To be presented at the 2nd World Congress on Intelligent Transport Systems,
November 9-11, 1995, Yokohama, Japan

Traffic Simulations on Parallel Computers Using Domain Decomposition Techniques*

by

U.R. Hanebutte and A.M. Tentner

Reactor Analysis Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439 USA
(708) 252-1866

*Work supported by the U.S. Department of Energy, Nuclear Energy Programs under Contract W-31-109-ENG-38

Traffic Simulations on Parallel Computers Using Domain Decomposition Techniques

U.R. Hanebutte and A.M. Tentner
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439
USA

ABSTRACT

Large scale simulations of Intelligent Transportation Systems (ITS) can only be achieved by using the computing resources offered by parallel computing architectures. Domain decomposition techniques are proposed which allow the performance of traffic simulations with the standard simulation package TRAF-NETSIM on a 128 nodes IBM SPx parallel supercomputer as well as on a cluster of SUN workstations. Whilst this particular parallel implementation is based on NETSIM, a microscopic traffic simulation model, the presented strategy is applicable to a broad class of traffic simulations. An outer iteration loop must be introduced in order to converge to a global solution. A performance study that utilizes a scalable test network that consist of square-grids is presented, which addresses the performance penalty introduced by the additional iteration loop.

1. INTRODUCTION

The design process of any advanced Intelligent Transportation System (ITS) requires traffic modeling of a very high level of fidelity that is capable of simulating large geographic regions. Such large scale simulations can only be achieved with the large computing power offered by parallel computing architectures. It should be noted that there are two fundamentally different approaches in the utilization of parallel computers. One may write new simulation software that incorporates new algorithmic models geared for parallel processing [1]-[4] or one may study techniques that allow existing proven simulation models [5], [6] to take advantage of parallel computing. The present study is concerned with the later. Domain decomposition techniques are proposed to perform traffic simulations with the standard simulation package TRAF-NETSIM in a parallel environment.

2. DOMAIN DECOMPOSITION

Conceptually, domain decomposition as shown in Fig. 1 is straightforward, however care has to be taken in its implementation. Rather then being simply a "divide and conquer" algorithm, the essence of the domain decomposition method lays in the integration procedure required to obtain a global solution. Further, the issues of inter-regional boundaries needs to be addressed. As Keyes points out in Ref. [7]: "An obvious hurdle to this approach is the specification of boundary conditions on the artificially introduced interfaces between subdomains, upon possession of which the subdomains would trivially decouple."

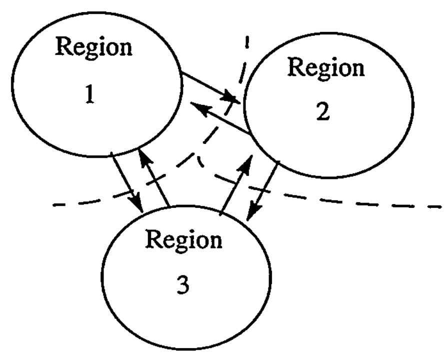

Figure 1 Partitioned traffic network domain.

Even though this particular parallel implementation is based on TRAF-NETSIM the strategy is applicable to a broad class of traffic simulations. The presented technique does not require any change in the existing flow models. The method relies solely on the ability to record outflow traffic volumes at road segments leaving each individual geographic region, and the capability to describe entry traffic volumes at boundary nodes (see Fig. 2). These two features are well supported by any traffic simulation model, and are used here to express connectivity relations between individual domains. Taking the exit volumes of abutting domains as source volumes each domain can be solved in isolation, exploiting a natural source of parallel processing. An outer iteration loop must be introduced in order to converge to a global solution.

In its current form, the algorithm does not provide for feedback in the upstream direction, i.e. the traffic flow condition in an entry link does not affect the vehicle movement in the associated exit link. However, such feedback can be incorporated in the algorithm by making the speed limit of the exit link dependent on the flow condition of the abutting entry link.

