GPU Implementation of the Stochastic On-Time Arrival Routing Algorithm

Anton Agafonov
Geoinformatics and Information Security Department
Samara National Research University
Samara, Russia
ant.agafonov@gmail.com

Aleksey Maksimov
Geoinformatics and Information Security Department
Samara National Research University
Samara, Russia
aleksei.maksimov.ssau@gmail.com

Alexander Borodinov
Geoinformatics and Information Security Department
Samara National Research University
Samara, Russia
aaborodinov@yandex.ru

***Abstract*—In this paper, we consider the stochastic on-time arrival problem in a transportation network with the following optimization criteria: maximizing the probability of arriving at a destination within a predefined time budget. This problem formulation allows considering not only the mean travel time of the road links but also the travel time variance. Existing approaches to solving this problem show good results in terms of the quality of the found route but have high computational complexity. This fact does not allow using them in practical navigational applications in real time. In this paper, we investigate the parallelization strategy of the stochastic on-time arrival routing algorithm using the CUDA GPU. Experimental studies conducted on the large-scale transportation network of the Samara city, Russia show that the proposed approach can reduce the computation time by an average of 5 times.**

***Keywords*—shortest path, reliable routing, intelligent transportation system, SOTA, CUDA**.

I. Introduction

The routing problem remains one of the most actual problems in transportation systems. Existing navigation systems and routing web-services usually consider transportation networks as directed graphs with deterministic edge weights and do not take into account the stochastic properties of traffic flows w hen s olving t he r outing p roblem. A t t he s ame time, the travel time of road segments depends on many factors including the day of the week, time of day, weather conditions, social events, etc [1]. However, taking into account not only the mean travel time, but also the variance of the travel time, that is, the reliability of the route, makes the task of finding the optimal route computationally difficult.

Unlike the practical applications, research papers study routing algorithms in stochastic and time-dependent transportation networks [2]–[4]. There are several formulations of the reliable path finding p roblem, d epending o n t he e valuation criterion used:

1. The least expected travel time (LET) path models [5], [6], in which the expected travel time of the road segments is considered as the evaluation criteria to compare the possible paths.
2. The α-reliable path models [4], [7], [8] minimize the time interval which is necessary to arrive at the terminal vertex (destination) on-time for a given probability α .
3. The most reliable shortest path models [3], [9] maximize the probability of arriving at the destination within a given time budget (stochastic on-time arrival - SOTA problem).

In this paper, we consider the reliable path finding problem with the following formulation: to determine the optimal navigation strategy that maximizes the probability of arriving at the destination within a predefined time interval (time budget).

In [3], the authors proposed an algorithm for the exact solution of the SOTA problem. One of the steps of the algorithm is the convolution calculation, which is the main computationally difficult task. In general, convolution cannot be calculated analytically, and therefore a discrete approximation scheme is required. In [10], the authors presented a modification of the solution [3] that takes into account current and forecast information about traffic flows in the network. These solutions have a large computation time and cannot be used in practical applications in real time. As a result, several studies were devoted to the task of decreasing the computation complexity of the algorithm or the development of approximation algorithms.

In the article [11], the authors presented several methods for accelerating the algorithm for solving the SOTA problem, including advanced convolution calculation algorithms based on the Fast Fourier Transform and zero-delay convolution calculation algorithms, as well as methods for determining the optimal order for calculating the navigation strategy. The paper [12] described a heuristic for finding an adaptive route in a stochastic network. The presented method makes it possible to provide the most computationally effective strategy for finding the path for general probability distributions. In [13], stochastic versions of two graph preprocessing methods were considered to solve the deterministic shortest path problem that can be adapted to the SOTA problem. A parallelization strategy for the SOTA problem using the GPU was proposed in [14]. In [15], the authors proposed to use the stable Levy distributions to describe the travel time of road segments, which allows replacing the complex convolution calculation operation to

determine the reliability of the path with recalculating the Levy distribution parameters. This approach allowed to significantly reduce the execution time of the algorithm, however, it finds a path with an increased travel time.

In this paper, we propose a strategy for parallelizing the reliable path finding algorithm using the CUDA GPU . The work is organized as follows. In the second section, the main notation, problem statement, and description of the algorithm are given. The parallelization strategy is presented in the third section. The fourth section describes the experimental setup and results of experimental studies. Finally, we give the conclusion and possible directions for further research.

