Time-dependent link travel time approximation for large-scale dynamic traffic simulations

Genaro Peque, Jr.[1] Hiro Harada[2] and Takamasa Iryo[3]

[1,2,3]Kobe University, Department of Civil Engineering, Japan
[1]gpequejr@panda.kobe-u.ac.jp
[2]171t139t@stu.kobe-u.ac.jp
[3]iryo@kobe-u.ac.jp

Abstract. Large-scale dynamic traffic simulations generate a sizeable amount of raw data that needs to be managed for analysis. Typically, big data reduction techniques are used to decrease redundant, inconsistent and noisy data as these are perceived to be more useful than the raw data itself. However, these methods are normally performed independently so it wouldn't compete with the simulation's computational and memory resources.

In this paper, we propose a data reduction technique that will be integrated into a simulation process and executed numerous times. Our interest is in reducing the size of each link's time-dependent travel time data in a large-scale dynamic traffic simulation. The objective is to approximate the time-dependent link travel times along the $y-$ axis to reduce memory consumption while insignificantly affecting the simulation results. An important aspect of the algorithm is its capability to restrict the maximum absolute error bound which avoids theoretically inconsistent results which may not have been accounted for by the dynamic traffic simulation model. One major advantage of the algorithm is its efficiency's independence from the input data complexity such as the number of sampled data points, sampled data's shape and irregularity of sampling intervals. Using a 10x10 grid network with variable time-dependent link travel time data complexities and absolute error bounds, the dynamic traffic simulation results show that the algorithm achieves around 80%−99% of link travel time data reduction using a small amount of computational resource.

Keywords: large-scale dynamic traffic simulation, piecewise linear approximation, route planning, parallel computing.

1 Introduction

1.1 Background

Large-scale dynamic traffic simulations are becoming widespread partly due to the exponential growth of the computer's processing power, memory capacity and parallelization capability. Along with it is the increasing need to manage the sizeable amount of raw data that it generates. Typically, big data reduction techniques are used to decrease redundant, inconsistent and noisy data as it is perceived to be more useful than

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

the raw data itself. However, these methods are normally performed independently so it wouldn't compete with the simulation's computational and memory resources. A major challenge is when data reduction is integrated into the simulation process that is executed numerous times since it needs to be simple, fast and efficient.

In this paper, we are interested in reducing the size of each link's time-dependent travel time data at each iteration of a large-scale dynamic traffic simulation. The simulation is planned to be executed using a parallel computer with a distributed memory architecture. Large-scale dynamic traffic simulations are normally conducted on massive networks with substantial number of links (e.g. a road network of Western Europe has 42.6 million links) [1, 2]. Since travel time is a function of a single variable that is strictly monotone (time), the problem can be defined as a piecewise linear approximation of the time-dependent link travel time data. There are already many existing algorithms developed for this problem but most are interested in retaining the data's significant features. Moreover, the space and time complexities of these algorithms are usually dependent on the input complexity such as the number of sampled data points, sample data's shape and irregularity of sampling intervals.

The piecewise linear approximation of polygons has several applications in various fields such as pattern recognition [3], motion planning in robotics [4], vector graphics and cartographic generalization [5], among others. The goal is to reduce the size or complexity of a given data as much as possible while retaining its significant features. One of the widely known piecewise linear approximation algorithms that have been studied in various fields is the Ramer-Douglas-Peucker algorithm [6, 7, 8, 9]. This algorithm decimates a curve composed of line segments into a similar curve with fewer points. The major advantage of this algorithm is its efficiency's independence from the input data complexity. However, the algorithm is still computationally inefficient for large-scale simulations and its error criterion isn't well-suited for the reduction of time-dependent link travel time data since it isn't possible to restrict the absolute error bound. This makes it inaccurate and inconsistent when applied to multiple links with different input complexities.

Piecewise linear approximation algorithms usually rely on error bounds or complexity minimization that further depends on the context of the application. In this paper, the application lies in the context of link travel time approximation for time-dependent route planning in large-scale dynamic traffic simulations. Increasing amounts of time-dependent travel time data are usually generated as the number of iterations or the length of the simulation time period of the traffic simulation increases. Depending on the design of the traffic simulator [10, 11, 12], some of the data may be, among other things, redundantly generated, insignificantly noisy and/or slowly changing (Fig. 1). All of which can be approximated according to an absolute error bound without significantly affecting the simulation results. Thus, an absolute error bound restriction capability is an integral part of the approximation method in order to maintain an accurate and consistent simulation result. Typically, travel time data is generated by a time-dependent piecewise linear function that is strictly monotone in time. This implies that time-dependent link travel time approximation is a special case of the piecewise linear approximation of polygons; the piecewise linear curve is an open polyline and a function of one variable that is strictly increasing.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

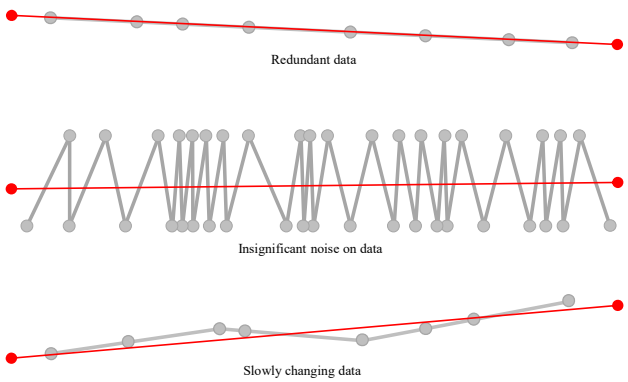

Fig. 1. Some types of data that can be approximated.

