The Development of Scalable Traffic Simulation Based on Java Technology

Narinnat Suksawat, Yuen Poovarawan, Somchai Numprasertchai

The Department of Computer Engineering
Faculty of Engineering, Kasetsart University
Bangkok Thailand
Telephone 9428555 ext 1420
Email: narinnat@thailand.sun.com, yuen@ku.ac.th, snp@ku.ac.th

Abstract

Distributed computing has become a very popular style for large-scale simulation software since the continuously increasing of personal computer's performance and network bandwidth. In this paper we described the development of Scalable traffic simulation and measure its performance. The simulation designed in this research is to simulate vehicle behavior at the individual traffic component level, in which we can monitor and measure vehicle behavior individually and be able to display traffic data in real time graphics mode. The distributed computing environment used consists of a set of computer nodes running Java simulator module connected and working together in a master/slave model. Java Message Service (JMS) is used to implement message passing between computing nodes. These nodes are working in a loosely couple parallel computing style. Performance testing is focused in two conditions; The first is fixed number of computing node where we measured the *Computing time/Number of Simulation objects*, Second is fixed number of simulation objects we measured the *Computing time /Number of Computing node*.

1. Introduction

Many years ago parallel or distributed computing was developed and has brought substantial benefits to the area where scientific problems are so complex that solving them requires extra powerful computing systems to solve such a problems as quantum chemistry, meteorology, fluid dynamic and many others. Traffic simulation is a very interesting problem domain especially when conducting simulation at a level of individual components since the complexity is measured in terms of data sets that are often too large for a processor to hold them in memory and process at once. Using single processor system it is not possible to control the movement of a very large number of vehicles along roads without sacrificing simulation performance. The situation becomes worse if the data size becomes larger later. Another way to relieve this trouble is designing the simulation system to be more scalable by using distributed computing.

This implies that by harnessing additional processors in parallel and decomposing the problem domain into sub-domains we can speed-up the simulation. Researchers at University of Edinburgh have proved the benefits of parallel computing. Their traffic simulation can move 200,000 vehicles spread over 7,000 roads at real time rates [1]. Data decomposition is the next important issue when designing parallel algorithms. Parallel algorithm initially distributes data across processors and makes them responsible for their portion. Thus, parallel algorithms for traffic simulations need to parallel

data such as road network geometry, road network topology and initial positions of vehicles. In [1] each processor is statically assigned a queue, the parallel item of data, which is then associated with all the lanes that comprise one direction of the road link. The relation processor-queue remains unchanged during the simulation run. As the number of vehicles on the particular processor vary between simulation frames this can result in situations where some processors have only a few or even none vehicles to control if all vehicles are outside the processor's region. If such situation occurs load imbalance increases and usage ratio of the processor falls. This led researchers at the University of Maribor to conduct researching for Algorithm for adaptive load balancing in parallel traffic simulation [2]. The parallel environment used in their environment consists of a set of workstations nodes connected in a network. PVM network programming environment is used to implement message passing between workstations.

PVM and MPI have long been used in the development environment for parallel computing. However since they are poor development tools, platform specific and difficulties to use in concurrent programming. We were motivated us to look for new development tool. We have found that Java based technologies are now more sophisticate in terms of standardize, scalability and platform independence.

Our computing environment is a set of computers node connected together in a loosely couples style using Java Message Passing to make a communication between nodes. These computers are truly platform independent and can be integrated from UNIX, Windows or even Macintosh based nodes.

In the remainder of this paper we first give an overview of essential requirements in designing traffic simulations. Then, we describe parallelism of traffic data, partitioning a traffic subset and implementation. Finally, we present the test results and the speed up ratio measured from our testing environment.

