Speedup of Dynamic Route Search for Large-scale Microscopic Traffic Simulation

Naoki Mita¹, Hideaki Uchida*, Hideki Fujii¹, Shinobu Yoshimura¹
¹Department of Systems Innovation, School of Engineering, The University of Tokyo
*uchida@save.sys.t.u-tokyo.ac.jp

Received: May 19, 2016; Accepted: November 1, 2016; Published: March 31, 2018

Abstract. The current approach to solving large-scale traffic problems is with the simulation of microscopic vehicle behaviors. However, few simulators are capable of conducting such simulations due to the high computational cost. In this study, we propose a new route search method using a simplified network to achieve speedup. We developed a practical implementation for a microscopic traffic simulator and applied the proposed method to a real road network in Okayama city under a dynamic routing condition. The results show that the proposed method reduces the number of links and nodes for route searches by 80% and the route search execution time by 98%.

Keywords: Traffic Simulator, Route Search, Simplified Network

1. Introduction

To solve traffic problems, various countermeasures have been discussed. These countermeasures include the improvement of road infrastructures and signal control, and the introduction of Intelligent Transport Systems (ITS) technology. To implement these countermeasures, it is important to assess beforehand whether the investment will have the desired effect. Because conducting experiments with human participants is often impractical due to costs and safety restrictions, virtual experiments using computer simulations are a good alternative. In many reported traffic simulators, it is possible to reproduce microscopic vehicle behaviors at an intersection. However, the modeling range may not necessarily be on a local scale, because small changes in the traffic can easily spread over a wide region. Moreover, to evaluate disaster preparedness and environmental problems in a traffic simulation, it is essential to assess traffic on a city scale. The resolution of such large-scale problems and the reproduction of detail are important requirements for a practical microscopic traffic simulator, although there is a tradeoff...
with the execution time for route searches. Thus, in this paper we focus on the route search function in a microscopic traffic simulator. Specifically, we present the practical implementation of a simplified network approach and evaluation to solve complicated and dynamic problems over a large area in a system called ADVENTURE_Mates (Multi-Agent based Traffic and Environment Simulator), developed by the authors [1, 2, 3, 12].

2. Related Work

As previously mentioned, the route search is the most time-consuming process in a microscopic traffic simulation. In this section, we summarize prior research on the speedup of route searches. In general, a route search in a traffic simulation is equivalent to the problem of finding the shortest path in graph theory. The route search uses an algorithm to find a path between two nodes in a graph such that the sum of the weights of the constituent links is minimized. A graph consisting of nodes and links corresponds to a road network consisting of intersections and sections, which are weighted by the length or mean travel time between nodes. The algorithm can be divided into two phases: preprocessing and query processing. First, an index is created in the preprocessing phase to speed up the search without knowledge of the origin or destination node. The second phase is the query phase, in which the origin and destination are known.

2.1. Static Route Search

Assuming the road network is static, the preprocessing phase needs to be conducted only once, and the resulting index can be used for a large number of queries on the same network. The conventional preprocessing techniques for static networks are transit node routing [4, 5], highway dimension [6], and highway node [7]. These techniques automatically extract a list of frequently selected nodes or links as the shortest path. This phase limits the search range around the origin and destination, because the list represents a group of important traffic hubs, such as junctions or highways. A state-of-the-art technique for preprocessing is hub-labeling [8], based on highway dimensions. This technique improves the query processing time, although it increases the memory space and the execution time required for preprocessing. However, the output of all of these techniques cannot be reused in the query phase when the weights of the links are changed dynamically.

2.2. Dynamic Route Search

The classic route search algorithm A* [9] is widely used for finding the shortest path. A* is an extension of the Dijkstra’s algorithm [10] and uses heuristic information to improve the search. Because the Euclidean distance between the origin and destination is heuristic, no preprocessing is required. Executing an algorithm without preprocessing is very beneficial for traffic simulations in which the routes of a large number of vehicles must be searched individually.
Thus, A* has been adopted by many simulators using a multi-agent system approach, including ADVENTURE_Mates. However, it is difficult to run a route search using A* with time complexity $O(n^2)$ in a large-scale road network. Although Real-time A* (RTA*) [11], which is an approximating technique, guarantees the real-time property, the sum of the execution times of the route searches increases exponentially, because a larger network typically means a higher number of vehicles in a simulation. Learning Real-time A* (LRTA*) [11] and Q-routing [12], which are techniques based on learning, are the same as RTA*.

