Distributed/Parallel Genetic Algorithm for Road Traffic Network Division for Distributed Traffic Simulation

Tomas Potuzak

Abstract. In this paper, a distributed/parallel method for division of road traffic networks for distributed road traffic simulation is described. The method is based on its sequential version, which we developed during our previous research. This sequential version utilizes the weights of traffic lanes representing the numbers of vehicles moving within them and a genetic algorithm for the division of the road traffic network into the required number of load-balanced sub-networks interconnected with minimal number of divided traffic lanes. The distributed/parallel version of the division method described in this paper uses a similar approach, but utilizes a distributed/parallel computing environment for a distributed/parallel execution of the genetic algorithm and, consequently, for the speedup of the entire method.

1 Introduction

A utilization of a distributed computing environment where combined power of multiple interconnected computers is utilized simultaneously is a commonly used approach for the speedup of a detailed road traffic simulation [1]. An example of a distributed computer can be a cluster of ordinary interconnected desktop computers (e.g. in a classroom of a university). Today, these computers often incorporate multicore processors, which enable to perform several processes or threads concurrently and, consequently, to achieve additional speedup of the simulation [2].

For the distributed simulation of a road traffic network, it is necessary to divide this network into required number of sub-networks, which are then simulated on particular nodes of the distributed computer as (possibly multithreaded) processes. A convenient division is an important factor influencing the resulting performance

Tomas Potuzak
University of West Bohemia, Faculty of Applied Sciences,
Department of Computer Science and Engineering,
Univerzitni 8, 306 14 Plzen, Czech Republic
e-mail: tpotuzak@kiv.zcu.cz

F. Zavoral et al. (eds.), *Intelligent Distributed Computing VII,*
Studies in Computational Intelligence 511,
DOI: 10.1007/978-3-319-01571-2_18,

of the entire distributed simulation [3]. During our previous research, we developed a method for road traffic network division, in which a genetic algorithm is employed [4]. The method is sequential and shows good results. However, it is quite slow on a standard desktop computer, especially for large road traffic networks [4].

In this paper, we describe the adaptation of our sequential division method for a distributed/parallel environment in order to maximally utilize the computing power of the environment for a faster division of road traffic networks.

2 Basic Notions

The distributed/parallel method for division of road traffic networks is intended for distributed/parallel road traffic simulation. Such a simulation can be performed as a set of singlethreaded processes on a distributed computer (i.e. without shared memory), a multithreaded process on a parallel computer (i.e. with shared memory), or as a set of multithreaded processes on a distributed/parallel computer (i.e. shared memory among threads of one process, but not among processes) [2].

2.1 Distributed/Parallel Road Traffic Simulation Description

The road traffic simulation is most often classified using its level of detail as *macroscopic*, *mesoscopic*, or *microscopic*. The macroscopic simulation deals only with aggregate traffic flows in particular traffic lanes [5]. In the mesoscopic simulation, there is some representation of vehicles (e.g. tasks in queuing networks), but modeling of their interactions is limited [6]. The microscopic simulation considers all vehicles as objects with their own parameters and interactions [7, 8], which makes it very time-consuming and convenient for distributed computing environment [3].

In this case, it is necessary to divide the road traffic network into required number of sub-networks, which are then simulated on particular nodes of the distributed computer as processes. The processes utilize a communication protocol based on message passing for the transfer of vehicles between the sub-networks and also for the synchronization. If the simulation processes are multithreaded, each thread computes part of the road traffic sub-network, which resides in shared memory of the threads. The threads in one simulation process must be synchronized as well [2].

2.2 Road Traffic Network Division Description

The division of the road traffic network is necessary only among the simulation processes (one sub-network per process), not among the simulation threads. The simulation threads of one process can access the sub-network in its shared memory and each of them can simulate a part of crossroads, traffic lanes, and so on [2].

Two issues should be considered during the division of road traffic network for the performance reasons – the load-balancing of the resulting sub-networks and the minimization of the inter-process communication. The load-balancing is necessary

in order to achieve similar speeds of the simulation processes and therefore to minimize their mutual waiting [9]. The minimization of the inter-process communication is necessary, since the communication is very slow [9]. The communication can be reduced by minimization of the number of divided traffic lanes, which leads to lower number of transferred vehicles (i.e. lower number of transferred messages, depending on the communication protocol) [3].

2.3 Genetic Algorithms Description

A genetic algorithm is an evolutionary algorithm [10] mimicking the natural genetic evolution and selection in nature in order to solve a problem [11]. Today, the genetic algorithms are widely used for solving of searching and (multi-objective) optimization problems in many domains [12].

