Towards space-time buffering for spatiotemporal proximity analysis of movement data

Hui Yuan[a,b,c], Bi Yu Chen[b,c,*], Qingquan Li[d,b,c], Shih-Lung Shaw[e,b,c], and William H. K. Lam[f]
[a]School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China; [b]State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan; University, Wuhan, China; [c]Collaborative Innovation Center of Geospatial Technology, Wuhan, China; [d]Shenzhen Key Laboratory of Spatial Smart Sensing and Services, Shenzhen University, Shenzhen, China; [e]Department of Geography, University of Tennessee, Knoxville, TN, USA; [f]Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong

Abstract

Spatiotemporal proximity analysis to determine spatiotemporal proximal paths is a critical step for many movement analysis methods. However, few effective methods have been developed in the literature for spatiotemporal proximity analysis of movement data. Therefore, this study proposes a space-time integrated approach for spatiotemporal proximal analysis considering space and time dimensions simultaneously. The proposed approach is based on space-time buffering, which is a natural extension of conventional spatial buffering operation to space and time dimensions. Given a space-time path and spatial tolerance, space-time buffering constructs a space-time region by continuously generating spatial buffers for any location along the space-time path. The constructed space-time region can delimit all space-time locations whose spatial distances to the target trajectory are less than a given tolerance. Five space-time overlapping operations based on this space-time buffering are proposed to retrieve all spatiotemporal proximal trajectories to the target space-time path, in terms of different spatiotemporal proximity metrics of space-time paths, such as Fréchet distance and longest common subsequence. The proposed approach is extended to analyze space-time paths constrained in road networks. The compressed linear reference technique is adopted to implement the proposed approach for spatiotemporal proximity analysis in large movement datasets. A case study using real-world movement data verifies that the proposed approach can efficiently retrieve spatiotemporal proximal paths constrained in road networks from a large movement database, and has significant computational advantage over conventional space-time separated approaches.

Keywords: Space-time buffering, Space-time overlapping, Spatiotemporal proximity, Movement data, Time geography.

1. Introduction

Spatial buffering is one of the most widely used analysis operations in geographical information science (GIScience) (Goodchild, 1987; Žalik et al., 2003). Given a geographic object and a distance tolerance, spatial buffering constructs a spatial region where the distance between any points to the target geographic object is less than the given tolerance. Spatial buffering is a powerful tool to determine spatial proximity between target geographic objects and surrounding objects, and has served as the fundamental proximity analysis foundation for many spatial analysis methods, such as spatial overlapping, clustering, spatial pattern recognition, etc.

With recent advances in information and communication technologies (ICTs), movement data of huge number of individuals can be collected by various location aware devices. Typical movement data include taxi trajectories (also called space-time paths), smart card data, mobile phone records, social

[*] To cite this paper: Yuan, H., Chen, B.Y., Li, Q.Q., Shaw, S.-L. and Lam, W.H.K., 2018, Toward space-time buffering for spatiotemporal proximity analysis of movement data. International Journal of Geographical Information Science, 32, pp. 1211-1246.

media data, and other user-generated geographical information (Chen et al., 2016a). These movement data record detailed individual activity travel behaviors in both space and time with a very large sample size or even the whole population, and have provided unprecedented opportunities for numerous geographical studies, which have long been constrained by the lack of such comprehensive human activity data (Miller and Goodchild, 2015; Kwan, 2016; Liu et al., 2015; Chen et al., 2018). However, movement data typical have complicated spatial and temporal characteristics. Representation, analysis, and visualization of movement data in both space and time dimensions pose a significant challenge for contemporary GIS platforms and associated spatial analysis functions to model the time dimension in geographical studies (Goodchild, 2013; Long and Nelson, 2013; Yang et al., 2017). The unprecedented wealth of movement data has propelled movement data analysis methods to the forefront of GIScience research (Kwan and Neutens, 2014; Miller and Goodchild, 2015; Shaw et al., 2016).

In the literature, considerable research effort has been devoted to developing movement data analysis methods, such as individual-group pattern recognition, trajectory clustering, classification and outlier detection, etc. (Zheng, 2015). Spatiotemporal proximity analysis determines spatiotemporal proximal space-time paths to a target space-time path, and is a critical step for many movement data analysis methods. In contrast to conventional spatial proximity analysis, spatiotemporal proximity analysis of movement data has to consider space and time dimensions simultaneously. For example, a space-time bundling relationship refers to the convergence of individuals' space-time paths for some joint activities; and cannot be correctly identified unless their space-time paths intersect not only at certain locations but also during the same time period.

To the best of our knowledge, few methods have been developed in the literature for spatiotemporal proximity analysis of movement data. Consequently, many studies have used a brute force approach, determining spatiotemporal proximal paths after checking spatiotemporal proximity relationships between the target and all other paths in the movement dataset (Zheng, 2015). However, the brute force approach is computational intensive for large movement datasets. Several studies have suggested spatiotemporal proximity analysis methods using space-time separated approaches to check for spatial proximity using conventional spatial buffering first, and then temporal coverage, or vice versa (Long et al., 2013; Miller, 2005). Nevertheless, space-time separated approaches can misidentify spatiotemporal proximity relationships, because achieving spatial proximity and temporal coverage separately are necessary rather than sufficient conditions for determining spatiotemporal proximity relationships. Therefore, new methods are required for spatiotemporal proximity analysis in space and time simultaneously.

The current study proposes a new spatiotemporal proximity analysis method considering space and time dimensions simultaneously. The proposed approach is based on space-time buffering, an extension of conventional spatial buffering into space and time dimensions. The remainder of this paper is structured as follows. Section 2 provides a brief review of the movement data analysis literature and summarizes the contributions of the current study. Section 3 describes the spatiotemporal proximity measures of space-time paths to provide the necessary background information, and Section 4 presents the proposed space-time integrated approach, consisting of space-time buffering and several space-time overlapping operations. Section 5 extends the proposed approach to analyze space-time paths constrained by a road network. Section 6 discusses the technique for implementing the proposed approach in the movement database, and Section 7 reports a real-world case study using a large movement dataset. Finally, Section 8 presents our conclusions and recommendations for further study.

2. Literature review

Current quantitative methods for movement data analysis may be broadly classified into three main strands. The first strand of research is related to time geography. Hägerstrand (1970) provided well-defined space-time entities, including space-time path (i.e., trajectory), station, prism, and lifeline, and their spatiotemporal proximity relationships, including space-time intersections and bundling, to model human activity-travel behaviors. In the last two decades, great strides have been made in establishing mathematical formulations of time geographic entities and relationships, and developing effective geo-computational methods for constructing time geographical entities in real road networks (Kim and Kwan, 2003; Kwan and Weber, 2003; Miller, 2005; Neutens et al., 2008; Yu and Shaw, 2008; Miller and Bridwell, 2009; Chen et al., 2016b). The original time geography concepts have been refined by considering uncertainties in travel conditions and anchor locations (Chen et al., 2013; Liao et al., 2014; Charleux, 2015; Kuijpers and Othman, 2017). Visualization tools also have been developed to visually explore time geographical entities and relationships in GIScience software platforms (Kwan, 2000; Shaw and Yu, 2009; Chen et al., 2011).

The second strand of research is related to the literature of moving object databases. These aim to provide database platforms for storing, indexing, and querying massive space-time paths. Moving objects database research started from the work of Güting (Erwig et al., 1999) and Wolfson (Wolfson et al., 1998), and many studies have developed sophisticated spatiotemporal data models, indexes, and query algorithms for handling space-time paths in planar space (Güting and Schneider, 2005). Research has recently focused on extending moving objects databases to process network constrained space-time paths (de Almeida and Güting, 2005; Güting et al., 2006; Li and Lin, 2006; Jeung et al., 2010; Xu and Güting, 2013; Zhang et al., 2016; Xiang et al., 2017).

However, most existing moving objects database techniques only consider the space-time path while ignoring other time geographical entities, such as space-time prism and lifeline. Chen et al. (2016a) extended moving objects data models and indexes to all time geographical entities and their space-time intersection relationships, including path-prism and prism-prism intersections. They proposed a compressed linear reference (CLR) technique to transform three-dimensional (3D) time geographical entities in the (x, y, t) space into two-dimensional (2D) entities in CLR space. Using this proposed CLR technique, conventional spatial databases and indexes can be employed to efficiently handle huge time geographical entities and intersection relationships in CLR space. However, systematic methods for quantifying spatiotemporal proximity of space-time paths in CLR space have not been established.