There are several papers focusing on the approximation of piecewise linear functions that might be used for large-scale dynamic traffic simulations with absolute error restriction capabilities [13, 14, 15]. For example, in Tomek [13] two simple heuristic algorithms were proposed for functions of one variable. Both algorithms used a limit on the absolute error values. The first algorithm was fast and gave satisfactory results for sufficiently smooth functions while the second algorithm was slower but gave better approximations for less well-behaved functions. In Imai and Iri [14], using the idea of Suri [16], they developed an algorithm for the edge-visibility problem [17] which produces an optimal solution for a piecewise linear function of one variable that is strictly monotone. The idea of the algorithm is to compute a minimum link path through a tunnel using a light source covering the entry of a tunnel. The tunnel is produced by an absolute upper and lower bound on the function given by an exogenously supplied absolute error value. Then, a light source illuminates a part of the tunnel and divides it into several invisible parts and a visible part. The intersection of boundaries of an invisible part and a visible part containing the exit of the tunnel is called a window where the first point of the approximated function lies. From the window, another light source illuminates part of the tunnel which creates another window. This process continues until the other end of the tunnel is reached. A major advantage of this algorithm is its $O(N)$ complexity where N is the number of points representing the piecewise linear curve.

In the transportation context, Neubauer [15] applied Imai and Iri's algorithm on a real-world time-dependent link travel time data. The algorithm was primarily used to preprocess the time-dependent travel time data to reduce its size and use it for route planning. However, there are three main concerns of its use in the approximation of time-dependent link travel time data. First, although Imai and Iri's [14] algorithm is simple and optimal given an absolute error value, lower bounds for link travel times need to be checked and possibly adjusted as it may produce values lesser than some links' free-flow travel times or even produce negative values. Neubauer [15] addressed this by using only relative upper bounds. When lower bounds were considered, it yielded first-in, first-out (FIFO) violations. Second, the calculation of line distances, intersections and angles to create a link's travel time estimate still requires a substantial

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

amount of computational and memory resources. Furthermore, the algorithm's efficiency is dependent on the input complexity, specifically the sample data's shape. Third, an integral part of the algorithm is to create a polygon by offsetting the piecewise linear curve above and below by a constant value. This implies that 2 copies of the N points of the piecewise linear curve (above and below) also needs to be stored in the memory during the approximation process. This is challenging when used during a large-scale dynamic traffic simulation with limited memory capacity such as a parallel computer with distributed memory architecture.

1.2 Contribution of this Paper

In this paper, a piecewise linear approximation algorithm is developed to reduce the size of the time-dependent link travel time data generated by the large-scale dynamic traffic simulation. Our main objective is to reduce each link's memory consumption at each iteration given that the maximum error bound can be restricted to avoid theoretically inconsistent results not accounted for by the dynamic traffic simulation model.

We propose an algorithm that only requires linear interpolation calculations with respect to the given input points. The motivation is to develop an algorithm that is simple, fast and whose efficiency is independent of the input complexity such as the number of sampled data points, sampled data's shape and irregularity of the sampling intervals. Additionally, the algorithm should be capable of absolute error bound restriction to avoid inaccurate and inconsistent results. The idea is to use linear interpolation along the $y-$ axis to calculate and reduce the Euclidean distance of the real link travel time and approximated link travel time. The approximated time-dependent link travel data will then be retrieved by each driver for route planning. Subsequently, we will show that only the Euclidean distances from the given input points to the estimated line segments need to be checked against the criterion function because these are the only points where the absolute maximum error for each ordered subset can occur. Furthermore, since linear interpolation only calculates points between any of the given input points, the estimated link travel time values are assured to be bounded by these points in all directions. One disadvantage of the algorithm is that the points of the approximated function are restricted to the subset of the input function. It doesn't allow arbitrary points [16, 18, 19]. However, this is insignificant for our application.

1.3 Outline of this Paper

This paper is structured as follows. In the next section, a brief description of a dynamic traffic simulation and the importance of link travel time data approximation is presented. In section 3, we formally define the piecewise linear approximation problem and introduce some notations. Additionally, the proposed piecewise linear approximation algorithm for time-dependent link travel time data reduction is introduced including the method in which travel time is retrieved after the approximation. In Section 4, using a 10x10 grid network with variable time-dependent link travel time data complexities and error bounds, results show that the algorithm is very simple, fast and efficient both in time and space and is highly suitable for integration in large-scale dynamic

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

traffic simulation. A summary of the importance of our contribution and conclusion is presented in the last section.

2 Large-Scale Dynamic Traffic Simulation

2.1 Travel Time Approximation in a Dynamic Traffic Simulation

Traffic simulators that can perform large-scale dynamic traffic simulations are becoming increasingly popular as these provide more detailed means to represent the interaction between travel choices, traffic flows, and time and cost measures in a temporally coherent manner. In most cases, these are iteratively conducted and involves an interplay of the vehicular traffic loading and travelers' route assignments [20] until a stopping criterion is met (Fig. 2).

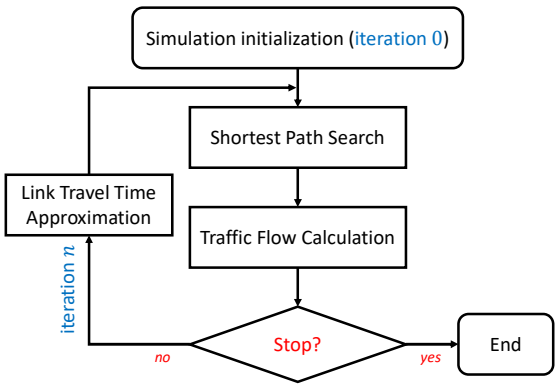

Fig. 2. Dynamic traffic simulation flowchart.

