X10-based Massive Parallel Large-Scale Traffic Flow Simulation

Toyotaro Suzumura[1,2], Sei Kato[1], Takashi Imamichi[1], Mikio Takeuchi[1]
Hiroki Kanezashi[2], Tsuyoshi Ide[1], and Tamiya Onodera[1]

[1] IBM Research – Tokyo [2] Tokyo Institute of Technology

Abstract

Optimizing city transportation for smarter cities can have a major impact on the quality of life in urban areas in terms of economic merits and low environmental load. In many cities of the world, transport authorities are facing common challenges such as worsening congestion, insufficient transport infrastructure, increasing carbon emissions, and growing customer needs. To tackle these challenges, it is highly necessary to have fine-grained and large-scale agent simulation for designing smarter cities. In this paper we propose a large-scale traffic simulation platform built on top of X10, a new distributed and parallel programming language. Experimental results demonstrate linear scalable performance in simulating large-scale traffic flows of the national Japanese road network and a hundred of cities of the world using thousands of CPU cores.

Keywords X10; Intelligent Transportation Systems; Large-scale simulation; Supercomputer

1. Introduction

Optimizing the behavior of all city resources such as the motion of people and vehicles, logistics, and energy contributes to humanity at large. Needless to say, such optimization efforts present enormous business opportunities in terms of intelligent infrastructure management, but they can also be useful to normalize a city that has lapsed into malfunction through congestion or disaster, and they can eliminate wasteful consumption. IBM Research specifically examined the optimization of traffic flows in a city as an important research subject. The modeling of dynamic traffic flows, i.e., traffic flows changing over time, is a historied discipline that has been studied actively since the 1970s. It has two major approaches in general: one approaches based on time-series prediction technology, and the other approaches based on simulation.

Historically, almost all related studies have been concentrated on the former. However, the latter approach has increased in relative importance since computer resources improved rapidly from around 2000. The greatest reason is that simulation can offer the function of scenario analysis: e.g., "if this route is closed at a certain instant, then how will traffic flow change in time?", or "what will happen if the motion control parameter of this signal is modified?" This requirement is indispensable for optimal urban design.

IBM Research Tokyo has conducted research for several years on the IBM Mega Traffic Simulator (Megaffic), a traffic flow simulator that employs agent modeling technology. An agent simulation enables flexible scenario analysis as described above, but in compensation, it presents disadvantages of very high calculation costs. For that reason, detailed traffic flow simulation on a city level has been regarded as impossible to date.

This paper presents a proposal of an approach of parallel distributed processing using X10 language to break through the problem of the calculation load that is imposed by conventional agent simulation. The construction of this paper is the following: Section 2 presents a description of the overall structure of Megaffic and introduces the mathematical modeling component of a traffic system, which is the largest characteristic of Megaffic; Section 3 explains the outline of XAXIS, an agent simulation platform using X10 language, and a XAXIS-based traffic simulation platform; Section 4 introduces results obtained experimentally using TSUBAME2.0, and finally concludes this paper.

2. Megaffic: Traffic Flow Simulator

Conventionally, a static model such as the user equilibrium assignment model has been employed for evaluation of traffic policies [6][7]. Such an approach that allocates transport demand to each link can calculate each link traffic and saturation ratio at an intersection using allocation principles. It excels in predictability of daily traffic volume. However it cannot estimate the dynamics that are important for traffic policy evaluation such as generation and dissolution of congestion, transport demand that varies among time zones, and signal indication. The analysis of dynamic traffic flow varying in time and space like congestion requires expression of the dynamics of vehicles that form traffic flow on a road network. This paper addresses how a driver is modeled to express this dynamics in Megaffic, and how such a model is connected to traffic flow analysis.

2.1 Outline of Megaffic

The authors have been developing a traffic flow simulator, Megaffic, based on component technologies presented in Fig. 1, to express the dynamic traffic flow that is indispensable for traffic policy evaluation and to achieve effective traffic policy evaluation. Megaffic adopts a multi-agent system for traffic flow modeling. Each vehicle is modeled as an autonomous agent. Multi-agent systems became prominent against the background of distributed artificial intelligence in the 1980s, and have been adopted for distributed cooperative problem-solving or complex system simulation. This multi-agent system will generate feedback that traffic flow has emerged as a result of interaction between agents and macroscopic traffic flow affects an individual driver. Consequently, it is expected that dynamic traffic flow that varies in time and space can be treated.