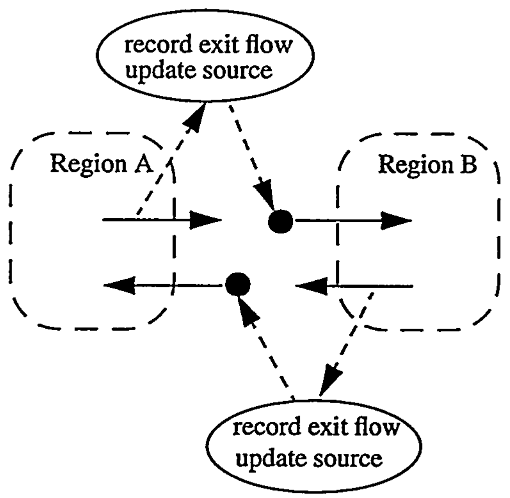

Figure 2 Concept of domain connectivity.

3. PARALLEL IMPLEMENTATION

3.1 The IBM SPx

The IBM SPx scalable parallel computer at Argonne National Laboratory [8] consists of 128 processing nodes. Each node is essentially an IBM RS/6000 model 370 workstation with a clock rate of 62.5 MHz. The local memory is 128 Mb, and the processor data and instruction cache is 32 Kb each. In addition, each processing node has a 1 Gb disk attached for local file I/O. The individual nodes are connected by a multistage network that consists of high-performance switches (63 μ sec latency, 35 Mb bandwidth). The peak performance obtained by performing one multiplication and one addition on 64-bit floating point numbers per clock cycle is 125 Mflops for each processing node. However, in practice, a FORTRAN code delivers 15-75 Mflops.

3.2 Parallel Programming System

To insure portability of the parallel code across multiple platforms, a high-level message passing library is essential. While the emerging Massage Passing Interface (MPI) standard [9] will be utilized in future code development, the P4 parallel programming system [10] was selected for the current implementation of the parallel NETSIM code on both, the SUN workstation cluster and the IBM SPx.

3.3 TRAF-NETSIM

TRAF-NETSIM is a large microscopic simulation code developed by the U.S. Department of Transportation (DOT), Federal Highway Administration (FHWA). Originally developed under the "Urban Traffic Control System" (UTCS-1) program in the early 1970's, the latest version (Version 5) has been released in March of 1995 [6]. NETSIM performs a microscopic, stochastic simulation of individual vehicles in an urban roadway system which is traffic controlled [5]. NETSIM in its current form combines a traffic assignment algorithm with the microscopic simulation. The traffic assignment module (a fairly new addition to the code package) is in essence a macroscopic network flow solver [11], which converts traffic demand information given in the form of origin/destination tables into source volumes (vehicles entering the road network) and turning percentages at each intersection. The traffic assignment problem represents a large research area by itself. The issue of solving very large networks using supercomputer and parallel processing is well addressed within the research community, see i.e. Ref. [12] and [13]. Therefore, the present study concentrates itself solely on the microscopic simulation part of NETSIM.

The source code available for this study (Version 4.21, dated January 1994) was designed to be used on a personal computer (PC) with the DOS operating system. The envisioned mode of operation is an interactive one utilizing a PC with graphic user interface support. With over 75 thousand lines of FORTRAN code, the NETSIM code would be extremely hard to modify for fine grain parallel execution. Extensive code modifications and restructuring would also require an extensive validation and verification effort in order to prove that these modifications do not affect the results of the simulation code. This makes the NETSIM code an ideal candidate for the domain decomposition approach proposed in this paper, which requires only minor modifications to the original code.

The NETSIM code takes advantage of bit-packing (i.e. multiple variables are stored in one storage location) to reduce memory requirements. However, this introduces limits on the maximum network size that can be simulated. Mahmassani et al. [14] gives the limit of 10,000 vehicles, 1650 links and 700 nodes as a problem size of NETSIM, which does not require extensive code modifications. We have adopted these values in the present study. The concept of domain decomposition enables one to perform simulations of much larger traffic networks, since above mentioned limitations do apply to the subdomains only.

3.4 Parallel Framework

The original NETSIM code has to be augmented with a set of routines to control the parallel execution, connectivity of the subdomains, and the global solution algorithm. The parallel framework is written as a SPMD (single program multiple data) program. It provides a framework to execute a copy of the original NETSIM code on each processing node. Advantage is further taken of the local file storage provided by the SPx.