An iteration in a simulation usually represents a time period (from peak hours to an entire day). Within this time period, time-dependent travel time data is generated for each link sampled in specific time intervals. Thus, the longer the simulation time period, the larger the size of the time-dependent link travel time data that needs to be stored for travelers' time-dependent route planning. Although it might not be necessary to store the data of each iteration for route planning, these are usually necessary for post-processing analysis.

In large-scale dynamic traffic simulations, massive networks will have substantial number of links that generate time-dependent travel time data. This would require a large amount of memory for storage (depending on the simulation time period and the number of iterations) which is usually a very limited resource. However, some of these sampled data may be, among other things, redundantly generated, insignificantly noisy and/or slowly changing (Fig. 1). All of which can be approximated according to an absolute maximum error bound without significantly affecting the simulation results. Thus, other than requiring a simple, fast and efficient algorithm, a necessary feature is the capability of the algorithm to restrict the absolute error bound consistently and accurately. In some dynamic traffic simulations, travelers' route assignments are assigned

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

based on the shortest paths to their destinations (i.e. a deterministic route choice model) [21]. This in turn depends on the time-dependent travel time data of the links along these routes. An inaccurate or inconsistent approximation would lead to a different simulation result if a deterministic route choice model is a requirement. More importantly, even if a deterministic route choice model isn't a requirement it may still lead to theoretically inconsistent results not accounted for by the dynamic traffic simulation model.

Therefore, we are motivated in reducing the size of the time-dependent link travel time data given that we are able to restrict the absolute error bounds of the approximation process.

3 The Piecewise Linear Approximation Algorithm

3.1 Piecewise Linear Approximation

The problem of retaining significant features of polygonal curves through iterative approximation was formalized by Ramer [6]. The problem is to approximate a polygon represented by points using an iterative approximation algorithm that produces another polygon with a lesser number of points. More formally, an arbitrary two-dimensional plane curve is represented by an ordered set C of N consecutive points along the curve. The points in set C can be interpreted as the vertices p_i of a polygon P with $N-1$ edges. A polygon $\tilde{P}$ with a reduced number $\tilde{N}-1$ of edges, whose vertices p_k coincide with the vertices of P, then corresponds to an ordered subset $\tilde{C}$ of points such that $\tilde{C} \subseteq C$. The problem is to find an ordered subset $\tilde{C}$ of the ordered set C such that the polygon $\tilde{P}$ with vertices $p_k \in \tilde{C}$ approximates the curve based on an application-dependent error criterion. A desirable property for the ordered subset $\tilde{C}$ is that its size be as small as possible.

The ordered subset $\tilde{C}$ of vertices p_k divides the set C into ordered subsets $S_k \in S$ of the following form,

$$S_k = \{p_s, p_{s+1}, \dots, p_{|S_k|}\}, \tag{1}$$

where $p_s = p_{k-1}$ and $p_{|S_k|} = p_k$; $s = 1,2,\dots,|S_k|$ and $k = 1,2,\dots,\tilde{N}$. The points p_s and p_{s+1} are consecutive points in polygon P and the points p_{k-1}, p_k are consecutive points in polygon $\tilde{P}$. The ordered subset $\tilde{C}$ also partitions the polygon $\tilde{P}$ into curve segments and S_k contains the points belonging to the $k-$ th curve segment. Then, the following constraints hold between the ordered sets $C, \tilde{C}$ and S_k,

$$\cup_{k=2}^{\tilde{N}} S_k = C \text{ and } \left(\cup_{k=2}^{\tilde{N}} (S_k \cap S_{k+1})\right) \cup \{p_1\} \cup \{p_N\} = \tilde{C}. \tag{2}$$

Constraints for the generation of the polygon $\tilde{P}$ can now be conveniently included into the conditions for the subsets S_k and $\tilde{C}$. Conditions on the subsets S_k correspond to conditions for the approximation of a curve segment $S_k \in C$ by a straight-line segment $\overline{p_{k-1}\, p_k}$ and conditions on $\tilde{C}$ can be regarded as global properties.

Generally, the objective is to find a subset $\tilde{C} \subseteq C$ of vertices with a minimum number of elements satisfying a specified criterion function $g(S_k) \leq \theta$, where θ is a constant, given a polygon P with a set of vertices $C = \{p_1, \dots, p_i, \dots, p_N\}$. The solution to

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

the problem with an unknown number of inequalities is complicated but methods such as dynamic programming [9] or graph searching techniques [22] can be used to get optimal solutions at the expense of additional computational resource. An example of a criterion function $g(S_k)$ used by Ramer [6] is the maximum-distance, $g(S_k) = \max\big(dist(p_{k-1}, p_k)\big) \leq \alpha$ where $dist(p_1, p_2)$ is the Euclidean distance between the points p_1 and p_2; $p_i = \big(x_{p_i}, y_{p_i}\big)$. In Ramer [6], an iterative approximation method was used to divide C into subsets which were tested against the maximum-distance criterion. If the subset satisfied the criterion, the subset was retained. Otherwise, the subset was divided and the resulting subsets were tested. The iteration was terminated when all subsets S_k satisfied the criterion $g(S_k) \leq \theta$. Various application-dependent criteria have been developed to test subsets such as perimeter error [9], error of area [5], mean square error or maximum deviation. Another possible criterion not based on error is to minimize the size of $\tilde{C}$ to a desirable number of points. In this paper, we deal with the former, i.e. $g(S_k) \leq \theta$.