II. Problem statement

The transportation network is considered as a directed graph $G = (N, A, P)$, where N is the set of nodes, $|N|$ is the number of nodes, A is the set of edges, $|A|$ is the number of edges, P is the probabilistic description of the edge weights (i.e. the road link travel times.

The weight of the edge $(i, j) \in A$ is considered as a random variable $T_{ij}(\tau)$ with a time-dependent probability density function $p^{\tau}_{ij}(t)$.

Denote the destination node as $d \in N$, time interval within which is necessary to reach the destination node (time budget) denote as T. An optimal routing strategy is defined as a policy that maximizes the probability of arriving at the destination node $d \in N$ from the origin node $o \in N$ within the given time budget T.

Let $u_i(t)$ be the probability of arriving at the destination node d from the node i in time less than t. Then, the optimal routing strategy can be formulated as follows:

$$u_i^{\tau}(t) = \max_{j \in N \wedge (i,j) \in A} \int_0^t p_{ij}^{\tau}(\theta) u_j^{\tau+\theta}(t-\theta) d\theta, \quad \forall i \in N \setminus \{d\}, t \in [0, T], \tau \geq 0, \qquad u_d^{\tau}(t) = 1, t \in [0, T], \tau \geq 0. \tag{1}$$

To solve the problem (1), a discrete algorithm was proposed in [3], which can be formulated in pseudo-code as follows (Algorithm 1).

In Algorithm 1, we use the following notations: Δt is the discretization interval, δ is the minimum realizable link travel time across the entire network.

The selection of the next vertex j in the transportation graph (and, accordingly, the next road link) using the remaining time budget t and the calculated array of arrival probabilities $u_i(x)$ is performed as follows:

$$j = \arg\max_{i \in N} u_i(t). \tag{2}$$

In the next section, we describe an algorithm parallelization strategy using the GPU.

III. Methodology

To decrease the running time of the reliable shortest path search algorithm, it is proposed to implement it on a graphics accelerator using CUDA.

Algorithm 1: Discrete SOTA algorithm

Step 0. Initialization
$k = 0$
$u_i^k(x) = 0, \quad \forall i \in N,\ i \neq d,\ x \in \mathbb{N},\ 0 \leq x \leq \frac{T}{\Delta t}$
$u_d^k(x) = 1, \quad x \in \mathbb{N},\ 0 \leq x \leq \frac{T}{\Delta t}$

Step 1. Update
for $k = 1, 2, \cdots, L$ **do**
$\tau^k = k\delta$
$u_d^k(x) = 1, \quad x \in \mathbb{N}, 0 \leq x \leq \frac{T}{\Delta t}$
$u_i^k(x) = u_i^{k-1}(x)$
$\forall i \in N, i \neq d, (i,j) \in A, x \in \mathbb{N}, 0 \leq x \leq \frac{\tau^k - \delta}{\Delta t}$
$u_i^k(x) = \max_j \sum_{h=0}^{x} p_{ij}(h) u_j^{k-1}(x-h)$
$\forall i \in N, i \neq d, (i,j) \in A, x \in \mathbb{N},$
$\frac{(\tau^k - \delta)}{\Delta t} + 1 \leq x \leq \frac{\tau^k}{\Delta t}$
end

A. CUDA

CUDA (Compute Unified Device Architecture) is a hardware-software parallel computing architecture developed by Nvidia that allows using the GPUs for general-purpose computing.

The host computer transfers data to the device's memory and calls a special function called the kernel. When calling the kernel function, two parameters are set: the number of blocks and the number of threads in the block. Each thread executes the same set of instructions, but with different data elements. Streams within one block can exchange the results of calculations using the shared memory mechanism.

Fig. 1 presents the computation model.

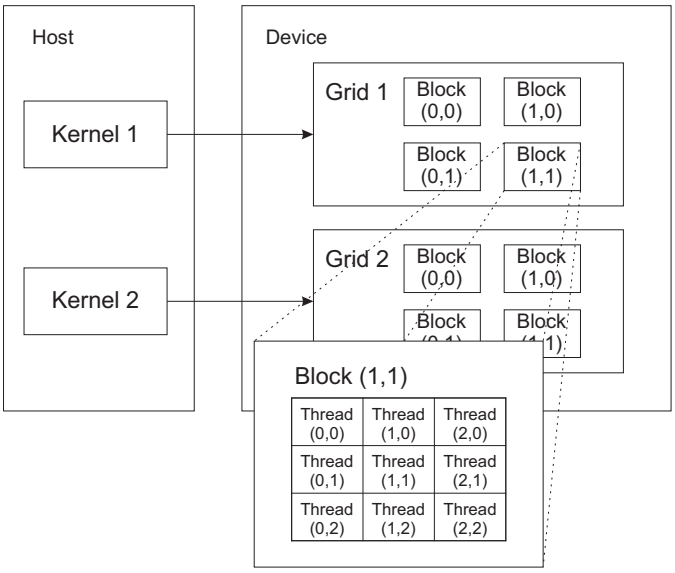

Fig. 1. Computation model.