2.3. Domain Decomposition Method for Route Search

It is difficult to apply the techniques for static conditions, described in section 2.1, under dynamic traffic conditions. Even RTA* cannot be utilized for a large-scale road network and a huge number of vehicles. Also, a route search is generally difficult to parallelize. Therefore, we describe methods to speed up the route search by partitioning a network into multiple regions. By using the domain decomposition method, an algorithm can be hierarchically managed. Two types of routes are defined in the decomposition method. A local route search is a search within a subdomain containing the origin and destination point, and a global route search is a search between subdomains. Domain decomposition can increase the parallel efficiency. In the literature, the following two well-known approaches have been reported as a solution for the large-scale dynamic network problem: a look-up table approach [13] and a simplified network approach [14, 15]. Ogura et al. [13] split the nationwide road network of Japan ($2.26 \times 10^5$ nodes and $5.21 \times 10^5$ links) into 47 subdomains organized into 513 sub-subdomains, and proposed the division of the search domain into a hierarchy. They recalculated the look-up table so that it lists not only the shortest routes from every boundary node of the sub-subdomain to the boundary nodes of the subdomain containing that sub-subdomain but also the route between two arbitrary boundary nodes of the subdomain. Although this approach is useful, recalculation of the look-up table is very time consuming when the travel cost changes dynamically. Nakamura et al. [14] proposed a simplified network. In this approach, the roads of the original network can be classified into two types of links: main roads and narrow streets. Then, a simplified network in which narrow streets are omitted is generated. The route search executed in this network is an approximation method. It is more robust to dynamic changes in travel cost than the look-up table approach.

3. Proposed Route Search Method Using a Simplified Network

Although a simplified network approach is an approximation method, greater improvement of execution time is expected in comparison with other dynamic approaches, and speedup with parallel computing is also expected. However, the simplified network was only conceptual in previous research and an actual simulator was not implemented. In this study, we propose a
practical method that automatically creates a simplified network from a real road network, and we implement this simplified network in our microscopic traffic simulator. Figure 1 shows the procedure for the microscopic traffic simulation with the simplified network. We introduce each step of this procedure in the following sections.

![Flowchart of a microscopic traffic simulator with a simplified network](image)

**Fig. 1** Flowchart of a microscopic traffic simulator with a simplified network

### 3.1. Generation of Simplified Network

Nakamura et al. [14] defined arterial links to be main roads and gathered links to be small streets in an area surrounded by arterial links. To reduce the degree of a real road network, they proposed a method for simplifying a real road network using combinations of these two types of links. This is a natural modeling approach because a driver’s route search is known to be hierarchical. For example, actual drivers tend to select main roads and neglect small roads in a route search. This section focuses on the automatic construction procedure for the simplified network proposed in this study. The overview of the generation of the simplified network is shown in Figure 2.
3.1.1. Generation of Arterial Network

In this study, we define an arterial link as a road having two or more lanes on each side, and an arterial node as the point where an arterial link connects with another link. When the threshold of the arterial link is defined, the simplified network is uniquely determined. The threshold should be set according to the scale of simulation area. The larger threshold might result in a sparser simplified network, smaller route search execution time and lower accuracy path. However, note that even if nodes are connected to the same road, it is possible that one is an arterial node and the other is not, because the number of lanes can change in the middle of the road. The network consisting of only arterial links and arterial nodes generated from an original network is called an arterial network. Both of an original network and an arterial network are directed graphs. In addition, the point adjacent to an arterial link is called an adjacent node, and the number of arterial links incident to the arterial node is called the arterial degree. For example, let $N$ be a node whose arterial degree is 0, $A_1$ be a node whose arterial degree is 1, $A_2$ be a node whose arterial degree is 2, and $A_3$ be a node whose arterial degree is 3 or more. In the arterial network, $A_1$ is a terminal point, $A_2$ is a node that represents a curve, and $A_3$ corresponds to an intersection. The arterial network can be constructed from the original network as follows.