Such an approach is related to the following three issues:
1) how to build a detailed driving behavior model incorporating the dynamics of the dynamic traffic flow described above,
2) how to conduct a high-speed multi-agent simulation to carry out what-if analysis (an analytic method that evaluates result according to various assumptions) on a city traffic policy, and

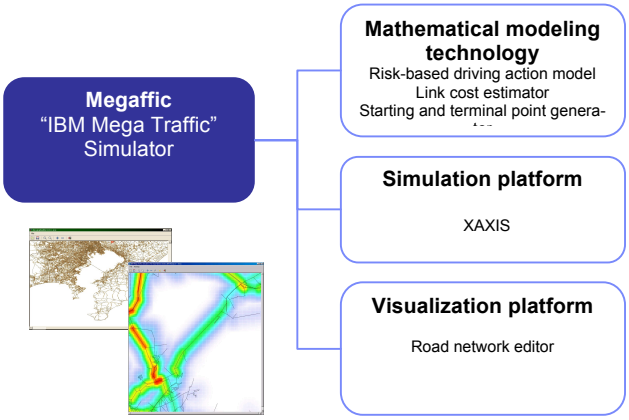

Figure 1 Construction of Megaffic.

3), which graphical user interface (GUI) is efficient for traffic policy evaluation using a microscopic traffic flow simulator.

The authors have developed three component technologies to resolve these three problems; mathematical modeling technology, simulation platform, and visualization platform, as presented in Fig. 1.

Conventionally, in construction of the driving behavioral model of a driver, especially a route choice model, a linear expected utility function is defined and the weight of each utility is set up, such as the number of right and left turns, fees, travel times, and distances. Then the best weight parameter is determined to reproduce the present condition. Such calibration is not efficient because it requires repetitive simulation, and no understanding of the weight obtained is acquired either. Therefore, the authors adopted an approach to set up these parameters automatically, in which probe car data that are data of position information related to vehicles are analyzed, and the cost distribution of each link is estimated. Then based on this link cost distribution, the parameters of a driving behavior model considering risks are extracted automatically.

Trade-offs of adopting a highly mathematical driving behavior model constructed using such an approach include calculation costs for each agent's decision making in the case of simulating traffic flow in the road network of the whole city. Accordingly, the authors have developed a general-purpose platform for the decision-making of an agent using a parallel distributed programming language X10. This language allows users not only to conduct runtime multi-agent simulation but to exploit multi-agent simulators for such as evacuation behavior efficiently in case of an earthquake and population market using this development platform.

Visualization technology is a technology component related to the third issue. The positions of all vehicles in the whole city at every moment are computed in microscopic traffic flow simulation. However, these huge simulation results are of no use for city road planners. It is necessary instead to offer proper GUI for them for efficient road policy evaluation. The authors therefore provide a tool that visualizes the sequential motion of vehicles first obtained as a result of a traffic simulation. In addition, the road network editor (Fig. 1 lower left) is software that enables us to edit objects that form a road network on a computer, such as a road and an intersection. This enables what-if analyses in cases involving changes such as a newly constructed highway or modification of signal indication parameter at an intersection.

3. XAXIS : Massive Agent Simulation Execution Platform

X10-based Agent eXecutive Infrastructure for Simulation (XAXIS) is an executive operation platform of a massive agent simulation mounted using a parallel programming language X10 [2][4]. The outline of X10 is described. Then the massive agent simulation execution platform XAXIS built on X10 is overviewed.