The last strand of research mainly focuses on the development of quantitative movement data analysis methods. Much research has focused on trajectory clustering to uncover activity travel behaviors by grouping similar space-time paths into clusters. Most current trajectory clustering algorithms incorporate two generic steps: (i) construct a similarity matrix by calculating similarity measures between any pair of space-time paths in the databases, and (ii) group similar space-time paths into clusters based on the constructed similarity matrix. Thus, evaluating similarity between space-time paths is a fundamental task for space-time clustering studies (Dodge et al., 2012; Pei et al., 2013).

Spatiotemporal proximity-based (also called distance-based) measures are the most commonly used similarity measures. Several spatiotemporal proximity-based measures have been proposed using various aggregated metrics of spatial distances between coinciding points of space-time paths at different time stamps (Zheng, 2015; Long et al., 2013). For example, closest pair distance (CPD) uses the minimum metric, Fréchet distance (FD) uses the maximum metric, and average of pairs distance (APD) uses the average metric. Outlier points of space-time paths could have significant impact on similarity evaluation. To address this, longest common subsequence (LCSS) distance evaluates similarity using the number of time stamps where the distance between two space-time paths are less than a given tolerance (Vlachos et al., 2002). These spatiotemporal proximity-based measures provide rigorous methods to quantify path similarity based on spatiotemporal proximity, and have been widely used for many trajectory clustering studies.

Another problem routinely encountered in movement data analysis is in identifying individual-group patterns, such as flock, leadership, and convoy patterns (Long and Nelson, 2013; Zheng, 2015). A flock is a group of at least m individuals travelling together within a circle of some given distance tolerance for at least k consecutive time stamps (Laube et al., 2005; Benkert et al., 2008; Turdukulov et al., 2014). The leadership pattern is a special case of the flock pattern, requiring at least one heading individual in each group (Laube et al., 2005). The flock pattern is restricted to circular group shape, which may not well describe real application group shape. Therefore, the convoy pattern relaxes the circular shape requirement, including any group shape where individuals are density connected with a given distance tolerance (Jeung et al., 2010). Determination of space-time bundling relationships amongst space-time paths is a critical step in individual-group pattern recognition. A complete survey of movement data analysis methods is beyond the scope of this article; interested readers can refer to Zheng (2015) and Long and Nelson (2013).

Although spatiotemporal proximity analysis is fundamental for many movement data analysis methods, few effective methods have been developed. Therefore, this study proposes a space-time integrated approach considering space and time simultaneously. The major contributions of the current study are summarized as follows.

1. Space-time buffering and overlapping operations are proposed for spatiotemporal proximity analysis of movement data. This study proposes space-time buffering as a natural extension of conventional spatial buffering into space and time dimensions. Given a space-time path and spatial tolerance, space-time buffering constructs a space-time region by continuously generating spatial buffers for any location along the space-time path. The constructed space-time region, called the space-time buffer, can delimit all space-time locations whose spatial distances to the target space-time path are less than a given spatial tolerance. Five space-time overlapping operations based on this space-time buffering are introduced to retrieve all spatiotemporal proximal space-time paths, in terms of CPD, FD, APD, and LCSS measures and space-time bundling relationships. Optimality of the proposed space-time overlapping is proved rigidly. The proposed space-time buffering and overlapping provide spatiotemporal proximity analysis foundations for many movement data analysis methods.

2. Space-time buffering and overlapping are further extended to analyze space-time paths constrained in road networks. Spatiotemporal proximity measures, i.e., CPD, FD, APD, LCSS, and space-time bundling, are refined for network constrained space-time paths by explicitly considering movement complexities in road networks. The space-time buffer concept is further extended to road networks, called network space-time buffer. Several lemmas are established to explore geometry characteristics of the network space-time buffer. An efficient geo-computational algorithm is developed to construct the network space-time buffer for each path segment with constant speed, rather than the straightforward approach of continuously generating spatial buffers along the space-time path. Extension of the proposed space-time buffering and overlapping to road networks greatly enhances the usefulness of spatiotemporal proximity analysis of movement data in urban areas, which are normally constrained by road networks.

3. Space-time buffering and overlapping are implemented in the database using the CLR technique. Spatiotemporal proximity measures, i.e., CPD, FD, APD, LCSS, and space-time bundling relationships, are established for space-time paths in CLR space. Then the transformation rule is developed for the equivalent representation of the network space-time buffer as a 2D multi-polygon entity in the CLR space. Consequently, all space-time paths and buffers can be

represented using the CLR technique, and stored and queried as 2D entities in spatial databases. Space-time overlapping can then be efficiently implemented by 2D spatial queries provided by the spatial databases. Implementation of space-time buffering and overlapping in CLR space provides a rigid foundation for quantifying spatiotemporal proximity of space-time paths in CLR space.

3. Spatiotemporal proximity measures for space-time paths

The space-time path, also called the trajectory in the literature, is a key space-time entity in time geography to model individual movement processes (Miller, 2005; Chen et al., 2016a). Figure 1 shows that the space-time path of an individual q, denoted by P^q, is represented by a polyline in (x, y, t) space, where x and y are two spatial dimensions, and t is a temporal dimension. Due to limitations of existing location aware technologies, individual locations are generally observed at regular or irregular intervals. Thus, the observation at time stamp t_i is formulated as a control point $c_i = (x_i, \mathrm{y}_i, t_i)$. Let s_{ij}^q be a path segment connecting two consecutive control points $c_i = (x_i, \mathrm{y}_i, t_i)$ and $c_j = (x_j, \mathrm{y}_j, t_j)$. We assume that the individual is moving along each path segment s_{ij}^q at a constant speed (Miller, 2005; Chen et al., 2016a),

$$v_{ij}^q = \frac{D\left((x_i, y_i), (x_j, y_j)\right)}{t_j - t_i} \tag{1}$$

where $D()$ represents any spatial proximity metric between two points satisfying the following four properties (Long et al., 2013).

(i) Non-negativity: $D\left((x_i, y_i), (x_j, y_j)\right) \geq 0$.

(ii) Reflexivity (uniqueness): $D\left((x_i, y_i), (x_i, y_i)\right) = 0$.

(iii) Symmetry: $D\left((x_i, y_i), (x_j, y_j)\right) = D\left((x_j, y_j), (x_i, y_i)\right)$.

(iv) Triangle inequality: $D\left((x_i, y_i), (x_j, y_j)\right) \leq D\left((x_i, y_i), (x_k, y_k)\right) + D\left((x_k, y_k), (x_j, y_j)\right)$.

In planar space, the Euclidian distance operator, $D_E()$, is commonly used as the spatial proximity metric satisfying the above four properties. Consequently, the path segment s_{ij}^q between two control points becomes a straight line; and P^q can be expressed as a set of consecutive straight line segments. Along the space-time path, individual locations at any time stamp $t_k \in [t_i, t_j]$, denoted by $P^q(t_k)$, can be estimated using linear interpolation as

$$P^q(t_k) = \{(x_k, y_k, t_k) \mid x_k = (1-\lambda)x_i + \lambda x_j; y_k = (1-\lambda)y_i + \lambda y_j; \lambda = (t_k - t_i)/(t_j - t_i)\} \tag{2}$$

Figure 1 shows that P^q is a 3D polyline in (x, y, t) space and the projection onto geographical space forms a 2D polyline L^q, comprising a set of straight line segments, $\{..., l_{ij}^q, ...\}$, where l_{ij}^q is a straight line segment connecting (x_i, y_i) and (x_j, y_j) in geographical space.