For each outer iteration the parallel framework generates on each processing node domain specific input information for a complete NETSIM simulation. This information is stored as a standard NETSIM input file (using the source volume and turning percentage input format) on local disks. After a NETSIM simulation is completed, the calculated link flows (vehicles per hour) are recorded and exit link flow data is exchanged between abutting domains. This step requires that all processing nodes have completed their assigned NETSIM simulation, which is insured by setting a synchronization point prior to the data exchange. Having the exit flow information from neighboring domains locally available, the source volumes are updated and an input file for the next outer iteration is generated on each processing node simultaneously. Provisions are made to allow for under/over relaxation and "smoothing" operations to take place between iteration steps.

To optimize the input generation process, each node reads from the global file system only once the NETSIM input description containing initial source volumes. Before terminating each processing node writes the link flow information for each link and each outer iteration step to a global file for post-processing.

4. PERFORMANCE STUDY

To investigate the properties of the parallel NETSIM implementation on the IBM SPx and SUN workstations a scalable test network is proposed. A similar regular square traffic network consisting of N^2 intersections has previously been studied by Mahmassani et al. [14]. Our study is unique because each link between two intersections is further divided into two links, thus enabling one to split the traffic network. A test network with 4^2 intersections is depicted in Fig. 3. Input for the scalable network is fully automated, a necessity if one considers the size of the input files. In addition to the NETSIM input file, connectivity information is recorded in a separate file. A 10x10 intersection problem input file contains 2370 lines. For the scaled problem of 10x10 intersections per region on 100 processors the connectivity file has 3600 entries.

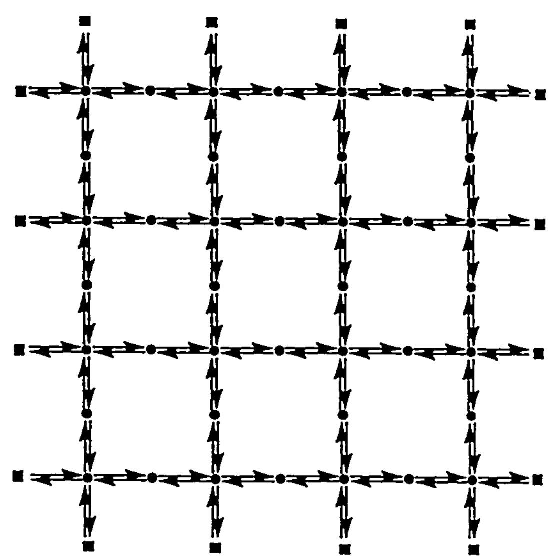

Figure 3 Scalable test network containing 16 intersections (56 nodes, 128 links).

The following parameters have been chosen for the performance study on a single processor. Each link is set to 500 feet (i.e. 1000 feet between intersections), turning percentages are 75%, 15% and 10% for thru, right and left turning traffic respectively; intersection approach control is set to green at all four lanes. It should be noted that a realistic 4-way stop intersection could not be modeled with the version of NETSIM available to this study. Other intersection approach control strategies, e.g. traffic lights, can also be modeled by the test network. The perimeter nodes of the network are connected to source nodes with traffic volume of 500 vehicles/hour.

The NETSIM code has been instrumented with timers to record wall clock time. A 8^2 intersection simulation (10 minute simulation period) requires 128 sec. and 195 sec. on an SPx node and a SUN S-20 workstation respectively. The average network congestion can be calculated to be 55 vehicles/mile, if based on the specified source strength (42 vehicles/mile if based on completed vehicle trips). For a similar traffic network (10 minutes simulation period) an execution time of 125 sec. and 137 sec. on a Cray X-MP is given in Ref. [14] for congestion levels of 50 and 60 vehicles/mile respectively. Thus, a single node of the SPx performed the NETSIM simulation of such size in roughly the same amount of time as a Cray X-MP.

NETSIM performance data for the scalable network containing 1 to 14^2 intersections has been collected for simulated time periods of 10 and 60 minutes. Based on theses data points an execution time model has been established. The execution time, t, on a single processor as a function of network size and simulated time period can be estimated by

$$t = a \cdot N^2 + b \cdot N + c + d \cdot NT, \qquad (1)$$

where N denotes the number of intersections in the network and T is the simulated time period in minutes. The coefficients a through d are given in Table 1 for the IBM SPx and a SUN Sparc Station 20.

Table 1
Machine specific coefficient for the execution time model of the scalable test network.

