occur, repartition is necessary to maintain equal workloads. Repartitioning is a very expensive step since human agents have large set of data. With 2D-tree, it is observed that most of the agents remain in the same CPU even after repartitioning, unless MPI_Dist_graph_create() maps the partitions to different CPUs. The repartitioning algorithm detects whether the same partition is assign to a CPU and exchanges only the newly assign agents. The point to note, here is that the interval should be such that it should not affect the movement of the boundary agents of the neighbouring cpu. This drastically reduces the communication overhead in repartitioning, which lowers any performance degeneration due to repartitioning.

Scalability

The effectiveness of the above major strategies was tested by conducting a series of simulations with 500,000 agents in a dense part of Kochi city, Japan. A DELL cluster with QLogic 12200 Infini Band switch and 16 computation nodes, each with hexa-core Intel Xeon X5680 CPUs and 47GB DDR3 memory, is used for the simulations. As shown in Fig.5, the above strategies have even pushed the scalability into the super linear region; super linear behavior is due to the nonlinear reduction of time for an agent to find its neighbors, when the number of agents in a CPU is halved.

VERIFICATION OF THE MODEL

In the process of verifying the simulation results, we have considered the evacuation of 1000 agents in a grid tile, of 800x600 cells, without the obstacles or building data. It is essentially a 1 dimensional problem, as the agents move straight to the exit without travelling any extra distance. Graph 1 shows the evacuation times of the agents considered with different settings; 1) no preparation time and velocity being constant, 2) no preparation time with varying velocities, 3) constant preparation time with varying velocity and 4) varying preparation time with varying velocities. The values of the preparation time were taken from the survey, mentioned in the introduction and are shown in Table 1. As shown in Graph 1, all the four cases produced the analytically expected behavior. This shows the ability of the developed MAS model in producing the desired results.



Graph 1: Evacuation Times without obstacles

EFFECTIVENESS IN REDUCING PRE- EVACUATION TIME OF EVCUEES

As mentioned previously, the main objective of the simulation is to reduce the possibly large pre-evacuation time of the people by employing the officials. The effectiveness of the officials was examined by conducting various simulations explained as under. The basic parameters of the simulation are shown in table [1].

TABLE 1: PARAMETERS USED IN THE SIMULATION

Parameter	Values
Average moving speed of Evacuees [m/s]	
Normal	1.4
Slow	1.1
Standard deviation of moving speed of Evacuees [m/s]	
Normal	0.6
Slow	0.3
Moving speed of officials[m/s]	3.0
Average preparation time [seconds]	1000
SD of preparation time [seconds]	240

Problem Setting

The current simulation is conducted on an urban city environment of Kochi city (GIS Tile ID: O-16); Fig.6 shows the domain used for the simulation.

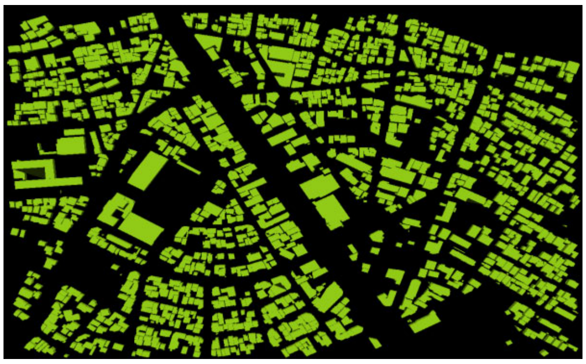


Figure 6: A dense urban grid tile of Kochi city [O-16]

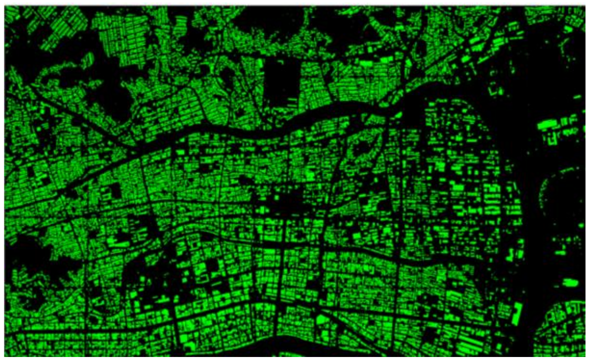
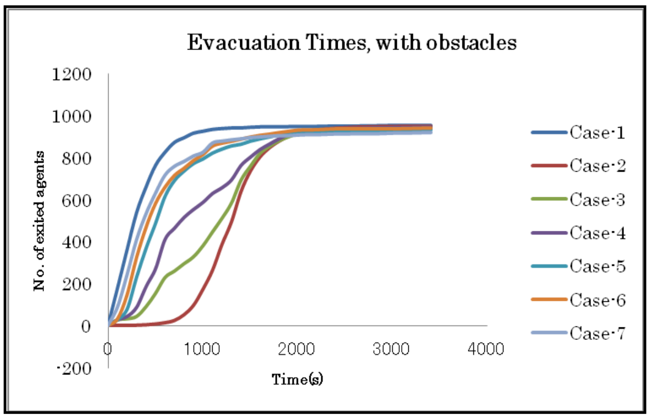