Two spatiotemporal proximity relationships are defined for space-time paths in time geography literature, including space-time intersection and bundling (Miller, 2005). Figure 2 shows these two relationships for a simple example. The intersection of two space-time paths refers to the co-existence of two individuals in space and time. Given two space-time paths P^q and P^b, the binary relationship of the space-time intersection, $\delta_{\mathrm{Intersect}}(\mathrm{P}^q, \mathrm{P}^b)$, can be expressed as

$$\delta_{\text{Intersect}}(\mathrm{P}^{\mathrm{q}},\mathrm{P}^{\mathrm{b}}) = \begin{cases} 1, & if \;\; D\left(P^{q}(t_k), P^{b}(t_k)\right) = 0, \exists t_k \\ 0, & otherwise \end{cases} \tag{3}$$

The space-time bundling refers to space-time path convergence for some shared activities within a given period $[t_s, t_e]$ (Miller, 2005). Bundling can occur not only at fixed activity locations but also during movement (e.g. carpooling). Space-time bundling does not require space-time path co-existence but allows spatial tolerance ω. This space-time bundling relationship, $\delta_{\text{Bundle}}(\mathrm{P}^{\mathrm{q}},\mathrm{P}^{\mathrm{b}})$, can be expressed as

$$\delta_{\text{Bundle}}(\mathrm{P}^{\mathrm{q}},\mathrm{P}^{\mathrm{b}}) = \begin{cases} 1, & if \;\; D\left(P^{q}(t_k), P^{b}(t_k)\right) \le \omega, \forall t_k \in [t_s, t_e] \\ 0, & otherwise \end{cases} \tag{4}$$

where $D\left(P^{q}(t_k), P^{b}(t_k)\right)$ is the spatial distance between $P^{q}(t_k)$ and $P^{b}(t_k)$ for the same time stamp, t_k.

Several spatiotemporal proximity metrics have been proposed (Zheng, 2015; Long and Nelson, 2013). As shown in Figure 3(a), given two space-time paths P^{q} and P^{b}, the CPD, denoted by M_{CPD}, uses the minimal distance between two space-time paths during the period of interest, and can be expressed as

$$M_{CPD}(P^{q}, P^{b}) = \min_{\forall t_k}\left(D\left(P^{q}(t_k), P^{b}(t_k)\right)\right) \tag{5}$$

This M_{CPD} metric is minimum (i.e., zero) when the space-time paths intersect. In contrast to M_{CPD}, FD, denoted by M_{FD}, uses the maximum distance between two space-time paths during the period of interest (see Figure 3(b)), and can be expressed as

$$M_{FD}(P^{q}, P^{b}) = \max_{\forall t_k}\left(D\left(P^{q}(t_k), P^{b}(t_k)\right)\right) \tag{6}$$

The FD is best conceptualized using the analogy of a person walking a dog, where FD is the minimum leash length required to connect the dog and its owner walking different space-time paths. The APD, denoted by M_{APD}, uses the average distance between two space-time paths during the period of interest(see Figure 3(c)), and can be expressed as

$$M_{APD}(P^{q}, P^{b}) = \underset{\forall t_k}{mean}\left(D\left(P^{q}(t_k), P^{b}(t_k)\right)\right) \tag{7}$$

Then, we have $M_{CPD}(P^{q}, P^{b}) \le M_{APD}(P^{q}, P^{b}) \le M_{FD}(P^{q}, P^{b})$. Since noise control points can cause large distances between two space-time paths, LCSS, denoted by M_{LCSS}, is commonly adopted for time series analysis to robustly evaluate spatiotemporal proximity. M_{LCSS} metric uses the time duration that the distance between two space-time paths is less than a given spatial tolerance ω (see Figure 3(d)), and can be expressed as

$$M_{LCSS}(P^{q}, P^{b}) = \int_{t_s^q}^{t_e^q} \delta(t_k) dt_k \tag{8}$$

$$\delta(t_k) = \begin{cases} 1, & if \;\; D\left(P^{q}(t_k), P^{b}(t_k)\right) \le \omega \\ 0, & otherwise \end{cases} \tag{9}$$

where $\delta(t_k)$ is the binary proximity relationship, and t_s^q and t_e^q are the start and end times of the target path, respectively. $\delta(t_k) = 1$ indicates that the distance between two paths is less than the given tolerance at time stamp t_k, and $\delta(t_k) = 0$ otherwise.

It should be noted that these four spatiotemporal proximity metrics are used for quantifying continuous space-time paths. In the literature, several discrete proximity metrics have been proposed for discrete forms of space-time paths, which comprise only a set of control points, including discrete Fréchet distance (Eiter and Mannila, 1994), dynamic time wrapping distance (Agrawal et al., 1993), edit distance on real sequence distance (Chen and Ng, 2004), and edit distance with real penalty distance (Chen et al., 2005). Although these discrete proximity metrics are useful for analyzing temporal event processes (Pei et al., 2013), such as seismic sequences of different tectonic units, road traffic congestion events, and extreme weather events in different geographic zones, they can produce biased proximity results for continuous space-time paths, because space-time path control points are not generally collected simultaneously, and the number of control points are not typically the same for different space-time paths. Therefore, discrete proximity metrics are not considered in this study.

4. Space-time integrated approach for spatiotemporal proximity analysis

This section introduces the proposed space-time integrated approach for spatiotemporal proximity analyses of movement data. We first introduce space-time buffering, and then five space-time overlapping operations.

Space-time buffering extends conventional spatial buffering of a point by incorporating the time dimension. Given a spatial tolerance ω, it continuously constructs spatial buffers for an individual q moving along P^q in (x, y, t) space. Analogous to conventional spatial buffering, the space-time region constructed by space-time buffering is called a space-time buffer. Let $STB^q(\omega)$ be the space-time buffer of P^q using spatial tolerance ω, mathematically defined as follows.

Definition 1. Given a space-time path P^q and a spatial tolerance ω, the space-time buffer $STB^q(\omega)$ delimits all possible space-time points (x_k, y_k, t_k) satisfying

$$STB^q(\omega) = \{(x_k, y_k, t_k) \mid D\left((x_k, y_k), P^q(t_k)\right) \le \omega, t_s^q \le t_k \le t_e^q\} \tag{10}$$

Figure 4 shows space-time buffering in planar space can be conceptualized using the analogy of a disk moving along P^q. At any t_k, the center of the disk is located at $P^q(t_k)$ and disk radius is set as ω. Therefore, the spatial distance between any point within the disk to $P^q(t_k)$ is less than ω. $STB^q(\omega)$ in planar space is a 3D entity in (x, y, t) space, and comprises a set of space-time cylinders, where each cylinder delimits the buffer for the corresponding path segment $s_{ij}^q \in P^q$.

It can be proved that the projection of $STB^q(\omega)$ onto geographic space is equivalent to the spatial buffer of corresponding projected polyline L^q as in the following proposition.

Proposition 1. Given a space-time path P^q, the projection of its space-time buffer $STB^q(\omega)$ onto the geographical space is equivalent to the spatial buffer of its projected polyline L^q.
Proof. See Proposition A1 in the appendix.

This space-time buffer, $STB^q(\omega)$, is useful to analyze spatiotemporal proximity relationships between the target path P^a and its surrounding paths. Analogous to conventional spatial overlapping, a space-time query using $STB^q(\omega)$ as the input is called a space-time overlapping operation. In this study, five space-time overlapping operations are introduced, CPD, FD, APD, LCSS, and bundling operations.

The CPD operation finds all space-time paths with M_{CPD} to path $P^q \leq \omega$. Figure 5(a) shows that this operation can be implemented by a buffer-path-intersection query to find all space-time paths intersecting $STB^q(\omega)$. Its optimality is proved as follows.

Proposition 2. The CPD operation can retrieve all space-time paths with M_{CPD} to $P^q \leq \omega$.
Proof. See Proposition A2 in the appendix.

The FD operation finds all space-time paths with M_{FD} to $P^q \leq \omega$. Figure 5(b) shows that this operation can be implemented using the buffer-path-contain query to find all space-time paths contained by $STB^q(\omega)$. Its optimality is proved as follows.

Proposition 3. The FD operation can retrieve all space-time paths with M_{FD} to $P^q \leq \omega$.
Proof. See Proposition A3 in the appendix.

The APD operation finds all space-time paths with M_{APD} to $P^q \leq \omega$. Figure 5(c) shows that this operation can be implemented using two steps. Given a space-time buffer $STB^q(\omega)$, the first step uses the CPD operation to find a set of candidate space-time paths. The second step calculates $M_{APD}(P^q, P^b)$ for each candidate path P^b, and removes P^b if $M_{APD}(P^q, P^b) > \omega$ holds. Its optimality is proved as follows.

Proposition 4. The APD operation retrieves all space-time paths with M_{APD} to $P^q \leq \omega$.
Proof. See Proposition A4 in the appendix.