3.2 Vertical Linear Interpolation Algorithm

The polygon P in this study is a special case of the piecewise linear approximation problem in [6] due to its special structure. The y coordinates of the x and y coordinates of points p_i in the ordered set C are defined by a function $y = f(x)$ where x is strictly monotone, $p \equiv (x, y) \in \mathbb{R}_+^2$ and $f: \mathbb{R}_+ \to \mathbb{R}_+$. This implies that polygon P is an open polyline with no intersecting edges. From now on this will be denoted as P'. The values of the x and y coordinates for each $p_i \in C$ will be denoted as x_{p_i} and y_{p_i}, respectively.

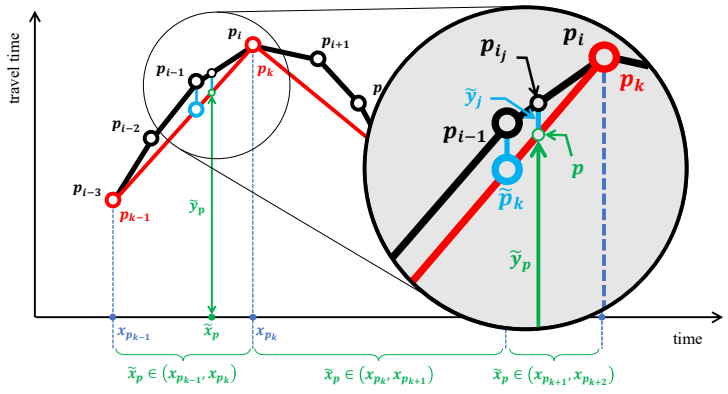

Fig. 3. Link travel time data approximated by the vertical linear interpolation algorithm.

For a time-dependent link travel time data, an estimated $\tilde{y}_p$ value is a time-dependent travel time for a particular time $\tilde{x}_p$ of a link along a certain route of a traveler's destination in a specific iteration. Hence, in the piecewise linear approximation of the time-dependent travel time data defined by the polyline P', we propose an algorithm which linearly interpolates the values $y_{\tilde{p}_k}$ of projected points $\tilde{p}_k$ to the line $\overline{p_{k-1}\, p_k}$ from each point p_s in the ordered subset S_k for all $S_k \in S$ as shown in Fig. 3 above. Linear

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

interpolation is chosen because it is very simple and fast to calculate and its efficiency is independent of the input complexity. The algorithm is characterized by the projection of each $p_s \in S_k$ to the line $\overline{p_{k-1}\, p_k}$ orthogonal to the $x-$ axis. The value of a projection from p_s is calculated using the point's x_{p_s} value and is determined by the equation,

$$y_{\tilde{p}_k} = \left| y_{p_{k-1}} + \left(x_{p_s} - x_{p_{k-1}}\right)\left(\frac{y_{p_k} - y_{p_{k-1}}}{x_{p_k} - x_{p_{k-1}}}\right)\right|. \tag{3}$$

From the values created by equation (3), we are interested in the criterion function $g(S_k)$ which represents the line with the maximum length among all of the lines projected by all the points $p_s \in S_k$ to the line $\overline{p_{k-1}\, p_k}$. This is given by $dist(p_s, \tilde{p}_k) = \left|y_{p_s} - y_{\tilde{p}_k}\right|$ where p_s and $\tilde{p}_k$ have both the same x_{p_s} value. More formally, the criterion function is defined by the equation,

$$g(S_k) = \max_{p_s}\{dist(p_s, \tilde{p}_k) | p_s \in S_k\} \leq \alpha\beta = \theta, \tag{4}$$

where the α is a constant and β is a scaling factor. The scaling factor β is necessary when the same constant α is used for different polylines P' with different properties. Then, each of the $g(S_k)$ value for each S_k is added into a max heap Q where its root node Q_r has a value of $\tilde{g} = \max_k g(S_k)$. The variable $\tilde{g}$ represents the maximum Euclidean distance of $\tilde{P}'$ from P' along the $y-$ axis.

The criterion in the selection of the next ordered subset $S_{\tilde{k}}$ to divide into two smaller ordered subsets $S_{\tilde{k}_1}$ and $S_{\tilde{k}_2}$ is given by $S_{\tilde{k}} = \arg\max_{S_k} g(S_k)$ where $S_{\tilde{k}} = S_{\tilde{k}_1} + S_{\tilde{k}_2} - \left(S_{\tilde{k}_1} \cap S_{\tilde{k}_2}\right), \tilde{k}_1 - 1 = \tilde{k} - 1$ and $\tilde{k}_2 = \tilde{k}$. Additionally, the ordered subsets $S_{\tilde{k}_1}$ and $S_{\tilde{k}_2}$ replaces $S_{\tilde{k}}$ in the set S. The point $p_{\tilde{\imath}}$ where the ordered subset $S_{\tilde{k}}$ is divided at is denoted by,

$$p_{\tilde{\imath}} = \arg\max_{p_s}\{dist(p_s, \tilde{p}_k) | p_s \in S_k\}, \tag{5}$$