First, we prepare the node list of the initial nodes of the network modification. The list is consisted of all $A_3$s. Then, the process explained below is conducted sequentially from the initial node, then the 1st adjacent node, and the 2nd adjacent node to reduce the size of the arterial network by disconnecting redundant links. Table 1 shows the details of the process. The arterial network is completed by execution of this processing for all of the initial nodes.
Links in original network are classified into an arterial link (solid line) and a non arterial link (dashed line) in accordance with the definition in Figure 2 (i). Nodes in original network are also labeled in accordance with its arterial degree. As a result of network modification, arterial network which is shown in Figure 2 (ii) is generated.

| Initial | 1st | 2nd | Reconnection of arterial link |
|---------|-----|-----|------------------------------|
| (a) $A_3$ | $N$ | - | delete between initial and 1st |
| (b) $A_1$ | - | | connect between initial and 1st |
| (c) $A_2$ | $N$ | | connect between initial and 1st |
| (d) $A_1$ | | | connect between initial and 2nd / omit 1st adjacent node |
| (e) $A_2$ | | | any one of (c)–(f) is repeated for 3rd and following |
| (f) $A_3$ | | | connect between initial and 2nd / omit 1st adjacent node |
| (g) $A_3$ | - | | connect between initial and 1st |

3.1.2. Extraction of Polygon Area

Polygon areas are extracted from the arterial link network for the purpose of constructing gathered links. They are the smallest units of the simulation and cannot be further divided into polygons. Extraction of a polygon area is equivalent to finding cycles in the arterial link network with a depth-limited search. The only node that can compose a cycle is $A_3$. Furthermore, the difference between the number of $A_1$s connected to the node and its arterial degree must be 2 or more. The node list $V$ that satisfies these conditions is given as input in the depth-limited search algorithm. However, because the procedure finds a cycle within the maximum search depth, called the limit, a cycle whose length is larger than the limit cannot be extracted. This limit is imposed because a sufficiently large cycle could include a smaller cycle as a subset. Otherwise, generating gathered links in this polygon area is impractical, because the interior is a very sparse network due to geographical factors such as mountains or rivers. Therefore, it is possible to extract a polygon area efficiently by specifying a suitable limit. Figure 2 (ii) shows that 2 polygon areas (a pentagon and a triangle) are extracted in an arterial network.

3.1.3. Generation of Gathered Links

The gathered link represents the routes between arterial links in each polygon area. The candidates for the origin and destination of the gathered link are the curves, $A_2$, closest to the
midpoints of each arterial link. The gathered link is generated with a local route search between candidates using only small streets in the polygon areas. If a route is reachable, then it is defined as a gathered link. In an \(n\)-sided polygon area, the gathered links constitute an \(n\)-order complete graph under the above processing. One example of a simplified network is shown in Figure 3. When any of the gathered links are selected as the result of a route search in the simplified network, the obtained route is mapped onto the corresponding realistic link in the real network.

![Fig. 3 Example of a simplified network [14]](image)

### 3.2. Calculation of Link Cost

In this study, both the static and dynamic costs are taken into account in the route search. Total link cost \(C(t)\) is calculated by adding the static cost \(C_S\) and the dynamic cost \(C_D(t)\), as defined in formula (1). \(C(t)\) and \(C_D(t)\) are defined in formula (2),

\[
C(t) = C_S + C_D(t) \tag{1}
\]

\[
C_S = \alpha_1 RL + \alpha_2 RW + \alpha_3 NL + \alpha_4 NR
\]

\[
C_D(t) = \alpha_5 MTT(t) \tag{2}
\]

where \(RL\) is the road length, \(RW\) is the road width, which is the inverse of the number of lanes, \(NL\) is the number of left-hand turns, and \(NR\) is the number of right-hand turns. In addition, \(MTT(t)\) is the mean travel time at simulation time \(t\), and \(\alpha_1 - \alpha_5\) are parameters, that Nakamura et al. estimated from probe-car data [14]. In the simplified network, generated by the procedure presented in the previous sections, the total link cost of each arterial and gathered link is calculated. The dynamic cost is recalculated at fixed time intervals to deal with the dynamic traffic flow changes. Then, the dynamic cost of the gathered link is calculated.
from that of the arterial link. The dynamic cost of gathered link $AE$ is the sum of arterial links $AB$ and $BE$ in Figure 3. Similarly, $AG$ is the average of $(OC + CF)$ and $(BE + ED)$. Therefore, it is possible to reduce the volume of recalculations by updating the link cost.

### 3.3. Hierarchical Route Search

As shown in Figure 1, the simplified network is generated in addition to the original network. In this section, we describe the details of the hierarchical route search with the proposed method and these two networks. Specifically, the original network is used for the local route search within a polygon area, and the simplified network is used for the global route search across polygon areas.

As an example, the route search is divided into four cases based on whether the origin and destination belong to the simplified or the original network. Likewise, these four cases are divided into two cases each, based on the relative position between the origin and the destination, namely, whether they are in the same polygon area. Table 2 shows the eight possible cases, where “Local” denotes the execution of the local route search within a polygon area, and “Global” denotes the execution of the global route search in the simplified network.