The LCSS operation finds all space-time paths with M_{LCSS} to $P^q \geq 0$. Figure 5(d) shows that this operation can be implemented using two steps. The first step uses the buffer-path-intersection query to find all candidate paths intersected with $STB^q(\omega)$. The second step obtains M_{LCSS} for each candidate path P^b over the common time period within $[t_s, t_e]$. Optimality is proved as follows.

Proposition 5. The LCSS operation retrieves all space-time paths with M_{LCSS} to $P^q \geq 0$.
Proof. See Proposition A5 in the appendix.

The bundling operation finds all space-time paths bundled with P^q during period $[t_s, t_e]$. Figure 5(e) shows that this operation can be implemented using two steps. The first step uses the buffer path intersection query to find all candidate paths intersected with $STB^q(\omega)$. The second step clips each candidate P^b to obtain its sub-path within $STB^q(\omega)$, and removes P^b if the sub-path does not cover the whole period $[t_s, t_e]$. Optimality is proved as follows.

Proposition 6. The bundling operation retrieves all space-time paths bundled with P^q during period $[t_s, t_e]$.

Proof. See Proposition A6 in the appendix.

The proposed space-time buffering and overlapping operations provide a new space-time integrated approach for analyzing spatiotemporal proximal paths considering space and time simultaneously. This overcomes several drawbacks of conventional space-time separated approaches, which check for spatial proximity first and then for temporal coverage, or vice versa (Long et al., 2013; Miller, 2005). Figure 6 shows the advantages of the proposed space-time integrated approach for a simple example of finding all space-time paths bundled with P^q during time period $[t_s, t_e]$. Using the conventional approach, a spatial buffer of tolerance ω is constructed for the projected polyline L^q. Spatial proximity is first checked to determine two candidate space-time paths, P^b and P^c, whose projected polylines intersect the constructed spatial buffer. For each candidate path, temporal coverage over $[t_s, t_e]$ is checked. Accordingly, both candidate space-time paths are found to be bundled with P^q. However, P^b is misidentified, since it is travelling in a different direction to P^q. Therefore, the conventional space-time separated approach can misidentify spatiotemporal proximal paths, because meeting spatial proximity and temporal coverage separately is necessary rather than sufficient conditions for spatiotemporal proximity relationships. Consequently, a verification step is required to check spatiotemporal proximity relationships (e.g. bundling) of all candidate space-time paths, causing additional computational costs in the spatiotemporal proximity analysis. In contrast, Propositions 2–6 prove that the proposed space-time integrated approach, using necessary and sufficient conditions, can exactly determine spatiotemporal proximal paths.

5. Extensions for analyzing network-constraint space-time paths

5.1. Space-time buffer concept in the road network

Human movements in urban areas are normally constrained by road networks. This section extends the proposed space-time integrated approach to analyze network-constrained space-time paths by explicitly considering road network complexities. A road network can be modelled as a directed graph $G=(N, A, \Psi)$, where N is the set of nodes, A is the set of links and Ψ is the set of turning movements. Each link $a_u \in A$ has a set of attributes, including link identity l_u, length d_u, and travel time, t_u. A movement $\psi_{uv} \in \Psi$ represents the allowed movement (e.g. right turn) from link a_u to its successor link a_v, while movement $\psi_{uv} \notin \Psi$ represents the restricted movement (e.g. left turn) from a_u to a_v. U-turns are only allowed at network nodes, and restricted at other locations on network links (Li et al., 2015).

Figure 7 shows a simple example of network-constrained space-time paths. In the road network, any geographical location (x_i, y_i) is located on a link a_u, and can be referenced as (l_u, m_i) using the linear reference system (LRS) (Miller and Shaw, 2001; Chen et al., 2016a), where $m_i \in (0,1)$ is the location's relative position on link a_u. $m_s = \varepsilon \approx 0$ and $m_e = 1-\varepsilon \approx 1$ refer to the starting and ending nodes of the link, respectively, where ε is a very small tolerance (e.g. $\varepsilon = 10^{-8}$). The LRS location (l_u, m_i) is an equivalent representation of the geographical location (x_i, y_i) in the road network; and can be converted to the geographical location (x_i, y_i) using dynamic segmentation (Miller and Shaw, 2001; Chen et al., 2016a).

A network constrained space-time path P^q can be expressed as a set of control points and a set of path segments in the road network. To maintain path topology in the road network, control points should include not only observations but also network nodes. Between each two consecutive control points $c_i^u = (l_u, m_i, t_i)$ and $c_j^u = (l_u, m_j, t_j)$, we assume individual q moving at constant velocity along link a_u. Accordingly, the path segment s_{ij}^q connecting two control points is not necessarily a straight line, but along underlying link a_u. The individual's travel speed v_{ij}^q on the path segment can be expressed as

$$v_{ij}^q = \frac{m_j d_u - m_i d_u}{t_j - t_i} \tag{11}$$

The individual's location $P^q(t_k)$ at $t_k \in [t_i, t_j]$ can be calculated following linear interpolation as

$$P^q(t_k) = \{(l_u, m_k, t_k) \mid m_k = (1-\lambda)m_i + \lambda m_j; \lambda = (t_k - t_i)/(t_j - t_i)\} \tag{12}$$

This network location (l_u, m_k) can be converted to geographic coordinates (x_k, y_k) using dynamic segmentation. Therefore, the network constrained P^q remains a 3D polyline in (x, y, t) space. Let R_u^q be a sub-path of P^q comprising all path segments on the same link a_u. The projection of R_u^q onto the geographical space is link a_u. Then, the projection of $P^q = <\ldots, R_u^q, \ldots>$ onto geographical space comprises a set of links $<\ldots, a_u, \ldots>$.

All spatiotemporal proximity relationships defined in Section 3, i.e., Eqs. (2–9), are valid for network constrained space-time paths. In contrast to planar space, the distance between two network locations should be evaluated by network distance $D_N()$ rather than Euclidean distance $D_E()$. Let $D_{SP}\left((l_u, m_i), (l_v, m_j)\right)$ be the shortest path distance from (l_u, m_i) to (l_v, m_j) in the road network. Using link length attributes, D_{SP} can be calculated by shortest path algorithms, such as Dijkstra's algorithm considering turn restrictions (Li et al., 2015; Chen et al., 2016b). This shortest path distance D_{SP} is not symmetrical, i.e., $D_{SP}\left((l_u, m_i), (l_v, m_j)\right) \neq D_{SP}\left((l_v, m_j), (l_u, m_i)\right)$, because all network links are directed and allow for movement in only one direction. In this study, network distance $D_N()$ is defined as the minimum metric between distances in opposite directions,

$$D_N\left((l_u, m_i), (l_v, m_j)\right) = \min\left(D_{SP}\left((l_u, m_i), (l_v, m_j)\right), D_{SP}\left((l_v, m_j), (l_u, m_i)\right)\right) \tag{13}$$

It is symmetric, and satisfies non-negativity, reflexivity, and triangle inequality properties.

Since travel speed in the road network varies for different city regions, it is reasonable to use travel time to measure the proximity between network locations rather than network distance, $D_N()$. The travel time between two network locations, $T_N()$, can be expressed as

$$T_N\left((l_u, m_i), (l_v, m_j)\right) = \min\left(T_{SP}\left((l_u, m_i), (l_v, m_j)\right), T_{SP}\left((l_v, m_j), (l_u, m_i)\right)\right) \tag{14}$$

where $T_{SP}\left((l_u, m_i), (l_v, m_j)\right)$ is the least travel time from (l_u, m_i) to (l_v, m_j), and can be calculated by shortest path algorithms using the link travel time attribute. The $T_N()$ metric also satisfies non-negativity, reflexivity, symmetry, and triangle inequality properties.

Based on either $D_N()$ or $T_N()$, space-time buffering can be used to analyze spatiotemporal proximity relationships of network-constrained space-time paths. These two network spatial distance metrics can be calculated using the same shortest path algorithms with different link attributes, i.e., link length for $D_N()$ and link travel time for $T_N()$. Another difference when using these metrics for constructing $STB^q(\omega)$ is the input spatial tolerance, ω: distance tolerance should be used for $D_N()$; while travel time tolerance should be used for $T_N()$. We only use the $D_N()$ metric hereafter to simplify the presentation.