where $S_{\tilde{k}_1} = \left\{p_{\tilde{k}_1 - 1}, \ldots, p_{\tilde{\imath}}\right\}$, $S_{\tilde{k}_2} = \left\{p_{\tilde{\imath}}, \ldots, p_{\tilde{k}_2}\right\}$ and $p_{\tilde{\imath}} = S_{\tilde{k}_1} \cap S_{\tilde{k}_2}; \tilde{k} - 1 < \tilde{\imath} < \tilde{k}$. When $S_{\tilde{k}}$ is divided into $S_{\tilde{k}_1}$ and $S_{\tilde{k}_2}$, it creates new $g\left(S_{\tilde{k}_1}\right)$ and $g\left(S_{\tilde{k}_1}\right)$. However, both $g\left(S_{\tilde{k}_1}\right)$ and $g\left(S_{\tilde{k}_1}\right)$ aren't guaranteed to be less than $\tilde{g}$. Thus, both $g\left(S_{\tilde{k}_1}\right)$ and $g\left(S_{\tilde{k}_1}\right)$ need to be evaluated and placed into the max heap Q. The process of ordered subset selection and division is repeated until the stopping criterion $\tilde{g} \leq \alpha\beta$ is satisfied or the max heap is empty.

Table 1 below summarizes the vertical linear interpolation algorithm (Algorithm 1) through a pseudocode.

Table 1. Vertical linear interpolation algorithm pseudocode.

Algorithm 1 (Vertical linear interpolation algorithm)

Required input: C, α and β
Initialization: $S_2 = C,\ g(S_2) = 0$ and $\tilde{C} \leftarrow p_1, p_N$
for each $p_s \in S_2 \backslash p_1, p_N$ **do**
  if $g(S_2) < dist(p_s, \tilde{p}_2) = \left|y_{p_s} - y_{\tilde{p}_2}\right|$ **do**

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

$g(S_2) \leftarrow dist(p_s, \tilde{p}_2)$
max heap, $Q \leftarrow g(S_2)$
$\tilde{g} \leftarrow Q_r$ where Q_r is the max heap's root node
$\tilde{C} \leftarrow p_{\tilde{\imath}}[Q_r]$
while $\tilde{g} > \alpha\beta$ and $Q \neq \emptyset$ **do**
  $\tilde{k} \leftarrow k[Q_r]$ and $p_{\tilde{\imath}} \leftarrow p_{\tilde{\imath}}[Q_r]$
  remove Q_r from Q
  split $S_{\tilde{k}}$ into $S_{\tilde{k}_1}$ and $S_{\tilde{k}_2}$ along $p_{\tilde{\imath}}$
  replace $S_{\tilde{k}} \in S$ with $S_{\tilde{k}_1}$ and $S_{\tilde{k}_2}$
  for each $p_{s_i} \in S_{\tilde{k}_i} \backslash p_{\tilde{k}_i - 1}, p_{\tilde{k}_i}, i \in \{1, 2\}$ **do**
    if $g(S_{k_i}) < dist(p_{s_i}, \tilde{p}_{k_i}) = \left| y_{p_{s_i}} - y_{\tilde{p}_{k_i}} \right|$ **do**
      $g(S_{k_i}) \leftarrow dist(p_{s_i}, \tilde{p}_{k_i})$
  for each $S_{k_i}, i \in \{1, 2\}$ **do**
    max heap, $Q \leftarrow g(S_{k_i})$
  $\tilde{g} \leftarrow Q_r$
  $\tilde{C} \leftarrow p_{\tilde{\imath}}[Q_r]$
Return $\tilde{C}$

In Fig. 4 below, a visual demonstration of how the vertical linear interpolation algorithm iteratively approximates an open polyline P' is shown. Iteration 1 shows S_k as the entire polyline P' and its line segment $\overline{p_{k-1}\, p_k}$ approximation (red line). Additionally, it shows that vertical lines orthogonal to the $x-$ axis (blue lines) are projected from the points $p_s \in S_k$ (black dots) to the line segment $\overline{p_{k-1}\, p_k}$. The vertical line with the maximum length $\tilde{g}$ (green line) is also shown. The set S_k is then divided into two line segments $\overline{p_{k-1}\, p_k}$ and $\overline{p_k\, p_{k+1}}$ in iteration 2 from the point $p_{\tilde{\imath}}$ at which $\tilde{g}$ occurred in the previous iteration. This process is repeated for each S_k where $\tilde{g}$ occurs until the stopping criterion $\tilde{g} \leq \alpha\beta$ is met. After the algorithm terminates, the points of the polyline $\tilde{P}'$ denoted by the ordered subset $\tilde{C}$ is returned.

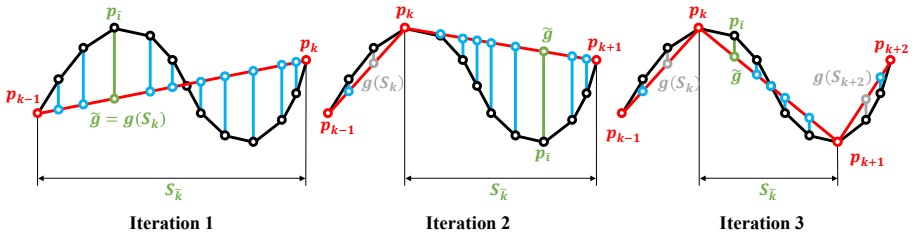

Fig. 4. Iterative piecewise linear approximation of an open polyline using Algorithm 1.