2. Traffic Simulation

Traffic simulation has been in existence and developed for many years. Their main purpose is to simplify models of traffic flow in order to produce results within timescales. Typically, there are two major models used to represent a traffic flow quantity. One is *macroscopic* and another one is called *microscopic*. *Macroscopic* model represents traffic flow in a particular road as a single quantity. Unfortunately, such models do not properly represent real traffic behaviors in congested situations, and do not reproduce the fluctuating nature of real world situations [1]. *Microscopic* simulation enables more accurate study of congestion formation and dispersion and emphasizes the insight into the nature of road traffic flow [2]. During each time step, the vehicles are moved towards its destinations, and from road to road if necessary, as in real-life. This research is used a kind of *Microscopic* model also.

2.1 Traffic Data

To simulate traffic environment and the movement of vehicle, we need traffic data. Traffic data is information that describes attributes of traffic components individually. In this research we used a simple form of these components. It consists of vehicles, link, junction, intersection and nodes:

Vehicle, each vehicle has individually physical property which is includes:

- Vehicle type
- Width
- Length
- Maximum speed
- Number of Lanes

Besides, it has it own behavior when moving which consists of:

- Cruise
- Accelerate
- Decelerate
- Following
- Stop

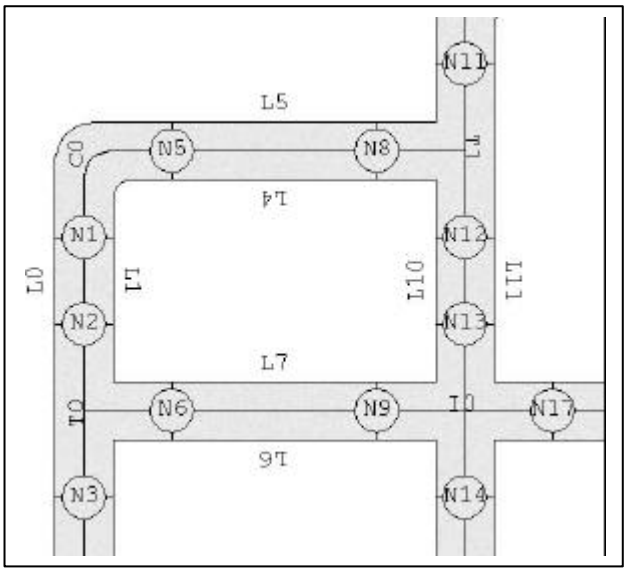

Figure 1: Traffic Data

Node is used to represents the conjunction between two components.
Corner is the same function with road except the representation of visualizes style.
Link, to represents the road in one direction between two nodes.
T-Junction, to represents the T-junction style between roads with traffic light control.
Intersection, to represents the intersection between roads with traffic light control.

All of these traffic components are able define a physical property such as:

- Width
- Length
- Number of Lanes
- Constraint speed

- Traffic signal control for T-Junction and Intersection

Figure 1. Illustrates how traffic components connected together.

2.2 Data Parallelism

We can achieve the parallelism of traffic data by decomposes in to be several data subsets. Each processing node can process each data subset concurrently. To partition traffic data, we may consider data from different location and traffic behaviors. Each subset should contain a group of traffic components, which have a high-density traffic around the center of subset area and should have a very light traffic around bound. This will improve the simulation performance in order to reduce traffic of vehicle interchanging in the network between processing nodes.