Figure 7 shows the network space-time buffer concept for a network-constrained space-time path P^q in the (x, y, t) space. For any t_k, a network spatial buffer is constructed to delimit all network locations satisfying $D_N\left(P^q(t_k),(l_u,m_k)\right)\leq\omega,\ \forall(l_u,m_k,t_k)$. From Eq. (13), all network locations satisfying either $D_{SP}\left(P^q(t_k),(l_u,m_k)\right)\leq\omega$ or $D_{SP}\left((l_u,m_k),P^q(t_k)\right)\leq\omega$ should be included in the network spatial buffer. Network locations satisfying the former condition form a forward spatial buffer, denoted by $A_k^f(\omega)$, and can be determined by forward shortest path search using $P^q(t_k)$ as the origin node. In contrast, network locations satisfying the latter condition form a backward spatial buffer, denoted by $A_k^b(\omega)$, and can be determined by the backward shortest path search using $P^q(t_k)$ as the destination node.

Accordingly, the network space-time buffer $STB^q(\omega)$ can be generated by continuously constructing forward and backward spatial buffers, $A_k^f(\omega)$ and $A_k^b(\omega)$, respectively, for any $P^q(t_k)$ along the path. Figure 7 shows that all space-time locations covered by $A_k^f(\omega),\forall t_k$ form a forward space-time buffer (light orange), denoted by $STB^f(\omega)$. In contrast, all space-time locations covered by $A_k^b(\omega),\forall t_k$ form a backward space-time buffer (light blue), denoted by $STB^b(\omega)$. The generated $STB^q(\omega)=STB^f(\omega)\cup STB^b(\omega)$. As shown in the figure, the generated $STB^q(\omega)=<\ldots,O_u^q,\ldots>$ comprises a set of space-time polygons upon the links, where O_u^q represents a space-time polygon upon link a_u. Each space-time polygon O_u^q may be the union of the corresponding two parts, $O_u^f\in STB^f(\omega)$ and $O_u^b\in STB^b(\omega)$.

5.2. Geo-computational algorithm for constructing a network space-time buffer

Based on the concept of network space-time buffering, a straightforward algorithm is to continuously construct forward and backward spatial buffers, $A_k^f(\omega)$ and $A_k^b(\omega)$, for at any space-time location $P^q(t_k)$ along the path. However, such a straightforward algorithm is computationally intractable. In this section, we propose a geo-computational algorithm to efficiently construct $STB^q(\omega)$ for a network-constrained space-time path P^q.

The proposed algorithm makes use of the constant speed characteristic of every path segment, $s_{ij}^q\in P^q$. Hence the forward and backward spatial buffers for s_{ij}^q are only required at two control points, $P^q(t_i)$ and $P^q(t_j)$, to construct the corresponding forward and backward space-time buffers, $STB_{ij}^f(\omega)$ and $STB_{ij}^b(\omega)$. Thus, $STB^q(\omega)$ can be efficiently generated after $STB_{ij}^f(\omega)$ and $STB_{ij}^b(\omega)$ are constructed for any path segment $\forall s_{ij}^q\in P^q$.

We first introduce the method to construct the forward space-time buffer, $STB_{ij}^{f}(\omega)$, for a path segment $s_{ij}^{q} \in P^{q}$. Let $A_{i}^{f}(\omega)$ and $A_{j}^{f}(\omega)$ be the forward spatial buffers constructed at control points, $P^{q}(t_{i})$ and $P^{q}(t_{j})$, respectively. Suppose that the projection of s_{ij}^{q} is on link a_{u} (called segment link). The forward space-time buffer $STB_{ij}^{f}(\omega) =< O_{u,ij}^{f}, O_{v,ij}^{f}, \ldots >$ consists of forward space-time polygons upon a_{u} and other nearby links $\forall a_{v} \in A_{j}^{f}(\omega)$.

Figure 8 shows that $O_{u,ij}^{f}$ upon a_{u} has five possible cases using the LRS measures. Spatial tolerance ω is converted into the LRS measure,

$$m_{\omega} = \omega / d_{u} \tag{15}$$

where d_{u} is the length of a_{u}. Adding m_{ω} into the control points, $P^{q}(t_{i}) = (l_{u}, m_{i}, t_{i})$ and $P^{q}(t_{j}) = (l_{u}, m_{j}, t_{j})$, we have two space-time locations $c_{i}^{fu} = (l_{u}, m_{i} + m_{\omega}, t_{i})$ and $c_{j}^{fu} = (l_{u}, m_{j} + m_{\omega}, t_{j})$, which together with the two control points form a forward space-time parallelogram $< P^{q}(t_{i}), c_{i}^{fu}, c_{j}^{fu}, P^{q}(t_{j}) >$. Then, $O_{u,ij}^{f}$ can be generated by cutting the parallelogram outside the ending node of link a_{u}. From the relationship between the parallelogram and the segment link, five $O_{u,ij}^{f}$ cases can be identified as follows.

Case (fu.a): $m_{i} + m_{\omega} \leq m_{e}$ and $m_{j} + m_{\omega} \leq m_{e}$, i.e., the whole parallelogram, including c_{i}^{fu} and c_{j}^{fu}, is within link a_{u}. Thus, $O_{u,ij}^{f} =< P^{q}(t_{i}), c_{i}^{fu}, c_{j}^{fu}, P^{q}(t_{j}) >$, as shown in Figure 8(a).

Case (fu.b): $m_{i} + m_{\omega} \leq m_{e}$ and $m_{j} < m_{e} < m_{j} + m_{\omega}$, i.e., c_{i}^{fu} is within link a_{u} but c_{j}^{fu} is outside the link. Thus, $O_{u,ij}^{f} =< P^{q}(t_{i}), c_{i}^{fu}, c_{1}^{fu}, c_{2}^{fu}, P^{q}(t_{j}) >$, as shown in Figure 8(b), where $c_{1}^{fu} = (l_{u}, m_{e}, t_{1}^{fu} = t_{j} - \dfrac{m_{j} + m_{\omega} - m_{e}}{\tan\alpha})$ and $c_{2}^{fu} = (l_{u}, m_{e}, t_{j})$.

Case (fu.c): $m_{i} + m_{\omega} \leq m_{e}$ and $m_{j} = m_{e}$, i.e., c_{i}^{fu} is within link a_{u} and $P^{q}(t_{j})$ is at the ending node of link a_{u}. Figure 8(c) shows that $O_{u,ij}^{f} =< P^{q}(t_{i}), c_{i}^{fu}, c_{1}^{fu}, P^{q}(t_{j}) >$, where $c_{1}^{fu} = (l_{u}, m_{e}, t_{1}^{fu} = t_{j} - \dfrac{m_{\omega}}{\tan\alpha})$.

Case (fu.d): $m_{i} < m_{e} < m_{i} + m_{\omega}$ and $m_{j} < m_{e} < m_{j} + m_{\omega}$, i.e., both c_{i}^{fu} and c_{j}^{fu} are outside a_{u}. Figure 8(d) shows that $O_{u,ij}^{f} =< P^{q}(t_{i}), c_{1}^{fu}, c_{2}^{fu}, P^{q}(t_{j}) >$, where $c_{1}^{fu} = (l_{u}, m_{e}, t_{i})$ and $c_{2}^{fu} = (l_{u}, m_{e}, t_{j})$.

Case (fu.e): $m_{i} < m_{e} < m_{i} + m_{\omega}$ and $m_{j} = m_{e}$, i.e., both c_{i}^{fu} and c_{j}^{fu} are outside a_{u} and $P^{q}(t_{j})$ is at the ending node of link a_{u}. Figure 8e shows that $O_{u,ij}^{f} =< P^{q}(t_{i}), c_{1}^{fu}, P^{q}(t_{j}) >$, where $c_{1}^{fu} = (l_{u}, m_{e}, t_{i})$.

In above figures, the angle α can be expressed as

$$\tan\alpha = \frac{m_{j} - m_{i}}{t_{j} - t_{i}} = \frac{v_{ij}^{q}}{d_{u}} \tag{16}$$

The forward spatial buffers, $A_{i}^{f}(\omega)$ and $A_{j}^{f}(\omega)$, could cover not only a_{u}, but also other nearby links. Let $\overline{A}_i^f(\omega) = A_i^f(\omega) - a_u$ and $\overline{A}_j^f(\omega) = A_j^f(\omega) - a_u$ be the set of nearby links covered by forward spatial buffers, excluding the segment link a_u. Then the following relationship between $\overline{A}_i^f(\omega)$ and $\overline{A}_j^f(\omega)$ holds.