A condensed algorithm flowchart for Algorithm 1 is shown in Fig. 5 below. The algorithm is performed to each link in the network at each iteration using the criterion function $\tilde{g} \leq \alpha\beta$. The algorithm flowchart is a generalized representation of a piecewise linear approximation algorithm where the type of estimation (violet box) and stopping criterion (diamond) can be replaced based on the application.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

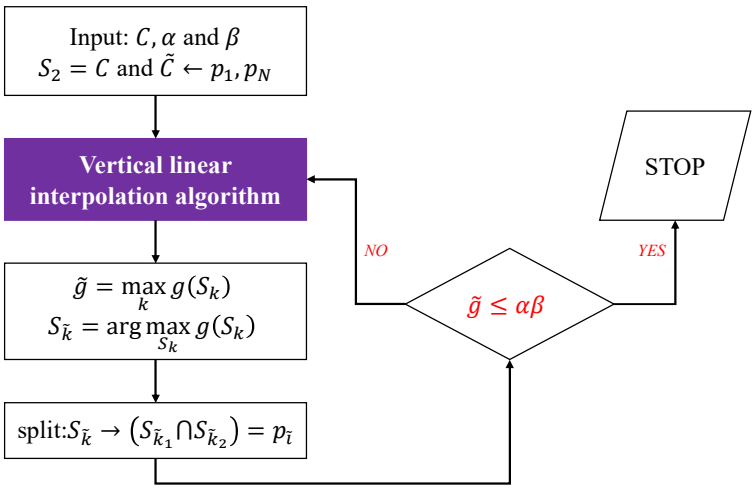

Fig. 5. Algorithm flowchart for the vertical linear interpolation algorithm.

The results below show that the maximum Euclidean distance between the real and estimated time-dependent link travel time data produced by Algorithm 1 are less than or equal to the maximum error bound $\alpha\beta$. Moreover, it also shows that Algorithm 1 only needs to check $dist(p_i, \tilde{p}_k) = |y_{p_i} - y_{\tilde{p}_k}|$ for all p_i in order and not any interpolated values to satisfy this.

Lemma 1. Using Algorithm 1, the Euclidean distances between any real link travel time $\tilde{y}_{p_{i_j}}$ belonging to the line segments $\overline{p_s\, p_{s+1}}$, $\overline{p_{s+1}\, p_{s+2}}$, ..., $\overline{p_{|S_k|-1}\, p_{|S_k|}}$, where $(p_s, p_{s+1}) \in S_k \times S_k$, and its corresponding estimated travel time $\tilde{y}_p$ in the line segment $\overline{p_{k-1}\, p_k}$ along the $y-$axis is bounded by $\tilde{y}_j \in [0, g(S_k)]$ for any S_k.

Proof. For a calculated $S_k = \{p_s, p_{s+1}, \dots, p_{|S_k|}\}$, we know that $p_s = p_{k-1}$ and $p_{|S_k|} = p_k$ in the line segment $\overline{p_{k-1}\, p_k}$. Hence, if the point p in $\tilde{x}_p$ is either $p = p_s$ or $p = p_{|S_k|}$, then $\tilde{y}_j = 0$. Then, it just needs to be shown that for any $p_{k-1} < p < p_k$, $\tilde{y}_j$ is bounded by $(0, g(S_k)]$ and $g(S_k)$ will occur at one of the points $p_s \in S_k$.

Let us create triangles by connecting line segments $\overline{p_s\, \tilde{p}_k}$, $\overline{\tilde{p}_k\, p_k}$ and $\overline{p_k\, p_s}$, $\forall p_s \in S_k$. For each triangle, each $\overline{p_s\, \tilde{p}_k}$ is orthogonal to the $x-$axis with length $dist(p_s, \tilde{p}_k) = \tilde{y}_j$. By the triangle side splitter theorem and Pythagorean theorem, any line segment inside the triangle parallel to the line segment $\overline{p_s\, \tilde{p}_k}$ is guaranteed to be proportionally shorter than $\overline{p_s\, \tilde{p}_k}$. Since S_k is finite, $g(S_k)$ is also guaranteed to be one of the $\{\overline{p_s\, \tilde{p}_k} | p_s \in S_k\}$. This would immediately imply that $\tilde{y}_j \in [0, g(S_k)], \forall S_k$. Similar arguments hold if line segments $\overline{p_s\, \tilde{p}_k}$, $\overline{\tilde{p}_k\, p_{k-1}}$ and $\overline{p_{k-1}\, p_s}$ are used.

Theorem 1. Given an ordered set C representing the open polyline P', an error bound α and a scaling factor β, the piecewise linear approximation algorithm, Algorithm 1, will produce an ordered subset $\tilde{C}$ that represents the polyline $\tilde{P}'$ with an absolute maximum Euclidean distance bounded by $\alpha\beta$ from P'.

Proof. From Lemma 1, we know that $g(S_k)$ will only occur at one of the given input points $p_s \in S_k, \forall S_k \in S$ at each iteration. Since the ordered set S is finite and $\tilde{g} =$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

$\max_k g(S_k)$, if Algorithm 1 only tests $\tilde{g}$ against $\alpha\beta$, N is finite and Algorithm 1 only stops when all calculated $\tilde{g} \leq \alpha\beta$, then, Algorithm 1 guarantees that all $dist(p_i, \tilde{p}_k) \leq \alpha\beta, \forall p_i \in C$.