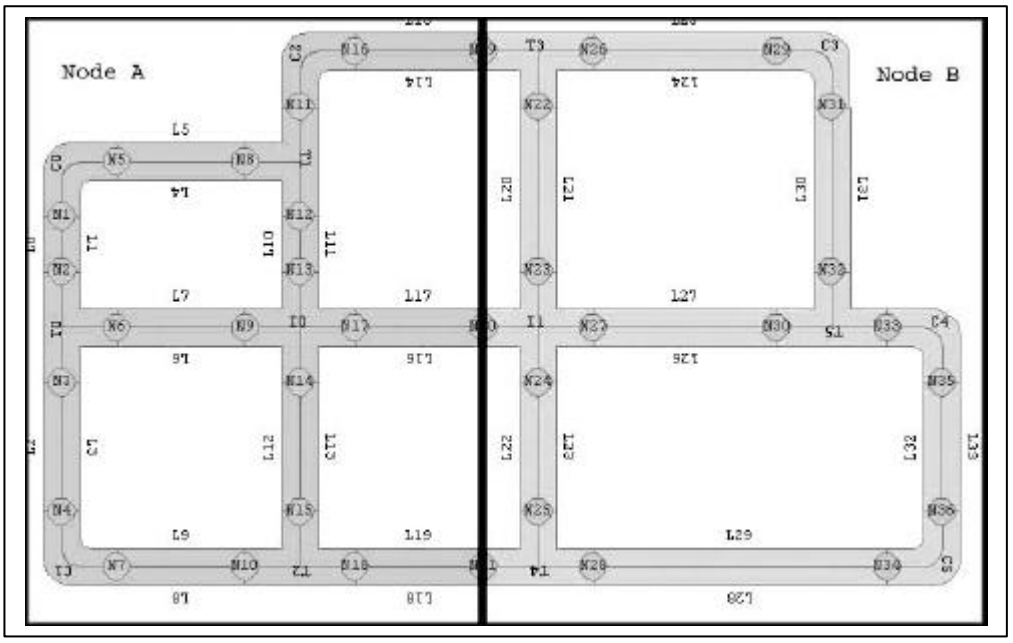

Figure 2: Two subsets partitioning

3. Implementation of The Scalable Traffic Simulation

The environment used in our research consisted of computer nodes connected together by a Fast Ethernet based switch. Our distributed environment can be considered as a MIMD system. Each computer node has a copy of simulation program and has it own traffic data subset to execute. It dispatches data with other computer nodes, which we called Processing nodes. One node has to manage the simulation and synchronize all nodes to work discrete time, which we called Manager node. The Manager node provides user interface and display the information summary of traffic situation.

JMQ is used to provide communication in among of these computing nodes. The communication is included controlling signal, synchronization signal and objects interchange.

