Transportation Research Part C 75 (2017) 1–16

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Large-scale dynamic transportation network simulation: A space-time-event parallel computing approach

Yunchao Qu [a,b], Xuesong Zhou [b,*]

[a] State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China
[b] School of Sustainable Engineering and the Built Environment, Arizona State University, Tempe, AZ 85287, USA

ARTICLE INFO

Article history:
Received 29 June 2016
Received in revised form 7 December 2016
Accepted 8 December 2016

Keywords:
Synchronous parallel strategy
Mesoscopic transportation simulation
Space-time-event network
Parallel discrete event simulation

ABSTRACT

This paper describes a computationally efficient parallel-computing framework for mesoscopic transportation simulation on large-scale networks. By introducing an overall data structure for mesoscopic dynamic transportation simulation, we discuss a set of implementation issues for enabling flexible parallel computing on a multi-core shared memory architecture. First, we embed an event-based simulation logic to implement a simplified kinematic wave model and reduce simulation overhead. Second, we present a space-time-event computing framework to decompose simulation steps to reduce communication overhead in parallel execution and an OpenMP-based space-time-processor implementation method that is used to automate task partition tasks. According to the spatial and temporal attributes, various types of simulation events are mapped to independent logical processes that can concurrently execute their procedures while maintaining good load balance. We propose a synchronous space-parallel simulation strategy to dynamically assign the logical processes to different threads. The proposed method is then applied to simulate large-scale, real-world networks to examine the computational efficiency under different numbers of CPU threads. Numerical experiments demonstrate that the implemented parallel computing algorithm can significantly improve the computational efficiency and it can reach up to a speedup of 10 on a workstation with 32 computing threads.

1. Introduction

Compared to the sequential computing mode utilized in most existing traffic simulation and planning models, parallel computing not only efficiently utilizes widely available distributed computing powers and communication networks, but also redefines what is tractable for time-critical transportation simulation and management strategy optimization. Emerging multi-core computer processor techniques are offering unprecedented available parallel computing resources, through a wide range of high-performance laptops and desktops currently available in the market. This paper aims to develop a parallel algorithm design for transportation simulation to exploit this paradigm change in computing and to facilitate the most efficient use of emergent parallel hardware.

* Corresponding author.
E-mail addresses: quyunchao0613@gmail.com (Y. Qu), xzhou74@asu.edu (X. Zhou).

http://dx.doi.org/10.1016/j.trc.2016.12.003
0968-090X/

1.1. Literature review

At the core of transportation simulation, traffic flow models are interested in the quantitative relationship between flow, density and speed, and modeling the interactions between different agents. In a transportation network, there may be many routes between each origin and destination and agents choose better routes to reduce travel time. Motivated by network-wide traffic management application needs, such as regional traffic mobility analysis and real-time route guidance, dynamic traffic assignment (DTA) models have been increasingly recognized as an important approach for assessing performance of different traffic system management and information provision strategies. There are macroscopic, mesoscopic or microscopic simulation-based methods for generating time-dependent travel time measures in general traffic simulators and DTA models (Mahmassani et al., 1994; Mahmassani, 2001; Ben-Akiva et al., 2002; Peeta and Ziliaskopoulos, 2001; Adler and Blue, 2002; Celikoglu and Dell'Orco, 2007; Chen et al., 2009; Di Gangi et al., 2016; Dell'Orco et al., 2016). In an effort to reach the right balance between representation detail and computational efficiency, this study focuses on how to implement a mesoscopic-based dynamic network loading model, within a parallel computing framework, on medium and large-scale real-world networks.

One key method to achieve computational efficiency while simulating medium and large-scale networks is parallel computing. A parallel simulation implementation is valuable for researchers to quickly examine the interactions of vehicular flow and analyze the complex traffic phenomena on a larger scale. It also helps to improve the computational efficiency of traffic model validation and calibration, as well as on-line traffic state estimation and prediction, e.g., through a simulation-based optimization framework. In early studies, parallel computing techniques have been applied in several transportation simulation systems (Junchaya and Chang, 1993; Wong, 1997; Ziliaskopoulos et al., 1997). PARAMICS (Cameron and Duncan, 1996) was implemented on a Connection Machine CM-2, and its graph partition algorithm divided the set of links into different sequences of queues and each queue contained a certain number of moving vehicles. Based on a shared memory platform, Nagel and Schmidt (2002) and Cetin et al. (2003) studied the parallelization of microscopic transportation simulation based on TRANSIMS (Transportation Analysis and Simulation System) using another sophisticated graph partition algorithm. Potuzak (2012) reported a distributed microscopic discrete time-stepped simulator DUTS (Distributed Urban Traffic Simulator) and performed road traffic simulation on a cluster of computers with multi-core processors. Kallioras et al. (2015) applied a GPU-based accelerated metaheuristics approach to solve the transit stop inspection and maintenance scheduling problem. In the field of traffic assignment, Florian and Gendreau (2001) offered a good review on parallel computing approaches for performing the shortest path algorithms, e.g., through network decomposition and network replication strategies. Liu and Meng (2013) demonstrated a solid effort for accelerating the Monte Carlo simulation method for solving probit based stochastic user equilibrium problems using a distributed computing system. Ng and Nguyen (2015) proposed a spatial partitioning method to implement parallel computing. Morosan and Florian (2015) applied a shared-memory strategy focused on parallel shortest path computation and reported that the speedup could reach up to 20 when solving the traffic assignment problem. Auld et al. (2016) developed an agent-based modeling software development kit POLARIS that contains a parallel discrete event simulation engine.

Other early implementation of parallel transportation simulations are also introduced by Barceló et al. (1996) and Lee and Chandrasekar (2002) and a parallel implementation of AIMSUN reported a speed-up of 3.5 on 8 CPUs using multiple threads. As a macro-particle model, a parallel version of DYNEMO has been implemented since 2001 (Nagel and Rickert, 2001). DYNASMART's research team reported their experiment in implementing functional decomposition (Mahmassani et al., 1994). DynaMIT introduced a parallelization concept of functional decomposition (i.e., task parallelization) (Sundaram et al., 2011). Based on GPU techniques, Zhen et al. (2011) recently proposed a parallel computing framework to speed up the traffic simulation and optimize the traffic signal timing.

A significant amount of attention has been devoted to advancing parallel implementation for traffic simulation models for specific hardware/software architecture. The major efforts are summarized as follows: (1) in a static fashion, partitioning different geographical areas of the studied region to different CPU cores, and (2) for distributed computing, designing sophisticated message passing and efficient synchronization methods to reduce communication overhead among different computing cores. In our research, from a broader perspective of parallel discrete event simulation, we aim to offer a more feasible task decomposition methodology to synchronize inter-correlated space-time simulation events. This space-time-event oriented approach could take advantages of automated coordination programming interfaces (e.g. through OpenMP) between threads, processors, distributed computers, and Graphical Processing Unit (GPU).

An important study by Nie et al. (2008) offered a comprehensive discussion on a unified dynamic network loading/simulation framework in capturing congestion propagation effects. They also clearly indicated that their double-buffer-based network loading framework could be used for further parallel simulation prototype development and system implementation. However, to ensure the actual speedup under specific parallel computing architecture, in-depth research is still critically needed to examine a number of important system implementation issues and address how to select an appropriate space-time resolution and simulation execution sequences for mesoscopic or microscopic simulation.

1.2. Space-time-event view for parallel computation

Many further developments (Fujimoto, 1990, 1993; Ferscha and Tripathi, 1998; Liu, 2009; Fujimoto, 2015) summarize a number of parallel processing algorithms in terms of time-parallel and space-parallel categories. In a space-time view