Figure 4 is the route search process for case (d) between different polygon areas. This search is the most complicated. In this case, $N$ is both the origin and the destination (red nodes) in the original networks (dashed lines). Subsequently, the nearest arterial nodes of these two points are defined as the arrival node (blue nodes). The arrival node is necessarily in the simplified network (solid lines). We can generate sequential routes by merging three routes, which are the local route from the origin to the arrival node of the origin, the global route from the arrival node of the origin to the arrival node of the destination, and the local route from the arrival node of the destination to the destination. All other cases are generated as subsets of the procedure in case (d).

**Table 2** Classification of hierarchical route search

| Case | Origin   | Destination | Route search between          |          |
|------|----------|-------------|--------------------------------|----------|
|      |          |             | same polygon area | different polygon areas |
| (a)  | Simplified | Simplified  | Local                          | Global   |
| (b)  | simplified | Original    | Local                          | Global + Local |
| (c)  | Original  | Simplified  | Local                          | Local + Global |
| (d)  | Original  | Original    | Local                          | Local + Global + Local |
4. Numerical Experiment

4.1. Experimental Condition

The proposed method is implemented in ADVENTURE_Mates and applied to the simulation of a real road network. The real network is a 3 km × 3 km square area around the central district of Okayama city, as shown in Figure 5. Figure 6 shows the simplified network automatically generated with the proposed method. In Figure 6, the blue lines indicate arterial links and the red lines indicate gathered links. The simplified network results in a 77.7% reduction in the number of links (from 381 to 85), and a 78.1% reduction in the number of nodes (from 265 to 58). Table 3 summarizes the properties for evaluating the link costs for the four cases. The route search for the two proposed cases is executed hierarchically by using both the simplified and the original networks, whereas the two conventional cases are executed only in the original network.
However, all cases employ A* as the common route search algorithm. The link costs for conventional case 2 and proposed case 2 include the dynamic factor. In these cases, the recalculation interval is 600 sec. Moreover, the number of trials is five for each of these four cases.

| Case            | Algorithm | Network       | Link cost          |
|-----------------|-----------|---------------|--------------------|
| Conventional case 1 | Original | Static        |                    |
| Conventional case 2 | A*       | Original      | Static + Dynamic   |
| Proposed case 1   | Original + Simplified | Static       |                    |
| Proposed case 2   | Original + Simplified | Static + Dynamic |                 |

4.2. Results

Figure 7 shows the total execution times of the conventional and proposed route search methods at 3,000 sec. According to the results for the total execution time, the improvement is only around 10–20%. On the other hand, according to the results of the execution time for the route search at 3,000 sec, proposed case 1 has a 98.5% improvement in performance compared with conventional case 1, as shown in Figure 8. Similarly, proposed case 2 has a 97.9% improvement in performance compared with that of conventional case 2. It is known that the proportion of the total execution time spent on the route searches becomes larger as the network scale becomes larger, as described above. Therefore, when applied to a network larger than this city-scale network, further speedup is expected with the proposed route search method.
It is qualitatively confirmed that the output of the proposed method is similar to that of the conventional method. In fact, the increase in the path length for the proposed cases is suppressed to about 1.2 times at most, as shown in Table 4, which shows the ratio of the path lengths obtained by the conventional and the proposed cases. Although the proposed method is an approximation, its accuracy of the path lengths is sufficient. Table 4 shows the average path lengths of all origin-destination pairs in the original network given in Figure 5. Case 1 shows the comparison between conventional case 1 and proposed case 1, and Case 2 shows the comparison between conventional case 2 and proposed case 2. Because the link cost of Case 2 depends on the travel time, the table shows the results for each simulation time. Figure 9 shows the results obtained by the route search for proposed case 2 at intervals of 1,200 sec. The figure clearly demonstrates how the selected route changes, depending on instantaneous traffic conditions.