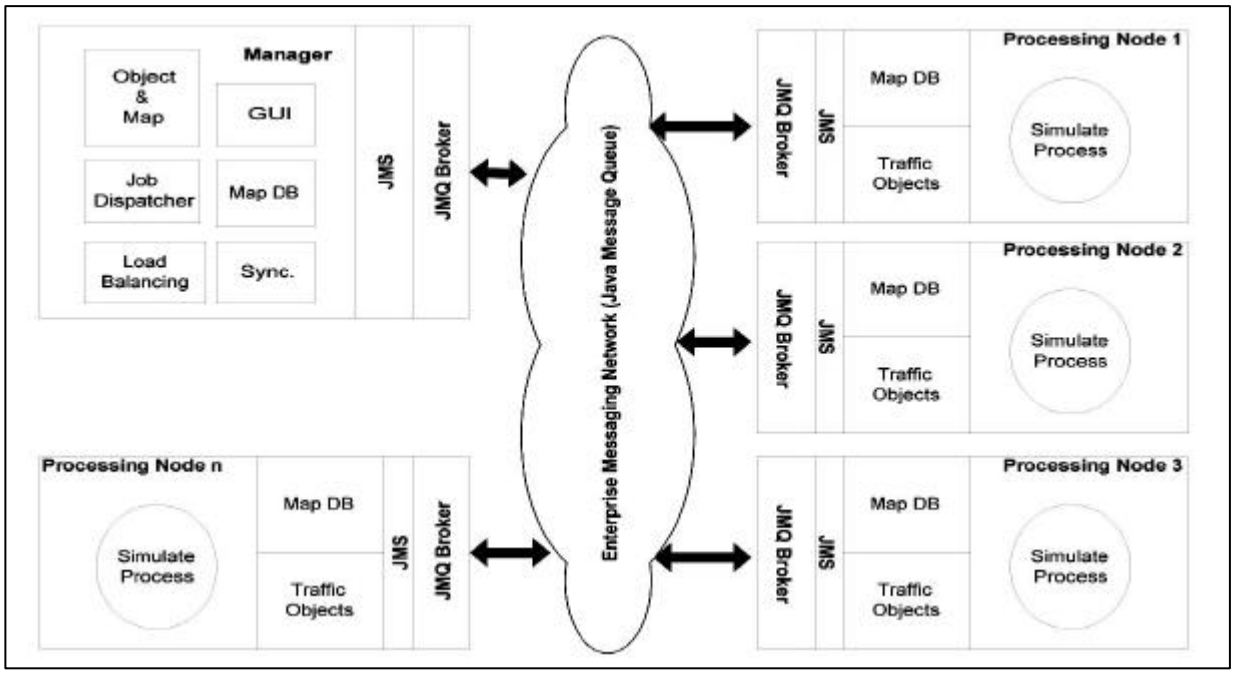

Figure 3: Architecture of Scalable Traffic Simulation

3.1 Manager Node

In order to start simulation initially, manager node first read data needed for the initialization of simulation environment from a configuration file. The configuration consists of road network, routes and data subset for each processing node. After finish initialization, Manager then sends the initialization data to processing nodes. It will remain idle until received ready confirmation from all Processing nodes.

Every step of simulation the manager node will send SYNC signal to the processing nodes, all processing node will process one step of simulation after receiving this signal. After finish the step it will send RESYNC signal back to the manager node along with other additional information such as number of vehicles.

Manager node is able to display road network and vehicle at any position along the road map in a real time. However it depend on the purpose of simulation. If we want to maximize the simulation performance, the visualization should be disabled because it will consume a huge amount of network bandwidth.

3.2 Processing Nodes

Manager node manages processing nodes by using JMS to deliver control and synchronize messages. First, Processing process does initialization and receives basic information from the host that should be present before the simulation can be started. Initial vehicles in network will first simulated at processing node by getting quantity from manager node. After finish the vehicle generating and initialization it will send a complete message to manager. On simulation time, Manager will send synchronize message to notify all processing nodes to start the processing of one step of simulation. During the step of simulation, if the processing node has any vehicle moved out from its data subset, the processing node will call JMS to transfer the vehicle to the destination processing node taking responsibility for the target data subset and wait for the acceptant signal from the destination node. However, if the destination node cannot accept the transferring vehicle the sender should stop the vehicle and try to do a transferring process again in the next step of simulation. Finally, when the requested task has been completed, notification will be sent back to the Manager.