3.1 Outline of Parallel Programming Language X10

X10 is a novel parallel distributed programming language that IBM Research is developing. It enables development of highly efficient applications at high throughput in a distributed system in a mixed environment with various models comprising multiple cores, accelerators, etc. In an execution environment equipped with many execution cores, it is important to demonstrate the parallelism and memory configuration to programmers. X10 offers a global address space that is partitioned into multiple places, called Partitioned Global Address Space (PGAS). A place is the abstraction of the locality of memory, and it typically corresponds to one computer. Activity can be generated dynamically in each place as an asynchronous execution subject equivalent to a lightweight thread. Activity can be generated using the syntax of an async sentence, and can be moved to other places using a sentence. Conventionally, to achieve a simulation using a massive computer with each node comprising multiple cores and being combined in the network, the Message Passing Interface (MPI) is used as a messaging mechanism among nodes. Programming models such as OpenMP and POSIX thread are employed for thread parallel in the 1 node. However, the concept of "place" and "activity" of X10 enables one to build an executive operation system on a massive computer cluster using a unified programming model at high throughput.

3.2 Architecture of XAXIS

Figure 2 shows the XAXIS architecture, the massive simulation execution platform with X10 mounted. An agent manager manages agents asynchronously executed by the activity of X10 in each place in an agent process of a simulation. In case of communication with agents in other places, a place of which the agent manager is in charge is chosen based on an agent's identifier, and a communication message is sent thereto. This communication message is hidden from users, and simulation developers are expected to concentrate only on mounting the logic of the agent. A message that is developed using X10 can be transmitted and received in XAXIS without explicit message communication in a distributed computer, but by calling the method of the agent manager in other places using at syntax. A mechanism for communication between places using activity that implements an asynchronous thread and at syntax enables us to mount the simulation execution environment itself easily.

However, because traffic simulation Megaffic is originally mounted on the agent framework IBM Zonal Agent-based Simulation Environment (ZASE) [1] described by Java, it is necessary that Megaffic be operated transparently on XAXIS. For that reason, we offer the same API as a programming API supplied by ZASE on XAXIS, so that applications on the execution platform of an agent simulation can be executed without modification. Moreover, X10 language offers a mechanism that compiles into C++ and Java codes. It is also one advantage of X10 because agent simulation codes in the existing Java written on ZASE can be executed transparently on XAXIS using its mechanism of translation to Java code.

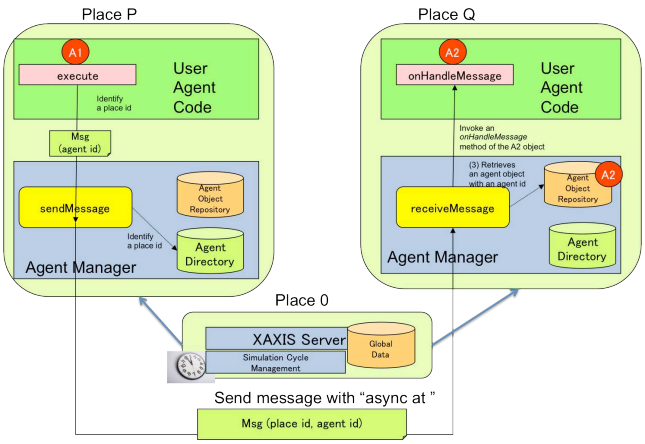

Figure 2 XAXIS Architecture.

3.3 Mounting of Traffic Simulation Megaffic on XAXIS

An important decision in mounting Java-based traffic simulation Megaffic on XAXIS is on which unit the activity of X10 is mapped. The present mounting was conducted by mapping not a vehicle but an intersection itself on the activity of X10. An intersection that operates as an activity controls vehicles (agent) on the roads flowing therein, such as forward movement of vehicles and switching multiple lanes. Then asynchronous processing is conducted so that execution waits until processing of the vehicles is completed, where all the intersections (activity of X10) take charge of every execution step corresponding to a time instant in a simulation. The termination of processing of all generated activities can be postponed for the X10 syntax of finish in mounting.

4. Massive Traffic Flow Simulation using the TSUBAME 2.0 Supercomputer

A traffic simulation was conducted on XAXIS using the TSUBAME 2.0 supercomputer at the Global Scientific Information and Computing Center, Tokyo Institute of Technology, and the performance evaluation of XAXIS was conducted. TSUBAME 2.0 is a supercomputer representing Japan, ranked fifth in the world in the Top500 in the supercomputer ranking and third in the world in Graph500 as of November, 2011. It comprises 1,442 nodes as a whole. Each node has an Intel Westmere-EP 2.93 GHz processor and 52 GB of memory.