3.3 Link Travel Time Retrieval

In order to retrieve the travel time $\tilde{y}_p$ for a specific time $\tilde{x}_p$ in a link, the tuple $(x_{p_{k-1}}, x_{p_k}) \in \tilde{C} \times \tilde{C}$ which contains the time $\tilde{x}_p$ (i.e., $x_{p_{k-1}} \leq \tilde{x}_p \leq x_{p_k}$) is located using binary search in the ordered set $\tilde{C}$. This tuple represents the line segment $\overline{p_{k-1}\, p_k}$ in the polyline $\tilde{P}'$ which contains the point p, i.e. $p \in [p_{k-1}, p_k]$. Then, in order to determine the link travel time $\tilde{y}_p$ at time $\tilde{x}_p$, linear interpolation is used,

$$\tilde{y}_p = \left| y_{p_{k-1}} + \left(\tilde{x}_p - x_{p_{k-1}}\right)\left(\frac{y_{p_k} - y_{p_{k-1}}}{x_{p_k} - x_{p_{k-1}}}\right)\right|. \tag{6}$$

Fig. 3 above shows an example of how the travel time $\tilde{y}_p$ might be retrieved from the ordered subset $\tilde{C}$ along the line segment $\overline{p_{k-1}\, p_k}$ for a specific time $\tilde{x}_p$. Additionally, it shows that the error between the estimated link travel time $\tilde{y}_p$ and the real link travel time $\tilde{y}_{p_{i_j}}$ is given by $\tilde{y}_j \leq \alpha\beta$. Each link travel time retrieval has a complexity of $O(\log \tilde{N})$ which implies that the lesser the size of the ordered subset $\tilde{C}$, the better.

4 Numerical Simulation

Algorithm 1 was tested on a 10×10 grid network with 5 different absolute error bounds; α equal to 1, 30, 180 (3 minutes), 300 (5 minutes) and 600 (10 minutes) seconds. Additionally, travel time sampling intervals were randomly performed between 1 and 20 seconds.

A plot of the last iteration of the dynamic traffic simulation shown in Fig. 6 below had a one-hour simulation time period and was iteratively calculated 100 times using 500 different random demand patterns from 9 randomly selected origin-destination (OD) pairs and one CPU. The scaling factor β was set equal to 1. In the plot of the figure, if all the values of the real and estimated link travel time data are the same (*No approximation*), it will appear as a straight line with a 45° angle. The more the absolute error value is increased, the more obvious the difference between the real and estimated travel time values become (e.g. error differences based on the absolute maximum error bounds). In the simulation, there were no noticeable difference in calculation time between the *No approximation* and *10-minute error* since only a few links (around 35% of the total links in the network) of the 10×10 grid network were used and the simulation time period was very short. Furthermore, the time-dependent link travel time data of an unused link appeared to be approximated by two points (i.e. the start and end points) which only required a single iteration of the approximation algorithm. The average data reduction percentage of the simulation was at 98%.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

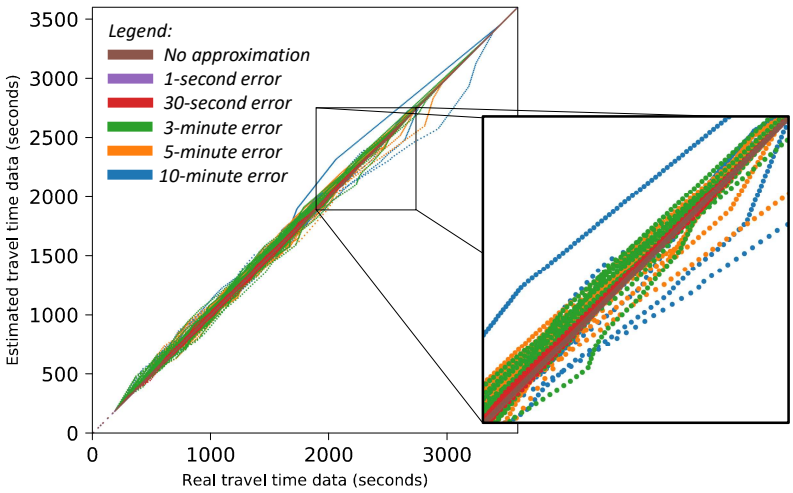

Fig. 6. Real and estimated link travel time data of all links sampled at random intervals.

In order to suitably check the effect of the piecewise linear approximation on the dynamic traffic simulation calculation time, 400 randomly selected OD pairs were fixed and a 24-hour simulation time period was iteratively calculated 100 times using 16 CPUs. The dynamic traffic simulation utilized around 98% of the total links in the network. Two peak periods where simulated in the 24-hour simulation time period, from 7:00 AM to 9:00 AM and 5:00 PM to 7:00 PM.

A table showing the calculation time and average data reduction percentage of the entire dynamic traffic simulation is presented below.

Table 2. Calculation time and average data reduction in a dynamic traffic simulation.

Approximation error ($\alpha\beta$)	Calculation time (s)	Average data reduction (%)
No approximation	5301	0.00
10-minute error	5305	99.91
5-minute error	5315	98.90
3-minute error	5350	95.10
30-second error	5486	89.72
1-second error	5539	82.18

Table 2 above shows that the entire 24-hour simulation iteratively calculated 100 times took around 90 minutes without the time-dependent link travel time approximation. Using a 1-second absolute error bound on the piecewise linear approximation of each link, a significant savings in memory is evident (82.18%) while only increasing the calculation time to 4 minutes. A 1 second error bound would normally be enough

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

to ensure that the simulation results will be insignificantly affected by the time-dependent link travel time approximation. As the value of $\alpha\beta$ is increased, the average data reduction rate also increased and the calculation time decreased, however, the time-dependent travel time data's accuracy decreased. This is expected since the number of iterations required by the vertical linear approximation algorithm also decreases and thus, the number of points in the set $\tilde{C}$ decreases.

5 Conclusion

In this paper a piecewise linear approximation algorithm was developed to reduce the size of the time-dependent link travel time data generated by a large-scale dynamic traffic simulation.