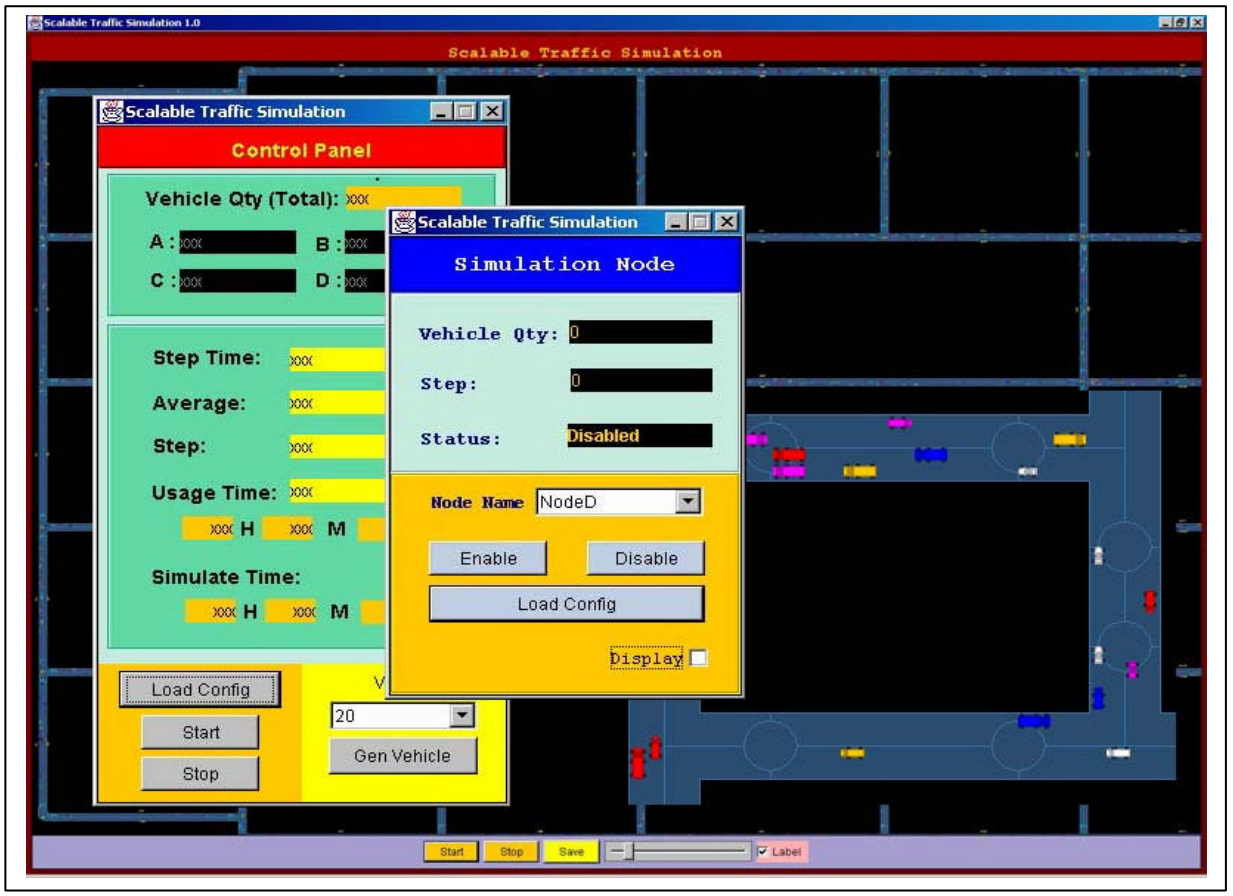

Figure 4: Graphics User Interface of Scalable Traffic Simulation

3.6 Visualization

The user interface in this implementation is used features of Java 2D API in order to produce high quality two dimensions images. The traffic network visualization is presented in two ways; one is

display overall traffic status at the manager node and another displays the network traffic at processing nodes, which is only a subset of traffic network components is displayed. Both of them are able to display in level of individual vehicle movements with zoom in and zoom out ability as showed in the figure 4.

4. Performance Tests

The performance tests demonstrate system ability to scale in performance on parallel machines. For this test a network with 592 links and 608 nodes, and modeling a 2-lane network components, in area of 2.4 square kilometers. Traffic was simulated using 1, 2, and 4 processing nodes of the Pentium III 600 with 128Mbytes memory on Windows 2000 Advanced Server. The term speedup means the ratio of simulation usage time running on multi-processing nodes compared to running on a single processing node.

The following table shows the performance index and the relative speedup for each of these simulation runs.

Table 1 shows that the relation between usage time and the number of vehicles are not linear characteristic because the usage time is depends on both processing power and the traffic situation such a case of congestions. This implies that when we increase the number of vehicle to be two times larger, the simulation situations will be changed not just only a bigger workload that each node has to processes but also a higher congestion will be another factor.