A general genetic algorithm works as follows. A representation of an *individual* (usually a vector of binary or integer values) is selected based on the solved problem [13] and a set of individuals – *initial population* – is most often randomly generated. Then, a *fitness value*, representing an objective assessment in relation to the solved problem, is calculated using a *fitness function* for each individual of this initial population [13]. The fitness function can be single- or multi-objective [12]. Once this is completed, a number of individuals with highest fitness values are selected to be "parents" of a new generation. The new generation is then created from these individuals using *crossover* and *mutation* operators. The crossover uses two individuals to produce new individuals (descendants) incorporating information from both parents, which can be then mutated (i.e. partially randomly changed) [10].

The created descendants form a new generation, whose size corresponds to the number of individuals of the initial population. Then, the fitness value is calculated for the individuals of this new generation and the entire process repeats until a preset number of generations is created or a stop condition is fulfilled [14].

3 Road Traffic Network Division Method

The sequential method for road traffic network division, which we developed, is described in [4] and [14] in detail. The method uses a less-detailed (macroscopic) simulation for assigning of the weights to particular traffic lanes representing the numbers of vehicles in these lanes during the simulation. The traffic network is then considered as a weighted graph with crossroad acting as nodes and sets of lanes connecting the neighboring crossroads acting as edges with weights corresponding to the sum of weights of the particular lanes [4].

3.1 Sequential Dividing Genetic Algorithm Description

The weighted graph is the input for the dividing genetic algorithm (DGA) together with the number of sub-networks, which shall be created. The DGA individual is

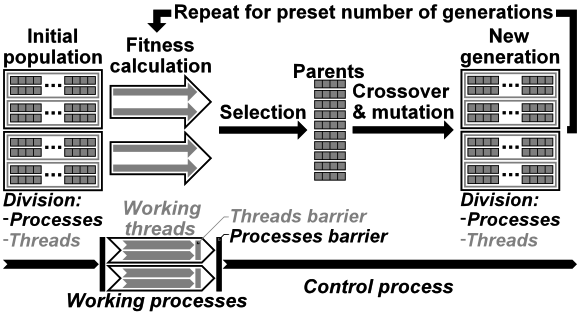

Fig. 1 Schema of the D/P-DGA for two processes and two threads per process. The control process performs the division of individuals among working processes, selection, crossover, and mutation. The working processes perform fitness calculation using multiple threads. Barriers are used for synchronization of both threads and processes.

then a vector of integers of length corresponding to the number of crossroads. Each value represents assignment of one crossroad to one sub-network [14].

The initial population consists of 90 randomly generated individuals, for which the fitness values are calculated. The fitness function prefers individuals representing assignment of the crossroads corresponding to load-balanced sub-networks with minimal number of divided traffic lanes between them. So, it consists of two parts – the *equability* representing the load-balancing and the *compactness* representing the minimal number of divided traffic lanes. For more details, see [4] and [14].

Once all fitness values are calculated, ten individuals with highest values are selected for the creation of a new generation using crossover. Each combination of the selected individuals is used to produce two descendants. First descendant receives all integers of even index of the first parent and of odd index of the second parent. Second descendant receives the remainder. Both descendants can be mutated (i.e. random change of several integers). This way a new generation of 90 individuals is created and the entire process repeats for a preset number of generations [4].

3.2 Distributed/Parallel Dividing Genetic Algorithm Description

In order to speed up the DGA, we developed its distributed/parallel version (D/P-DGA), which is the main contribution of this paper. The D/P-DGA is operational, but is still under development. So far, only the fitness calculation is parallelized, since it is the most time-consuming part of the DGA. Moreover, its parallelization is straightforward, because it is an independent computation for each individual.

The implementation of the D/P-DGA is a part of the DUTS Editor, a system for design and division of road traffic networks developed at Department of Computer Science and Engineering of University of West Bohemia (DSCE UWB). The D/P-DGA computation generally consists of multiple multithreaded working processes and one control process, which communicate using message passing. The schema of the D/P-DGA for two processes with two threads per process is depicted in Fig. 1.

4 Tests and Results

The D/P-DGA was thoroughly tested using three computers Intel i7 3.07 GH (4 processor cores with hyper-threading), 12 GB RAM, 1 TB HDD, and Windows 7 interconnected by 1 Gb Ethernet. Three road traffic networks were divided into 4 sub-networks. All three were regular square grids of 64, 256, and 1024 crossroads. The sequential (1 process, 1 thread on 1 computer), the parallel (1 process, 2-4 threads on 1 computer), the distributed (1 control and 2 working processes, 1 thread per working process on 3 computers), and the distributed/parallel (1 control process, 2 working processes, 2-4 threads per working process on 3 computers) executions were tested. The results (averaged from ten attempts) are depicted in Table 1.