Figure 7: Urban Kochi city, Tile L11- R15

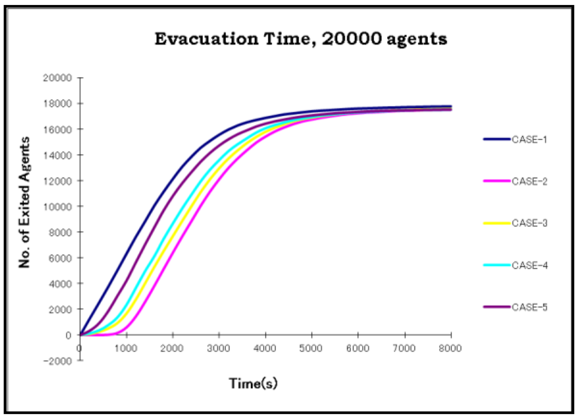
The developed HPC enhanced MAS code can simulate up to millions of agents. However, a small problem of 1000 agents in an area of 0.48 km^2 (a grid of size, 800x600 cells) is considered in this preliminary study. The density of officials is increased from 0.5%, 1%, 3%, 5% and 10%.



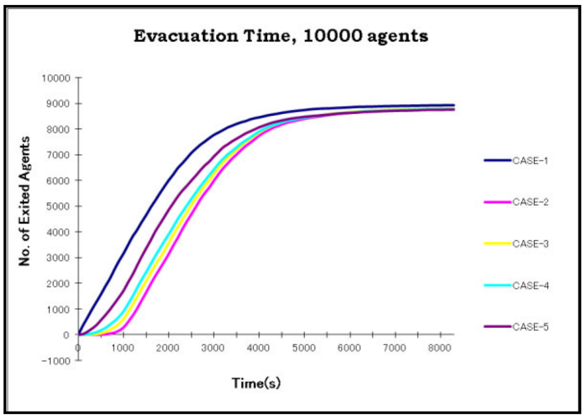
Graph 2: Evacuation time with obstacles

Graph 2 presents the cumulative evacuation time with varying velocity and varying preparation time. Case-1 is the ideal, without preparation time, case-2 to case -7 have varying preparation time with increasing number of officials from 0%, 0.5%, 1%, 3% 5 % and 10% respectively. We can observe that the usage of 0.5% officials has significant effect on the evacuation times of the agents. The irregularity of each graph shows the complexity of the problem, where more than one parameter, like speed, distribution of agents, preparation time and complexity of the environment, is in question. The evacuation time has shown considerable difference with every increase of number of officials. It can also be noted that the usage of 3%,5% and 10% have obtained almost similar results, stating that a usage of 0.5% officials has shown considerable improvement.

We have considered a larger domain of 16.5 km^2 with 35 grid tiles (L11 - R15) of Kochi city (Figure 7), with 10000 and 20000 agents respectively. The density of officials is increased from 0.5%, 1%, 3%, as we have observed that 5% and 10% officials does not show much difference.



Graph 3: Evacuation time with 10000 agents in L11 - R15 grid



Graph 4: Evacuation time with 20000 agents in L11 - R15 grid

It can be observed from Graph 3 and Graph 4 that the behavior is similar to 1000 agents case (Graph 2) with each increase of the percentage of officials. Hence, in a mass evacuation of 20000 people, it can be stated that employing officials is quite positive.

CONCLUSIONS

A HPC enhanced MAS code is developed for simulating mass evacuation in a large urban area. We studied the effectiveness of reducing pre-evacuation time, using officials to personally instruct the residents to evacuate immediately. The officials are employed to personally go to an agent and ask him to start evacuation immediately. This behavior of the officials has resulted in quite significant decrease in the total evacuation times of the evacuees. An optimum level of 3% officials is found to show satisfactory results almost similar to using 10% officials. A similar effect can be expected with using 0.5% officials with other factors which help in reducing the evacuation time. Since simulating large urban area is computationally intensive, we enhanced our code with HPC and attained super linear scalability up to 128 cores with half a million agents. The Evacuation time can further be reduced by improving the functions of the official and by employing the resident leader agents who have less influence power but can help in passing information to the evacuees.

REFERENCES

M.Hori, T.Ichimura and K.Oguni (2006), “Development of Integrated Earthquake Simulation for estimation of strong ground motion, structural responses and human actions in urban areas”, *Asian Journal of Civil Engineering (building and Housing)*, Vol.7, No. 4, 381-392.

M.Hori, “Multi Agent Simulation for evacuation process analysis”, *Introduction to Computational Earthquake engineering*, Chapter 14. 317-358.