4 Processing nodes

Number of Vehicle	Usage Time	Simulation Step	Average Step Time
5,000	14.851 minutes	7,956	112 ms
10,000	56.36 minutes	12,145	167 ms
20,000	97.34 minutes	18,027	324 ms

Table 1: Performance measuring at fixed number of node

Table 2 and Figure 5 shows the relation between number of processing node and simulation usage time. We increase the number of processing nodes to be stepped of two times from 1, 2 and 4. Ideally, speed up should be increased around twice for each step but as we have founded from our test result, the speed up is increased around 1.4~1.6 times, which is around 75% of the ideal. This result illustrates that the key of speed up ratio is not depend on only a processing power of each nodes but it has some effect from communication time that each node paid for synchronization, controlling and vehicle interchange. The major factor of communication is the vehicle interchange time because it consumes far more higher network bandwidth compares to synchronization or controlling message, therefore with a minimize of the vehicle interchange time should bring on a maximize of speed up.

Vehicle Quantity = 20,000

Number of Node	Usage Time	Simulation Step	Average Step Time	Speed up
1	207.49 minutes	17,785	700 ms	1.00
2	128.05 minutes	17,952	428 ms	1.62
4	83.34 minutes	18,027	278 ms	2.49

Table 2: Performance measuring at fixed quantity of vehicle

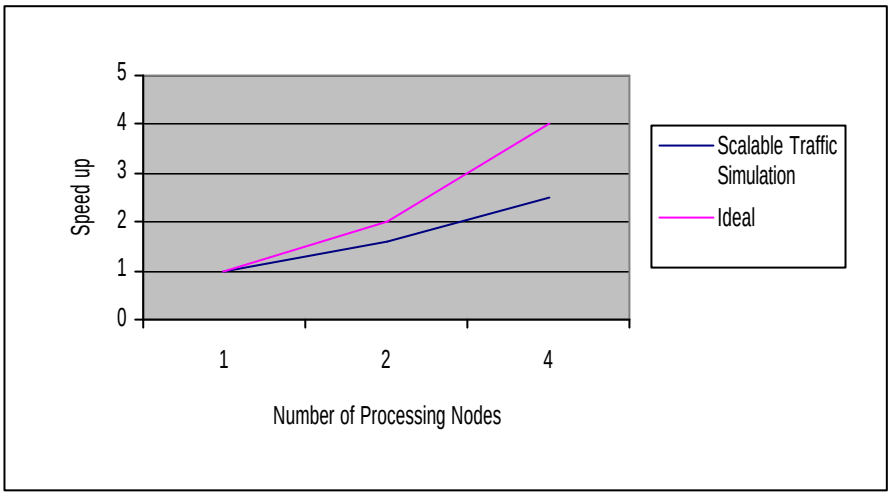

Figure 5: Speed up ratio

The vehicle interchange time is very sensible from the partitioning of traffic map subset. To minimize the vehicle interchange time the area around the subset's boundary should have a lower rate from the vehicle moving across the bound.

5. Conclusions

This paper presents the development of scalable traffic simulation by using Java technology. This development concludes that we can achieve a system with high simulation performance and scalability by decomposing the traffic area to be subsets and distribute it to process concurrently at the processing nodes. The simulation performance is depending on two major factors, processing power and the communication time of the computer nodes.

References

[1] "PARAMICS - Moving Vehicles on the Connection Machine", Proceedings of Supercomputing '94, Washington D.C., November 1994.

[2] Andrej Tibaut, "Parallel Traffic Simulation with an Algorithm for Adaptive Load-Balancing", http://www.uni-weimar.de/~ikm/PROC97/DOCS/061/INDEX.HTM.

[3] Ryota Horigushi, Masao Kuwahara, “A NETWORK SIMULATION MODEL FOR IMPACT STUDIES OF TRAFFIC MANAGEMENT 'AVENUE Ver.2'”, Third Annual World Congress on Intelligent Transport Systems, Orlando, Florida, Oct. 1996.
[4] Ryota Horigushi, Toshio Yoshii, “A BENCHMARK DATA SET FOR VALIDITY EVALUATION OF ROAD NETWORK SIMULATION MODELS”, The 5th World Congress on Intelligent Transport Systems, Seoul, Oct. 1998