Lemma 1. $\overline{A}_j^f(\omega) = \overline{A}_i^f(\omega + m_j d_u - m_i d_u)$.
Proof. See Lemma A1 in the appendix.

Lemma 2. $\overline{A}_i^f(\omega) \subset \overline{A}_j^f(\omega)$.
Proof. See Lemma A2 in the appendix.

According to Lemma 2, the forward space-time buffer $STB_{ij}^f(\omega)$ comprises forward space-time polygons for all nearby links $\forall a_v \in \overline{A}_j^f(\omega)$. Suppose that sub-link (l_v, m_s) to (l_v, m_i^{fv}) is covered by $\overline{A}_i^f(\omega)$; and sub-link (l_v, m_s) to (l_v, m_j^{fv}) is covered by $\overline{A}_j^f(\omega)$. Then, their corresponding space-time locations are $c_{i,s}^{fv} = (l_v, m_s, t_i)$, $c_i^{fv} = (l_v, m_i^{fv}, t_i)$, $c_{j,s}^{fv} = (l_v, m_s, t_j)$ and $c_j^{fv} = (l_v, m_j^{fv}, t_j)$. For each nearby link $a_v \in \overline{A}_j^f(\omega)$, the forward space-time polygon $O_{v,ij}^f$ can be determined as one of the following four cases, as shown in Figure 9.

Case (fv.a): $m_i^{fv} = m_e$ and $m_j^{fv} = m_e$, i.e., the whole of link a_v is covered by both $\overline{A}_i^f(\omega)$ and $\overline{A}_j^f(\omega)$. Figure 9(a) shows that $O_{v,ij}^f = < c_{i,s}^{fv}, c_i^{fv}, c_j^{fv}, c_{j,s}^{fv} >$.

Case (fv.b): $m_s < m_i^{fv} < m_e$ and $m_j^{fv} = m_e$, i.e., link a_v is partially covered by $\overline{A}_i^f(\omega)$ and fully covered by $\overline{A}_j^f(\omega)$. Figure 9(b) shows that $O_{v,ij}^f = < c_{i,s}^{fv}, c_i^{fv}, c_1^{fv}, c_j^{fv}, c_{j,s}^{fv} >$, where $c_1^{fv} = (l_v, m_e, t_1^{fv} = t_i + \dfrac{m_e - m_i^{fv}}{\tan \overline{\alpha}})$.

Case (fv.c): $m_s < m_i^{fv} < m_e$ and $m_s < m_j^{fv} < m_e$, i.e., link a_v is partially covered by both $\overline{A}_i^f(\omega)$ and $\overline{A}_j^f(\omega)$. Figure 9(c) shows that $O_{v,ij}^f = < c_{i,s}^{fv}, c_i^{fv}, c_j^{fv}, c_{j,s}^{fv} >$.

Case (fv.d): $m_j^{fv} \le m_e$, i.e., link a_v is partially covered by $\overline{A}_j^f(\omega)$, but outside $\overline{A}_i^f(\omega)$. Figure 9(d) shows that $O_{v,ij}^f = < c_1^{fv}, c_j^{fv}, c_{j,s}^{fv} >$, where $c_1^{fv} = (l_v, m_e, t_1^{fv} = t_j - \dfrac{m_j^{fv} - m_s}{\tan \overline{\alpha}})$.

In the above figures, the angle $\overline{\alpha}$ can be expressed as

$$\tan \overline{\alpha} = \frac{v_{ij}^q}{d_v} \tag{17}$$

We now introduce the method to construct the backward space-time buffer, $STB_{ij}^b(\omega)$, for each path segment s_{ij}^q. Let $A_i^b(\omega)$ and $A_j^b(\omega)$ be the backward spatial buffers constructed at control points $P^q(t_i)$ and $P^q(t_j)$, respectively. Then, $STB_{ij}^b(\omega) = < O_{u,ij}^b, O_{v,ij}^b, \ldots >$, comprises backward space-time polygons upon link a_u and other nearby links, $\forall a_v \in A_i^b(\omega)$.

Similar to the forward buffer case, the backward space-time polygon $O_{u,ij}^b$ upon segment link a_u also has five possible cases, as shown in Figure 10. However, the space-time locations

$c_i^{bu} = (l_u, m_i - m_\omega, t_i)$ and $c_j^{bu} = (l_u, m_j - m_\omega, t_j)$ are determined by subtracting $m_\omega = \omega / d_u$ from control points, $P^q(t_i)$ and $P^q(t_j)$, respectively. Thus, $O_{u,ij}^b$ can be generated by cutting the backward space-time parallelogram $< c_i^{bu}, P^q(t_i), P^q(t_j), c_j^{bu} >$ outside the starting node of link a_u. The five $O_{u,ij}^b$ cases are as follows.

Case (bu.a): $m_s \le m_i - m_\omega$ and $m_s \le m_j - m_\omega$, i.e., the whole backward parallelogram, including both c_i^{bu} and c_j^{bu}, is within a_u. Figure 10(a) shows that $O_{u,ij}^b =< c_i^{bu}, P^q(t_i), P^q(t_j), c_j^{bu} >$.

Case (bu.b): $m_i - m_\omega < m_s < m_i$ and $m_s \le m_j - m_\omega$, i.e., c_j^{bu} is within link a_u but c_i^{bu} is outside the link. Figure 10(b) shows that $O_{u,ij}^b =< c_1^{bu}, P^q(t_i), P^q(t_j), c_j^{bu}, c_2^{bu} >$, where $c_1^{bu} = (l_u, m_s, t_i)$ and $c_2^{bu} = (l_u, m_s, t_2^{bu} = t_i + \dfrac{m_s - m_i + m_\omega}{\tan\alpha})$.

Case (bu.c): $m_i = m_s$ and $m_s \le m_j - m_\omega$, i.e., c_j^{bu} is within link a_u and $P^q(t_i)$ is at the starting node of link a_u. Figure 10(c) shows that $O_{u,ij}^b =< P^q(t_i), P^q(t_j), c_j^{bu}, c_2^{bu} >$, where $c_2^{bu} = (l_u, m_s, t_2^{bu} = t_i + \dfrac{m_\omega}{\tan\alpha})$.

Case (bu.d): $m_i - m_\omega < m_s < m_i$ and $m_j - m_\omega < m_s < m_j$, i.e., both c_i^{bu} and c_j^{bu} are outside link a_u. Figure 10(d) shows that $O_{u,ij}^b =< c_1^{bu}, P^q(t_i), P^q(t_j), c_2^{bu} >$, where $c_1^{bu} = (l_u, m_s, t_i)$ and $c_2^{bu} = (l_u, m_s, t_j)$.

Case (bu.e): $m_i = m_s$ and $m_j - m_\omega < m_s < m_j$, i.e., both c_i^{bu} and c_j^{bu} are outside link a_u and $P^q(t_i)$ is at the starting node of link a_u. Figure 10e shows that $O_{u,ij}^b =< P^q(t_i), P^q(t_j), c_2^{bu} >$, where $c_2^{bu} = (l_u, m_s, t_j)$.

The backward space-time polygons for all nearby links can now be determined. Let $\overline{A}_i^b(\omega) = A_i^b(\omega) - a_u$ and $\overline{A}_j^b(\omega) = A_j^b(\omega) - a_u$ be the set of nearby links covered by backward spatial buffers, excluding the segment link a_u. Then, the following relationships between $\overline{A}_i^b(\omega)$ and $\overline{A}_j^b(\omega)$ hold.

Lemma 3. $\overline{A}_i^b(\omega) = \overline{A}_j^b(\omega + m_j d_u - m_i d_u)$.
Proof. See Lemma A3 in the appendix. □

Lemma 4. $\overline{A}_j^b(\omega) \subset \overline{A}_i^b(\omega)$.
Proof. See Lemma A4 in the appendix.