We were able to show that the developed algorithm is simple and fast because it only requires the linear interpolation of the given input points and a simple criterion function to conduct the approximation. This makes it very suitable for large-scale dynamic traffic simulation integration. Correspondingly, it can also be used to preprocess or post-process time-dependent link travel time data. Furthermore, we showed that it is efficient because even with its simplicity and speed, it is very effective in reducing the time-dependent travel time data regardless of the input complexity. More importantly, the algorithm is capable of producing accurate and consistent approximations at each iteration by restricting the absolute maximum error. This is very important for traveler's route planning since it avoids theoretically inconsistent results unaccounted for by the simulation model or even avoids altering simulation results.

Results from our dynamic traffic simulations showed that the algorithm was able to accurately and consistently restrict the absolute maximum error of the real and estimated time-dependent link travel time data. Moreover, it was able to reduce the link travel time's average space consumption by up to 82% using an absolute maximum error value of $\alpha\beta = 1$ second while only increasing the calculation time of the dynamic traffic simulation to 4 minutes. A 1 second error bound is enough to ensure that the simulation results will be insignificantly affected by the time-dependent link travel time approximation.

6 Acknowledgement

This work was supported by the Post K computer project (Priority Issue 3: Development of Integrated Simulation Systems for Hazard and Disaster Induced by Earthquake and Tsunami).

References

1. Delling, D., Wagner, D.: Time-Dependent Route Planning. In: Ahuja R.K., Möhring R.H., Zaroliagis C.D. (eds) Robust and Online Large-Scale Optimization. Lecture Notes in Computer Science, vol 5868. Springer, Berlin, Heidelberg (2009).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44

2. Delling, D., Schultes, D., Wagner, D.: Highway hierarchies star. In: 9th DIMACS Implementation Challenge (2006).
3. Prasad, D., Leung, M., Quek, C., Cho, S.: A novel framework for making dominant point detection methods non-parametric. Volume 30, Issue 11, Pages 843-859, November (2012).
4. Nguyen, V., Gachter, S., Martinelli, A., Tomatis, N., Siegwart, R.: A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. Autonomous Robots. 23 (2): 97 (2007).
5. Visvalingam, M. Whyatt, J.: Line Generalisation by Repeated Elimination of the Smallest Area (Technical report). Discussion Paper. Cartographic Information Systems Research Group (CISRG), The University of Hull 10 (1992).
6. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves. Computer Graphics and Image Processing, 1, 244-256 (1972)
7. Douglas, D., Peucker, T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer 10(2), 112-122 (1973)
8. Duda, R., Hart, P.: Pattern classification and scene analysis. Wiley, New York (1973).
9. Sato, Y.: Piecewise linear approximation of plane curves by perimeter optimization. Pattern Recognition, Vol. 25, No. 12, pp. 1535-1543. Great Britain (1992).
10. Horni, A., Nagel, K., Axhausen, K.: The Multi-Agent Transport Simulation MATSim. Ubiquity Press, London (2016).
11. Krajzewicz, D., Hertkorn, G., Rossel, C., Wagner, P.: SUMO (Simulation of Urban MObility) - an open-source traffic simulation. In: Al-Akaidi A (ed) Proceedings of the 4th Middle East symposium on simulation and modelling (MESM20002), S. 183-187, 4th Middle East Symposium on Simulation and Modelling, Sharjah (United Arab Emirates) (2002).
12. Smith, L., Beckman, R., Baggerly, K., Anson, D., Williams, M..: TRANSIMS: TRansportation ANalysis and SIMulation System: Project Summary and Status (1995).
13. Tomek, I.: Two algorithms for piecewise-linear continuous approximation of functions of one variable. IEEE Transactions on Computers, C-23(4):445-448 (1974).
14. Imai, H., Iri, M.: An optimal algorithm for approximating a piecewise linear function. Journal of information processing, 9(3):159-162 (1987).
15. Neubauer, S.: Space Efficient Approximation of Piecewise Linear Functions. Student research project (Studienarbeit), Universitat Karlsruhe (TH) (2009).
16. Suri, S.: A linear time algorithm with minimum link paths inside a simple polygon. Computer Vision, Graphics, and Image Processing, 35(1):99-110 (1986).
17. Guibas, L., Hershberger, J., Leven, D., Sharir, M., Tarjan, R.: Linear time algorithms for visibility and shortest path problems inside simple polygons. In SCG '86: Proceedings of the second annual symposium on Computational geometry, pages 1-13, New York, NY, USA (1986).
18. Mitchell, J., Polishchuk, V., Sysikaski, M.: Minimum-link paths revisited, Computational Geometry, Volume 47, Issue 6, pp. 651-667 (2014).
19. Kostitsyna, I., Loffler, M., Polishchuk, V., Frank, S.: On the Complexity of Minimum-Link Path Problems. Journal of Computational Geometry, 8(2):80-108 (2016).
20. Chiu, Y. C., Bottom, J., Mahut, M., Paz, A., Balakrishna, R., Waller, T., and J. Hicks, J.: Dynamic Traffic Assignment: A Primer. Transportation Research Circular E-C153, Transportation Research Board, Washington, DC (2011).
21. Cantarella, G., Cascetta, E.: Dynamic Processes and Equilibrium in Transportation Networks: Towards a Unifying Theory. Transportation Science, Vol. 29, No. 4, pp. 305–329 (1995).
22. Nilsson, N.: Problem-Solving Methods in Artificial Intelligence. McGraw Hill (1971).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_44