4.1 The Nationwide Japanese Road Network

The entire Japanese road network is used as a road system for experimental data of the massive traffic flow simulation, which includes 993,731 intersections and 2,552,160 roads (link connecting two intersections). The road network consists of one-dimensional meshes that divide the road system into 128 meshes, and two-dimensional meshes that further divide the one-dimensional mesh code. Moreover, the identifier of each intersection consists of 13 digits, whose first 4 digits represent the primary mesh code, the next 2 digits denote the secondary mesh code, and the lowermost 7 digits are characteristic values for each intersection in the secondary mesh.

4.2 Estimation of Execution Time in One Node

TSUBAME 2.0 carries 52 GB of memory per node, and can store the entirety of Japan's road networks in memory. First, a simulation was conducted with 24 threads using one node (hyper-threaded, 24 CPU cores). Its execution performance in one node was measured. To estimate the pure system performance, real data, which contain trip representing starting points to terminal points for each vehicle actually passed along, were not employed. Instead, the following three patterns of trips were generated artificially: (a) a trip whose starting point and terminal point exist in secondary mesh codes, (b) a trip whose primary mesh code is the same whereas secondary mesh code differs, and (c) a trip whose starting point and terminal point exist in different primary mesh codes. Although these trips are categorized respectively as short-distance, medium-distance, and long-distance trips, no significant performance difference was observed in execution times.

Then the numbers of steps and trips were varied within 1,000–20,000 and 10,000–200,000, respectively, and execution times were calculated, to compute the total estimated time for performing a simulation for one day using one node. Regression analysis was conducted using these results for the relation of execution time and the number of steps and trips. A relation was obtained as execution time (s) = 0.0197 × (the number of steps) + 1.342 × (the number of trips) - 350. This regression equation suggests that a simulation for 24 hr takes 55.83 hr, assuming that a simulation for 24 hr requires 86,400 steps and assuming that one step comprises 50 trips.

4.3 Acceleration by Space Partitioning

Next, acceleration was tried by partitioning the road network in Japan into multiple domains. The graph partitioning tool METIS [5] was used for partitioning the network into 100 domains so that intersections were distributed equally to each domain. One group (corresponding to one partitioned domain) is allocated to one node on TSUBAME. One group consisted of about 10,000 intersections in average, and parallel distributed processing was executed using 100 nodes and 1,200 CPU cores in total. A simulation was conducted in conditions of 3,600 steps (corresponding to 1 hr) and 180,000 trips. Figure 3 shows the result, where the horizontal axis denotes the identifier of each group and the vertical axis represents the execution time in seconds. The simulation was completed within 300 s for almost all domains except for some groups. The average execution time for the whole groups was 269.95 s, which corresponds to 1.73 hr for a simulation for 24 hr using 100 nodes. Compared with the execution time estimate of 58.8 hr with one node in the preceding chapter, 33.9 times greater acceleration was achieved.

More execution time was necessary in some groups in large cities, even though the numbers of intersections and roads were distributed evenly among others. Such examples include the following: the whole of Osaka in Group 5; the southeastern part of Nagoya city, Aichi in Group 12; Sapporo city, Hokkaido in Group 46; and Fukuoka city, Fukuoka in Group 72. This increase in execution time is attributable to enhanced traffic volume density per road and subsequent load increase at intersections processed as agents. This issue can be improved by further partitioning of these domains using METIS. TSUBAME 2.0 is a supercomputer comprising 1,440 nodes (17,280 CPUs). Therefore, further fine-grained partitioning of the Japan nationwide road network will enable completion of a simulation of activity for 24 hr within several minutes.

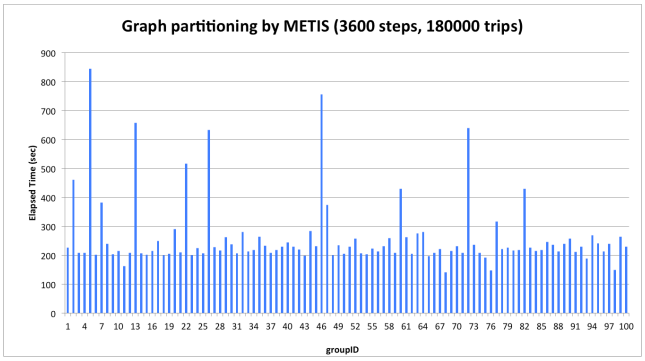

Figure 3 Execution time of each group after partitioning.