### Table 4  Comparison of path lengths

| Proposed and conventional cases | Simulation time [sec] | Proposed / Conventional |
|---------------------------------|------------------------|-------------------------|
| Case 1 (conv. 1 and prop. 1)    | -                      | 1.122                   |
| Case 2 (conv. 2 and prop. 2)    | 1,200                  | 1.081                   |
|                                 | 2,400                  | 1.105                   |
|                                 | 3,600                  | 1.173                   |
|                                 | 4,800                  | 1.220                   |
|                                 | 6,000                  | 1.165                   |
|                                 | 7,200                  | 1.165                   |

![Fig. 9](image)  Route transitions for proposed case 2

### 5. Conclusion

In this study, we implemented a route search algorithm for a simplified network in a microscopic traffic simulator. We applied this algorithm to a simulation of traffic in Okayama city,
Japan, in order to evaluate the performance of the proposed method. By employing the hierarchical route search for the simplified road network, the proposed method was able to reduce the number of links and the number of nodes by approximately 80%, and then to reduce the route search execution time by 98%. Examples of the route search with the conventional and the proposed methods were presented to evaluate path lengths. In addition, the applicability of the method under dynamic changes in traffic conditions was indicated by observing the route transitions. However, further investigation is needed to determine the optimal recalculation intervals for minimizing the dynamic link costs. Additional studies could also implement and evaluate the proposed method in parallelized MATES, such as reported by Kohashi et al. [3]. The reduction in the execution time is expected to be large in simulations of larger networks.

**Acknowledgment**

This work was supported by JSPS KAKENHI Grant Number 15H01785.

**References**

[1] S. Yoshimura: MATES : Multi-Agent based Traffic and Environmental Simulator -Theory, Implementation and Practical Application-, *Computer Modeling in Engineering and Sciences*, Vol.11, No.1, pp.17-26, (2006).

[2] H. Fujii and S. Yoshimura: Precise Evaluation of Vehicles Emission in Urban Traffic Using Multi-agent-based Traffic Simulator MATES, *Computer Modeling in Engineering and Sciences*, Vol. 88, No. 1, pp.49-64, (2012).

[3] T. Kohashi, S. Bunya, H. Fujii and S. Yoshimura: Parallelization of Intelligent Multi-Agent based Traffic and Environment Simulator MATES, *Transactions of JSCES*, No.20100003, (2010). (in Japanese)

[4] H. Bast, S. Funke, D. Matijevic, P. Sanders and D. Schultes: In Transit to Constant Time Shortest-Path Queries in Road Networks, *Proceedings of the 9th Workshop on Algorithm Engineering and Experiments*, pp.46–59, (2007).

[5] H. Bast, S. Funke, P. Sanders and D. Schultes: Fast Routing in Road Networks with Transit Nodes, *Science*, Vol.316, No.5824, p.566, (2007).

[6] I. Abraham, A. Fiat, A. V. Goldberg and R. F. Werneck: Highway dimension, shortest paths, and provably efficient algorithms, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.782-793, (2010).

[7] D. Schultes and P. Sanders: Dynamic Highway-Node Routing, Proceedings of the 6th International Workshop on Experimental Algorithms, pp.66-79, (2007).

[8] I. Abraham, D. Delling, A. V. Goldberg and R. F. Werneck: A Hub-based Labeling Algorithm for Shortest Paths in Road Networks, *Proceedings of the 10th international conference on Experimental algorithms*, pp.230-241, (2011).
[9] P. E. Hart, N. J. Nilsson and B. Raphael: A Formal Basis for the Heuristic Determination of Minimal Cost Paths, IEEE Transactions on Systems Science and Cybernetics, Vol.4, No.2, pp.100-107, (1968).

[10] E. W. Dijkstra: A Note on Two Problems in Connexion with Graphs, Numerische Mathematik, Vol.1, pp.269-271, (1959).

[11] R. E. Korf: Real-time heuristic search, Artificial Intelligence, Vol.42, No.2–3, pp.189-211, (1990).

[12] H. Uchida, H. Fujii, S. Yoshimura and S. Arai: Learning Routing Policy for Changes in Road Network, IPSJ Journal, Vol.53, No.11, pp.2409-2418, (2012). (in Japanese)

[13] I. Iimura and M. Kato: A Route Finding Method for the Nationwide Road Networks of Japan by Restricting Search Areas through the Use of Look-up Table, Journal of Information Processing, Vol.35, No.12, pp.2831-2841, (1994). (in Japanese)

[14] T. Nakamura, T. Yoshii and R. Kitamura: Development of a Simplified Network Which Represents all Roadway Links, Infrastructure Planning Review, Vol.23, No.2, pp.441-446, (2006). (in Japanese)

[15] S. Kitamura, T. Yoshii, T. Yamamoto and T. Morikawa: Field Observation of Vehicular Movement in Neighborhood Streets toward Development of Large-scale Traffic Simulation Model, Proceedings of the 27th Conference of Infrastructure Planning and Management, (2003). (in Japanese)