According to Lemma 4, $STB_{ij}^b(\omega)$ comprises backward space-time polygons for all nearby links $\forall a_v \in \overline{A}_i^b(\omega)$. Suppose that sub-link (l_v, m_i^{bv}) to (l_v, m_e) is covered by $\overline{A}_i^b(\omega)$; and sub-link (l_v, m_j^{bv}) to (l_v, m_e) is covered by $\overline{A}_j^b(\omega)$, with corresponding space-time locations $c_i^{bv} = (l_v, m_i^{bv}, t_i)$, $c_{i,e}^{bv} = (l_v, m_e, t_i)$, $c_j^{bv} = (l_v, m_j^{bv}, t_j)$, and $c_{j,e}^{bv} = (l_v, m_s, t_j)$. Then, similar to the forward case, the backward space-time polygon $O_{v,ij}^b$ for each nearby link $a_v \in \overline{A}_i^b(\omega)$ is one of the following four as shown in Figure 11.

Case (bv.a): $m_i^{bv} = m_s$ and $m_j^{bv} = m_s$, i.e., the whole of link a_v is covered by both $\overline{A}_i^b(\omega)$ and $\overline{A}_j^b(\omega)$. Figure 11(a) shows that $O_{v,ij}^b = < c_i^{bv}, c_{i,e}^{bv}, c_{j,e}^{bv}, c_j^{bv} >$.

Case (bv.b): $m_i^{bv} = m_s$ and $m_s < m_j^{bv} < m_e$, i.e., link a_v is fully covered by $\overline{A}_i^b(\omega)$ and partially covered by $\overline{A}_j^b(\omega)$. Figure 11(b) shows that $O_{v,ij}^b = < c_i^{bv}, c_{i,e}^{bv}, c_{j,e}^{bv}, c_j^{bv}, c_1^{bv} >$, where $c_1^{bv} = (l_v, m_s, t_1^{bv} = t_j - \frac{m_j^{bv} - m_s}{\tan\overline{\alpha}})$.

Case (bv.c): $m_s < m_i^{bv} < m_e$ and $m_s < m_j^{bv} < m_e$, i.e., link a_v is partially covered by both $\overline{A}_i^b(\omega)$ and $\overline{A}_j^b(\omega)$. Figure 11(c) shows that $O_{v,ij}^b = < c_i^{bv}, c_{i,e}^{bv}, c_{j,e}^{bv}, c_j^{bv} >$.

Case (bv.d): $m_s < m_i^{bv} < m_e$, i.e., link a_v is partially covered by $\overline{A}_i^b(\omega)$ but outside $\overline{A}_j^b(\omega)$. Figure 11(d) shows that $O_{v,ij}^b = < c_i^{bv}, c_{i,e}^{bv}, c_1^{bv} >$, where $c_1^{bv} = (l_v, m_e, t_1^{bv} = t_i + \frac{m_e - m_i^{bv}}{\tan\overline{\alpha}})$.

After both $STB_{ij}^f(\omega) = < ..., O_{u,ij}^f, ... >$ and $STB_{ij}^b(\omega) = < ..., O_{u,ij}^b, ... >$ are constructed for path segment s_{ij}^q, the space-time buffer can be determined as $STB_{ij}^q(\omega) = < ..., O_{u,ij}^q = O_{u,ij}^f \cup O_{u,ij}^b, ... >$. Consequently, once $STB_{ij}^q(\omega)$ is constructed for all path segments $\forall s_{ij}^q \in P^q$, space-time buffer $STB^q(\omega)$ for space-time path P^q can be generated as the union of all $STB_{ij}^q(\omega)$. The detailed steps of the proposed geo-computational algorithm are as follows.

Geo-computational algorithm for constructing a network space-time buffer

Inputs: Space-time path $P^q = < \mathrm{L}, R_u^q, \mathrm{L} >$, spatial tolerance ω.

Return: Space-time buffer $STB^q(\omega) = < ..., O_u^q, ... >$.

```
01: For each $R_u^q$ in $P^q$
02:   Call ForwardNetBuffer($R_u^q, \omega$) to construct $\overline{A}_i^f(\omega)$ for all control points $\forall P^q(t_i) \in R_u^q$.
03:   Call BackwardNetBuffer($R_u^q, \omega$) to construct $\overline{A}_i^b(\omega)$ for all control points $\forall P^q(t_i) \in R_u^q$.
04:   For each $s_{ij}^q$ in $R_u^q$
05:      Generate $O_{u,ij}^f$ based on Cases (fu.a–fu.e), and set $O_u^q = O_u^q \cup O_{u,ij}^f$.
06:      For each nearby link $a_v$ in $\overline{A}_j^f(\omega)$
07:         Generate $O_{v,ij}^f$ based on Cases (fv.a–fv.d), and set $O_v^q = O_v^q \cup O_{v,ij}^f$.
08:      End For
09:      Generate $O_{u,ij}^b$ based on Cases (bu.a–bu.e), and set $O_u^q = O_u^q \cup O_{u,ij}^b$.
10:      For each nearby link $a_v$ in $\overline{A}_i^b(\omega)$
11:         Generate $O_{v,ij}^b$ based on Cases (bv.a–bv.d), and set $O_v^q = O_v^q \cup O_{v,ij}^b$.
12:      End For
13:   End For
14: End For
15: Return space-time buffer $STB^q(\omega)$ with all generated space-time polygons $\forall O_u^q$.
```

In the proposed algorithm, the *ForwardNetBuffer* procedure efficiently constructs forward spatial buffers $\overline{A}_i^f(\omega)$ for all control points $\forall P^q(t_i) \in R_u^q$, using a single forward shortest path search. Intuitively, to obtain $\overline{A}_i^f(\omega)$ for every control point $P^q(t_i)$, one forward link based shortest path search (Chen et al., 2016b; Li et al., 2015) is performed to determine all network links with network distance to c_i^u less than ω. Consequently, shortest path searches are repeatedly performed to obtain forward spatial buffers for all control points upon the same link a_u. However, this repeated shortest path search process introduces significant computational overhead, because $\overline{A}_j^f(\omega)$ of a control point already contains $\overline{A}_i^f(\omega)$ of the previous control point (i.e., $\overline{A}_i^f(\omega) \subset \overline{A}_j^f(\omega)$ from lemma 2).

In view of this, we introduce an alternative incremental shortest path search technique in *ForwardNetBuffer*. The procedure uses Lemma 1 to convert $\overline{A}_i^f(\omega)$ of control point $P^q(t_i)$ to $\overline{A}_s^f(\omega + m_i d_u - m_s d_u)$ of the link starting point. Thus, $\overline{A}_i^f(\omega)$ of all control points can be constructed using the forward shortest path search originating from the same link starting point. The procedure first constructs $\overline{A}_s^f(\omega)$ for the starting node with spatial tolerance ω. Then, to construct $\overline{A}_k^f(\omega) = \overline{A}_s^f(\omega + m_k d_u - m_s d_u)$ for the second control point, $P^q(t_k)$, the procedure continues the previous shortest path search with increased spatial tolerance, $\omega + m_k d_u - m_s d_u$. The procedure continues to construct $\overline{A}_i^f(\omega) = \overline{A}_s^f(\omega + m_i d_u - m_s d_u)$ for every control point, until $\overline{A}_e^f(\omega) = \overline{A}_s^f(\omega + m_e d_u - m_s d_u)$ for the link ending node. Therefore, the proposed *ForwardNetBuffer* procedure efficiently determines $\overline{A}_i^f(\omega)$ for all control points $\forall P^q(t_i) \in R_u^q$ using a single forward shortest path search. Compared to the repeated search process, this incremental search process only explicitly constructs $\overline{A}_e^f(\omega)$ of the link ending node, and avoids the computational effort required in constructing $\overline{A}_i^f(\omega)$ for all other control points.

The incremental shortest path search technique is also adopted for the *BackwardNetBuffer* procedure to efficiently construct a backward spatial buffer $\overline{A}_i^b(\omega)$ for all control points, $\forall P^q(t_i) \in R_u^q$, using a single backward search process. Similar to the forward procedure, the *BackwardNetBuffer* procedure converts $\overline{A}_i^b(\omega)$ of every control point $P^q(t_i)$ to $\overline{A}_e^b(\omega + m_e d_u - m_i d_u)$ of the link-ending node using Lemma 3. Consequently, $\overline{A}_i^b(\omega)$ for all control points can be constructed by a single backward shortest path search rooted at the link-ending node. The procedure first performs the backward shortest path search with spatial tolerance ω to construct $\overline{A}_e^b(\omega)$ of the link-ending point, then continues the path search with increased spatial tolerance $\omega + m_e d_u - m_i d_u$ to construct $\overline{A}_i^b(\omega)$ for every control point; and stops the search after constructing $\overline{A}_s^b(\omega)$ for the link-starting point. Therefore, the procedure efficiently determines $\overline{A}_i^b(\omega)$ for all control points in a single backward shortest path search.