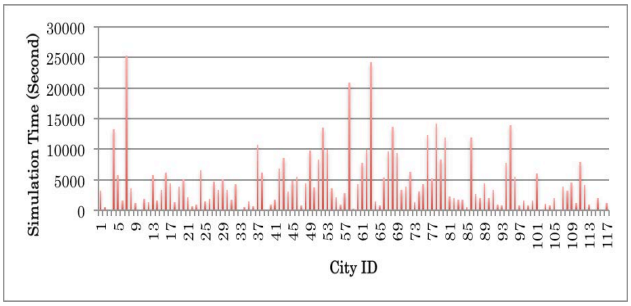

Figure 4 Simulation execution time for cities of the world.

Figure 5 Visualization of simulation for Rio de Janeiro, Brazil

4.4 Experiments Using Road System of Cities in the World

Simultaneous simulations for the road networks of 127 major cities throughout the world, such as Singapore, London, and Mexico City, were also conducted using Open Street Map [3], the worldwide geographic information data, in addition to the whole Japanese road network. The Open Street Map is open data provided under the Eclipse Public License, and almost all global maps are now available. In this experiment, a traffic simulation for each city was conducted simultaneously on each computing node using 1,524 cores (127 nodes). Figure 4 presents the result of this experiment with 1,000 steps and 50,000 trips. The horizontal and vertical axes respectively present the ID of cities in the world and execution time. The scale of a city road network varies in this case, containing from tens of thousands to 2 million intersections. Computations for a small city were completed within 10 min, although those for a large city took a correspondingly longer time. It is possible to accelerate a simulation for large cities taking a long time by domain partitioning such as the whole Japanese road network in the future. A screen shot of simulation results for Rio de Janeiro in Brazil is displayed with the visualization tool of Megaffic as an example in Fig. 5, with which the movement of vehicles in time and space can be browsed.

5. Conclusion and Future Direction

This paper introduces an execution platform environment where X10 language is applied to a massive simulation to break through the problem of enhanced calculation load in the conventional agent simulation. The contents describe the experimentally obtained result of performance evaluation using the TSUBAME 2.0 supercomputer with road networks throughout Japan and major cities all over the world. A performance evaluation using the whole Japanese road network indicated that a simulation for 24 hr with 1 node of TSUBAME took 55.83 hr, although it took only 1.73 hr with the execution environment of 100 nodes (parallel computation with 1,200 cores). No execution platform operating in such a massive environment for an agent simulation has ever existed. In addition, a massive experiment conducted in this study is the first trial. XAXIS itself is a general-purpose agent simulation. It will enable the conduct of various simulations comprehensively in the future, such as its practical use in consumer behavior in business, cyber security, and biological simulations at a molecular level.

References

[1] Yamamoto, G., Tai, H., and Mizuta, H. A Platform for Massive Agent-based Simulation and its Evaluation. AAMAS 2007, 900-902.

[2] Kawachiya, K. X10: A Programming Language for Multicore Era. Information Processing, 52(3) (2011), 342-356.

[3] Open Street Map, http://openstreetmap.jp/

[4] Saraswat, Vijay A., Sarkar, Vivek, and von Praun, Christoph. 2007. X10: concurrent programming for modern architectures. In Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming (PPoPP '07). ACM, New York, NY, USA, 271-271.

[5] Karypis, G., and Kumar, V. Multilevel k-way Partitioning Scheme for Irregular Graphs. Journal of Parallel and Distributed Computing, 48 (1998), 96-129.

[6] Florian, M. A Traffic Equilibrium Model of Travel by Car and Public Transit Modes. Transportation Science, 11(2) (1977), 166-179.

[7] Wardrop, J.C. Some Theoretical Aspects of Road Traffic Research. Proc. Institute of Civil Engineers Part 2, 9 (1952), 325-378.