Machine	*a*	*b*	*c*	*d*
IBM SPx	0.003	0.533	1.52	0.115
SUN S-20	0.015	0.517	1.69	0.138

Fig. 4 compares the experimentally obtained performance numbers with the execution model given in Eq.(1). The model correlates well with the actual measurements

Since the aim of the parallel NETSIM implementation is the simulation of very large networks, a scaled problem size performance study is presented, where each subdomain is a road network containing 10^2 intersections. A perfect load balance among processors is guaranteed since this problem has uniform subdomains with similar traffic congestion level. In Table 2 the number of intersections, nodes, and links contained in the network is given as a function of the number of processing nodes.

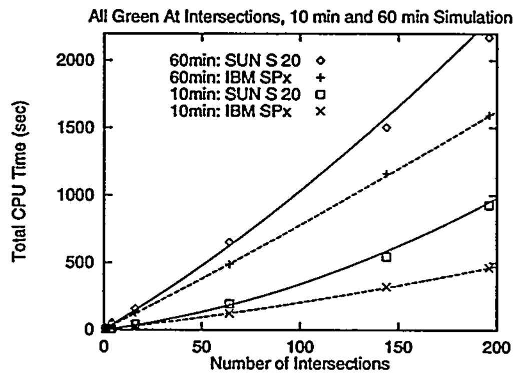

Figure 4 Execution time of the scalable test network on a single processor.

Table 2
Total number of intersections, nodes and links of the test network (10^2 intersections per subdomain), for various processor numbers P.

P	Inters.	Nodes	Links
1	100	320	800
4	400	1,280	3,200
9	900	2,880	7,200
16	1,600	5,120	12,800
25	2,500	8,000	20,000
36	3,600	11,520	28,800
49	4,900	15,680	39,200
64	6,400	20,480	51,200
81	8,100	25,920	64,800
100	10,000	32,000	80,000

For a 10^2 intersection network (10 minutes simulation period) the predicted time, *t(N)*, obtained by the execution time model is 200 sec., while the 100 times larger network has a predicted single processor execution time of 88 hours.

If one assumes that the execution time model is also valid for very large network sizes which are beyond the limit of the current NETSIM code, a performance gain due to domain splitting can be predicted. The predicted gain is given by

$$gain_{theoretical} = \frac{t(N)}{P \cdot t\left(\frac{N}{P}\right)}, \qquad (2)$$

where $\frac{N}{P}$ represents the number of intersections contained in each subdomain and P denotes the number of subdomains.

For the test problem the predicted gain is included in Table 3 for the SUN S-20 and the IBM SPx. This is merely a theoretical value, since it neglects the fact that an additional outer iteration loop is required to reach the global solution. Further performance penalties due to communication and synchronization costs are encountered when subdomains are distributed among separate processors. However, the theoretical gain should compensate for most or in some cases all of the additional costs induced by the global solution algorithm and the parallel overhead. Thus, speedups close to linear, i.e. computing a problem on P processors requires only 1/P of the time which is required on a single processor, may be seen for the parallel execution.

Two execution times (wall clock time) are further given in Table 3 for one outer iteration of the NETSIM code. The time T_{traf} denotes the time required by the TRAF-NETSIM code, while T_{total} includes all additional parallel overhead, i.e. synchronization, communication and input generation step. One notices, that up to a large number of processors the total execution time is very close to the time the NETSIM code requires by itself. One reason for the performance degradation on very large processing clusters is the fact that the original NETSIM code writes intermediate print statements to the standard output, which the current implementation with the P4 library collects at one processor. Even though there are only a few of these statements per processor, the actual handling causes some delay in the program execution.

Table 3

Performance of the test network given in Table 2 (10 minutes simulation period) on the IBM SPx for one outer iteration and predicted gain on the IBM SPx and SUN S-20.

P	T_{traf} (sec)	T_{total} (sec)	Gain SPx	Gain S-20
1	220	220	1	1
4	218	219	1.4	2.3
9	219	220	2.2	4.5
16	221	236	3.2	7.6
25	220	272	4.6	11.5
36	273	274	6.3	16.7
49	275	276	8.2	20.0
64	226	248	10.5	28.6
81	227	286	13.0	36.0
100	280	332	15.9	50.0

However, the number of iterations is dependent on the type of traffic pattern on the network. A minimum of 3 outer iterations is required on such 2x2 subdomain problem to "transfer information" from one corner of the domain to the opposite one, at each iteration only neighboring domains exchange information. The large network on 100 processors requires 14 outer iterations (the length of the diagonal of a domain containing 10x10 regions). One could reason, that 2 to 3 sweeps along the diagonal of the domain would be sufficient to reach global convergence. If this conjecture holds, 28 to 42 outer iterations would be needed for the large network on 100 processors. Convergence could be reached faster by setting the source volumes on interregional boundaries for the first iteration step to values obtained by a macroscopic traffic simulation.