B. Parallel algorithm implementation

Introduce the notation:

$$u_{ij}(x) = \sum_{h=0}^{x} p_{ij}(h) u_j(x-h),$$

that is, $u_{ij}(x)$ is the probability of reaching the terminal vertex d from the origin vertex i during the time interval x when moving on the road edge (i, j).

Algorithm 1 is executed sequentially in time (a cycle by the variable k), however, the calculation of the probabilities of arrival $u_i^k(x)$ can be performed in parallel mode. The implementation of this process in CUDA consists of calling the kernel function (Algorithm 2), which calculates the probability of reaching the terminal vertex from each vertex of the graph at a given time step.

Algorithm 2: Parallel algorithm

for $k = 1, 2, \cdots, L$ **do**
  process$<< N_c, T_c >>(k)$
end

function `process(`k`)`**:**
  `/* calculate index of the processed vertex` x `*/`
  $x = threadIdx.x + blockDim.x * blockIdx.x$;
  load $p(1), \cdots, p(t)$ and corresponding u from the memory ;
  calculate $u_{ix}^k \forall (i, x) \in A$;
  calculate $u_x^k = \max_i u_{ix}^k$;
return

In the Algorithm 2, $threadIdx$ is the thread index, $blockDim$ is the block dimension, $blockIdx$ is the processing block index.

Thus, the calculation of the probabilities of reaching the destination node for a given time interval will be performed in parallel for each node of the graph in a separate stream of the graphic accelerator.

IV. Experiments

Experimental studies of the base and parallel algorithms were carried out for a large-scale transportation network of Samara, Russia, consisting of 47274 road links (edges) and 18582 nodes.

To compare the running time of the base and parallel implementations of the algorithm for finding a reliable shortest path, 6 pairs of different origin-destination nodes were selected in the graph, after which the navigation problem was solved for each pair of nodes and different days of the week, the start time and the time budget. The origin-destination nodes were selected so that the average travel time was from 15 to 60 minutes. A total of 6300 experiments were conducted.

Characteristics of the PC used: Intel Core i7-9700K 3.60 GHz, 64 GB RAM, graphics accelerator GeForce RTX 2080 Ti. The average running time of the algorithms is presented in the table 1.

TABLE I. Average running time of the algorithms

	Base algorithm	CUDA implementation
Running time, sec	20.58	3.88

The implementation of the algorithm using the CUDA architecture allows reducing the running time by an average of 5 times.

The running time of the algorithm depends on the time budget, which determines the number of iterations. Fig. 2 shows the dependency of the running time of the base and parallel algorithms on the time budget.

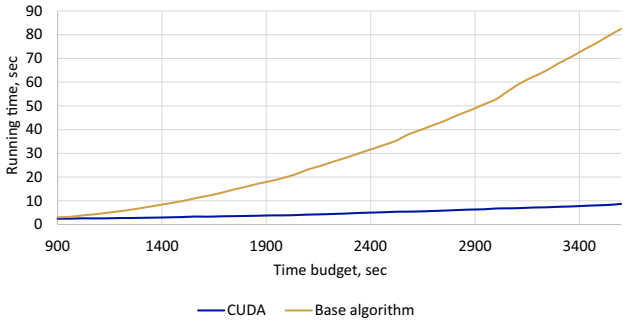

Fig. 2. The dependence of the algorithm running time on the time budget.

With an increase in the time budget, the gain of the parallel algorithm in running time increases.

A detailed analysis of the running time of the parallel algorithm allows us to conclude that most part of the running takes the data exchange between the host computer and the device.

V. Conclusion

In this paper, we consider the problem of finding a reliable shortest path in a stochastic network that maximizes the probability of arriving at a destination within a predetermined period of time. A parallelization strategy for the algorithm using a graphic accelerator was developed and a parallel algorithm was implemented on the CUDA software-hardware architecture.