Table 1 Results of sequential, parallel, distributed, and distributed/parallel executions

Processes / threads count	1000 generations			10000 generations			100000 generations		
	64	256	1024	64	256	1024	64	256	1024
	Computation time [ms]								
1 / 1[a]	618	2443	12689	5601	22826	113632	55385	221702	1105034
1 / 2[b]	340	1301	6653	3149	12003	58195	30635	119415	576759
1 / 4[b]	219	741	3585	1945	6726	31787	19076	66767	311794
2 / 1[c]	713	1781	6669	7061	17167	63267	69934	169770	627977
2 / 2[d]	608	1253	3968	5842	12002	38812	58227	119026	379333
2 / 4[d]	539	942	2330	5147	8987	22191	49910	87985	207598

[a] Sequential execution [b] Parallel execution [c] Distributed execution [d] Distributed/parallel execution.

The computation time decreases with increasing number of threads. Nevertheless, the results for the distributed/parallel execution are far worse than for the parallel execution, because the individuals and calculated fitness values are transferred between the control and the working processes using the message passing. This significantly degrades the overall performance. The results are better for the largest road traffic network, because there is better computation-to-communication ratio.

5 Conclusion

In this paper, we presented a distributed/parallel dividing genetic algorithm (D/P-DGA) for division of road traffic networks for distributed road traffic simulation. Using the distributed/parallel computation of the fitness values, the total computation time is significantly reduced, which was shown during a thorough testing.

Also, it has been determined that the parallel execution offers better speedup (up to 3.54 using 4 threads) than the distributed/parallel execution (up to 5.32, but using 2 working processes with 4 threads - i.e. 8 threads in total) in comparison

to the sequential execution due to the lack of inter-process communication. Still, the distributed/parallel execution offers good speedup for large road traffic networks and has the advantage of better scalability, since it is possible to add more working processes using larger number of computers.

In our future work, we will focus on better parallelization of the genetic algorithm. So, the selection, crossover, and mutation will be performed in a parallel way similar to the calculation of the fitness values. We will also explore the possibilities of the inter-process communication reduction.

References

1. Nagel, K., Rickert, M.: Parallel Implementation of the TRANSIMS Micro-Simulation. Parallel Computing 27(12), 1611–1639 (2001)
2. Potuzak, T.: Distributed-Parallel Road Traffic Simulator for Clusters of Multi-core Computers. In: 2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications – DS-RT 2012, pp. 195–201 (2002)
3. Potuzak, T.: Methods for Reduction of Interprocess Communication in Distributed Simulation of Road Traffic. Doctoral thesis, University of West Bohemia, Pilsen (2009)
4. Potuzak, T.: Methods for Division of Road Traffic Networks Focused on Load-Balancing. Advances in Computing 2(4), 42–53 (2012)
5. Lighill, M.H., Whitman, G.B.: On kinematic waves II: A theory of traffic flow on long crowed roads. Proceedings of the Royal Society of London, S. A 229, 317–345 (1955)
6. Nizzard, L.: Combining Microscopic and Mesoscopic Traffic Simulators. Raport de stage d'option scientifique Ecole Polytechnique, Paris (2002)
7. Gipps, P.G.: A behavioural car following model for computer simulation. Transp. Res. Board 15-B(2), 403–414 (1981)
8. Nagel, K., Schreckenberg, M.: A Cellular Automaton Model for Freeway Traffic. Journal de Physique I 2, 2221–2229 (1992)
9. Potuzak, T.: Utilization of a Genetic Algorithm in Division of Road Traffic Network for Distributed Simulation. In: ECBS-EERC 2011 – 2011 Second Eastern European Regional Conference on the Engineering of Computer Based Systems, pp. 151–152 (2011)
10. Poli, R., Langdon, W.B., McPhee, N.F.: A field guide to genetic programming (2008), Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk` (with contributions by J.R. Koza)
11. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
12. Farshbaf, M., Feizi-Darakhshi, M.: Multi-objective Optimization of Graph Partitioning using Genetic Algorithms. In: 2009 Third International Conference on Advanced Engineering Computing and Applications in Sciences, pp. 1–6 (2009)
13. Menouar, B.: Genetic Algorithm Encoding Representations for Graph Partitioning Problems. In: 2010 International Conference on Machine and Web Intelligence (ICMWI), pp. 288–291 (2010)
14. Potuzak, T.: Suitability of a Genetic Algorithm for Road Traffic Network Division. In: KDIR 2011 - Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, pp. 448–451 (2011)