5. CONCLUSION

A parallel framework for performing microscopic traffic simulations has been established and TRAF-NETSIM simulations have been carried out on up to 100 processors of the IBM SPx at Argonne National Laboratory. The developed strategy is applicable to a broad class of traffic simulation codes. While all software issues have been resolved and each individual iteration step demonstrates good performance, the global convergence of the algorithm needs further study.

The global solution procedure is subject of current investigations. Under/over relaxation and "smoothing" operations between iteration steps and their effect on global convergence are also being studied. These test will be carried out on problems small enough to allow comparison with simulations performed as one domain on a single processor. Further, one may study acceleration strategies which are based on low-order methods, such as macroscopic traffic models.

Also, load balance and "flow-sensitive" partitioning need to be addressed in the future. The goal of flow-sensitive partitioning is to constrain subdomain boundaries to come only between weakly coupled points while preserving load balance.

6. REFERENCES

[1] Ewing, T.F., Canfield, T., Hanebutte, U.R., Levine, D. and Tentner, A., "Large-Scale Intelligent Transportation Systems Simulation," *Proc. High Performance Computing '95* (part of the1995 International Simulation Conference), Phoenix, Arizona, April 9-13, 1995, p. 253-257.

[2] Nagel, K. and Schleicher, A., "Microscopic traffic modeling on parallel high performance computers," *Parallel Computing 20, p. 125-146,* 1994.

[3] Mc Arthur, D., Cameron, G. and Smith, M. "Parallel Microscopic Simulation of Traffic on the Scottish Trunk Road Network,"*Proc. ISATA,* October 1994.

[4] Kerridge, J. et al., "Modelling Vehicle Behaviour in Congested Networks Using Highly Parallel Systems," *Proc. of the 74rd Annual Meeting of Transportation Research Board,* Washington, DC, January 1995.

[5] Mekemson, J.R., Herlihy, E.T. and Wong, S-Y, "Traffic Models Overview Handbook," *Federal Highway Administration Report No. FHWA-SA-93-050,* June 1993.

[6] Yedlin, M. et al., "The New TRAF-NETSIM: Version 5.0," *Proc. of the 73rd Annual Meeting of Transportation Research Board,* Washington, DC, January 1994.

[7] Keyes, D.,"Domain Decomposition: A Bridge Between Nature and Parallel Computers," *ICASE/ LaRC Short Course on Parallel Computation,* Hampton, VA, July 26-30 1993.

[8] Gropp, W., Lusk, E. and Pieper, S.C., "Users Guide for the ANL IBM SP1," *Argonne National Laboratory Report, ANL/MCS-TM-198,* October 1994.

[9] Foster, I., *Designing and Building Parallel Programs,* Addison-Wesley Inc., (also, world wide web URL: http://www.mcs.anl.gov/dbpp), 1994.

[10] Butler, R. and Lusk, E., "Monitors, message, and clusters: The p4 parallel programming system," *Parallel Computing, 20, p.547-564,* 1994.

[11] Sheffi, Y., *Urban transportation networks: Equilibrium analysis with mathematical programming methods,* Englewood Cliffs, N.J., Prentice-Hall, 1984.

[12] Nagurney, A., *Network Economics, A Variational Inequality Approach,* Kluwer Academic Publishers, Norwell, Mass, 1993.

[13] Boyce, D.E., Tarko, A., Berka, S. and Zhang, Y.,"Estimation of link travel times with a large network flow model for a dynamic route guidance system",*Proc. of IVHS America,* Atlanta, Georgia, April 17-20, 1994, p. 37-43.

[14] Mahmassani, H.S., Jayakrishnan, R. and Herman, R.,"Microscopic Simulation of Traffic in Networks: Supercomputer Experience," *Journal of Computing in Civil Engineering Vol 4, No 1, p. 1-19,* January 1990.