Experimental studies conducted on a large-scale transportation network of Samara, Russia have shown that parallel implementation of the algorithm reduces the computation time by an average of 5 times.

Further research may focus on developing an algorithm with fewer data transfers between the host computer and the graphics accelerator.

Acknowledgment

The work was partially supported by RFBR research projects nos. 18-07-00605 A, 18-29-03135-mk.

REFERENCES

[1] A.A. Agafonov, A.S. Yumaganov and V.V. Myasnikov, "Big data analysis in a geoinformatic problem of short-term traffic flow forecasting based on a K nearest neighbors method," Computer Optics, vol. 42, no. 6, pp. 1101–1111, 2018. DOI: 10.18287/2412-6179-2018-42-6- 1101-1111.

[2] P. Chen, R. Tong, G. Lu and Y. Wang, "The alpha-reliable path problem in stochastic road networks with link correlations: A moment-matching-based path finding algorithm," Expert Systems with Applications, vol. 110, pp. 20-32, 2018. DOI: 10.1016/j.eswa.2018.05.022.

[3] S. Samaranayake, S. Blandin and A. Bayen, "A tractable class of algorithms for reliable routing in stochastic networks," Transportation Research Part C: Emerging Technologies, vol. 20, no. 1, pp. 199-217, 2012. DOI: 10.1016/j.trc.2011.05.009.

[4] Y. Nie and X. Wu, "Shortest path problem considering on-time arrival probability," Transportation Research Part B: Methodological, vol. 43, no. 6, pp. 597-613, 2009. DOI: 10.1016/j.trb.2009.01.008.

[5] L. Fu and L.R. Rilett, "Expected shortest paths in dynamic and stochastic traffic networks," Transportation Research Part B: Methodological, vol. 32, no. 7, pp. 499-516, 1998. DOI: 10.1016/S0191-2615(98) 00016-2.

[6] S. Gao and I. Chabini, "Optimal routing policy problems in stochastic time-dependent networks," Transportation Research Part B: Methodological, vol. 40, no. 2, pp. 93-122, 2006. DOI: 10.1016/j.trb.2005.02.001.

[7] A. Chen and Z. Ji, "Path finding under uncertainty," Journal of Advanced Transportation, vol. 39, no. 1, pp. 19-37, 2005. DOI: 10.1002/atr.5670390104.

[8] B. Y. Chen, W. H. K. Lam, A. Sumalee, Q. Li, H. Shao, and Z. Fang, "Finding Reliable Shortest Paths in Road Networks Under Uncertainty," Networks and Spatial Economics, vol. 13, no. 2, pp. 123-148, 2013. DOI: 10.1007/s11067-012-9175-1.

[9] Y. Fan and Y. Nie, "Optimal routing for maximizing the travel time reliability," Networks and Spatial Economics, vol. 6, no. 3-4, pp. 333–344, 2006. DOI: 10.1007/s11067-006-9287-6.

[10] A.A. Agafonov and V.V. Myasnikov, "Method for the reliable shortest path search in time-dependent stochastic networks and its application to GIS-based traffic control," Computer Optics, vol. 40, no. 2, pp. 275-283, 2016. DOI: 10.18287/2412-6179-2016-40-2-275-283.

[11] S. Samaranayake, S. Blandin and A. Bayen, "Speedup techniques for the stochastic on-time arrival problem," OpenAccess Series in Informatics, 2012, vol. 25, pp. 83-96. DOI: 10.4230/OASIcs.ATMOS.2012.83.

[12] M. Niknami and S. Samaranayake, "Tractable pathfinding for the stochastic on-time arrival problem," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9685, pp. 231-245, 2016.

[13] G. Sabran, S. Samaranayake and A. Bayen, "Precomputation techniques for the stochastic on-time arrival problem," Proceedings of the Workshop on Algorithm Engineering and Experiments, 2014, pp. 138-146. DOI: 10.1137/1.9781611973198.13.

[14] M. Abeydeera and S. Samaranayake, "GPU parallelization of the stochastic on-time arrival problem," 21st International Conference on High Performance Computing, HiPC, 2014. DOI: 10.1109/HiPC.2014.7116896.

[15] A. Agafonov and V. Myasnikov, "Stochastic On-time Arrival Problem with Levy Stable Distributions," 4th International Conference on Intelligent Transportation Engineering (ICITE), Singapore, pp. 227-231, 2019. DOI: 10.1109/ICITE.2019.8880254.