The computational complexity of the proposed geo-computational algorithm is analyzed. In the worst case, both *ForwardNetBuffer* and *BackwardNetBuffer* procedures run in $O(|\Psi| + |A| Log |A|)$, where $|\Psi|$ is the number of allowed turns in the road network and $|A|$ is the number of links in the road network. Therefore, the proposed geo-computational algorithm runs in

$O(|R_u^q||\Psi|+|R_u^q||A|Log|A|)$, where $|R_u^q|$ is the number of links covered by the space-time path, P^q. Thus, the algorithm's computational performance strongly depends on *ForwardNetBuffer* and *BackwardNetBuffer* efficiency.

To further improve computational efficiency, an avoiding label initialization technique (Chen et al., 2014) is adopted. In conventional shortest path searching algorithms (Li et al., 2015), the label initialization step is required to set "null" labels to all network links before the search process. Since the network spatial buffer constructed at each control point is relatively small, the shortest path algorithm only explores a small number of links. Thus, label initialization for all network links can cause significant computational overheads in the construction of network space-time buffers, where numerous shortest path searches are performed at all control points. The avoiding label initialization technique (Chen et al., 2014) assigns a unique search ID to each shortest path search. Labels generated by previous shortest path searches are identified by comparing their associated search ID, and unnecessary label initialization steps can be avoided.

6. Implementation of the space-time integrated approach in movement database

The proposed space-time integrated approach is implemented in an effective spatiotemporal data model for representing and computing network-constrained time geographic entities and relationships (Chen et al., 2016a). This spatiotemporal data model builds on the CLR technique to transform 3D network-constrained time geographic entities in (x,y,t) space to 2D entities in CLR space. Subsequently, efficient and effective spatial databases and indexes can be directly employed to store, index, and query network-constrained time geographical entities and relationships in CLR space. Therefore, we employed a CLR spatiotemporal data model to implement the proposed space-time buffering and overlapping operations.

The CLR technique proposed by Chen et al. (2016a) is briefly introduced. The linear reference (l_u,m_i) is an equivalent one-to-one representation of geographical location (x_i,y_i) in the road network. Since m_i, is a real number between 0 and 1 and link ID, l_u, is a unique positive integer number, they can be integrated into a single real value

$$z_i^u = l_u + m_i \tag{18}$$

The z_i^u value is also an equivalent representation of (x_i,y_i) and can be easily transformed to the geographical location using dynamic segmentation. Therefore, any 3D point $c_i^u=(x_i,y_i,t_i)=(l_u,m_i,t_i)$ in the road network can be represented as a unique 2D point $\tilde{c}_i^u=(z_i,t_i)$ in (z,t) space, which is called the CLR space.

Using the CLR technique, a 3D network-constraint space-time path P^q in the (x,y,t) space can be transformed into the CLR space as a 2D entity. Figure 12 shows the space-time path in CLR space transformed from that in Figure 7. The transformed space-time path in CLR space, $\tilde{P}^q$, comprises a set of disjoint LineString elements (i.e., polylines) $<\ldots,\tilde{R}_u^q,\ldots>$, where each LineString $\tilde{R}_u^q$ corresponds to R_u^q in (x,y,t) space, representing the individual's continuous movement on a specific link a_u. LineString $\tilde{R}_u^q$ is further represented by a set of path segments, $<\ldots,\tilde{s}_{ij}^q,\ldots>$, where each $\tilde{s}_{ij}^q$ is a straight line in CLR space connecting two consecutive control points $\tilde{c}_i^u=(z_i,t_i)$ and $\tilde{c}_j^u=(z_j,t_j)$, on the same link, a_u, which corresponds to s_{ij} in (x,y,t) space. Chen et al. (2016a) proved that $\tilde{P}^q$ in CLR space is an equivalent representation of P^q in (x,y,t) space as follows.

Proposition 7. The network constrained space-time path $\tilde{P}^q$ in CLR space is an equivalent representation of P^q in (x, y, t) space.
Proof. See Proposition 1 in Chen et al., (2016a). □

Following the previous work, we transform the network space-time buffer, $STB^q(\omega)$, into CLR space by converting any space-time location $c_i^u = (l_u, m_i, t_i)$ to $\tilde{c}_i = (z_i, t_i)$. In practice, any $STB^q(\omega)$ can be transformed by converting each $STB_{ij}^q(\omega)$ of $s_{ij}^q \in P^q$, into CLR space as follows.

From the definition, $STB_{ij}^q(\omega)$ is the union of $STB_{ij}^f(\omega)$ and $STB_{ij}^b(\omega)$ for a path segment s_{ij}^q upon link a_u. Firstly, $STB_{ij}^f(\omega) =< O_{u,ij}^f, O_{v,ij}^f, ... >$, is transformed into CLR space. Any $O_{u,ij}^f =< ..., c_k^u, ... >$ case fu.a–fu.e (Figure 8) is transformed into a unique polygon $\tilde{O}_{u,ij}^f =< ..., \tilde{c}_k, ... >$ in CLR space by converting any point $c_k^u = (l_u, m_k, t_k)$ to $\tilde{c}_k = (z_k^u = l_u + m_k, t_k)$, and any $O_{v,ij}^f =< ..., c_k^v, ... >$ case fv.a–fv.d (Figure 9) is also transformed into a unique polygon $\tilde{O}_{v,ij}^f =< ..., \tilde{c}_k, ... >$ in CLR space by converting any point $c_k^v = (l_v, m_k, t_k)$ to $\tilde{c}_k = (z_k^v = l_v + m_k, t_k)$. Thus, $STB_{ij}^f(\omega) =< O_{u,ij}^f, O_{v,ij}^f, ... >$ can be transformed into a unique $\widetilde{STB}_{ij}^f(\omega) =< \tilde{O}_{u,ij}^f, \tilde{O}_{v,ij}^f, ... >$ in CLR space.

Secondly, $STB_{ij}^b(\omega) =< O_{u,ij}^b, O_{v,ij}^b, ... >$ is transformed into CLR space using a similar approach. Any $O_{u,ij}^b =< ..., c_k^u, ... >$ case bu.a–bu.e (Figure 10) is transformed into a unique polygon $\tilde{O}_{u,ij}^b =< ..., \tilde{c}_k, ... >$ in CLR space by converting any point $c_k^u = (l_u, m_k, t_k)$ to $\tilde{c}_k = (z_k^u = l_u + m_k, t_k)$, and any $O_{v,ij}^b =< ..., c_k^v, ... >$ case bv.a–bv.d (Figure 11) is transformed into a unique polygon $\tilde{O}_{v,ij}^b =< ..., \tilde{c}_k, ... >$ in CLR space by converting any point $c_k^v = (l_v, m_k, t_k)$ to $\tilde{c}_k = (z_k^v = l_v + m_k, t_k)$. Thus, $STB_{ij}^b(\omega) =< O_{u,ij}^b, O_{v,ij}^b, ... >$ can be transformed into a unique $\widetilde{STB}_{ij}^b(\omega) =< \tilde{O}_{u,ij}^b, \tilde{O}_{v,ij}^b, ... >$ in CLR space.

Consequently, $STB_{ij}^q(\omega) = STB_{ij}^f(\omega) \cup STB_{ij}^b(\omega)$, can be transformed into a unique space-time buffer, $\widetilde{STB}_{ij}^q(\omega) = \widetilde{STB}_{ij}^f(\omega) \cup \widetilde{STB}_{ij}^b(\omega)$, in CLR space. After $\widetilde{STB}_{ij}^q(\omega)$ for $\forall s_{ij}^q \in P^q$ are determined, the network space-time buffer in CLR space, $\widetilde{STB}^q(\omega)$, can be obtained as the union of all determined $\widetilde{STB}_{ij}^q(\omega)$. Figure 12 shows a simple example of $\widetilde{STB}^q(\omega)$ transformed from the $STB^q(\omega)$ in Figure 7. $\widetilde{STB}^q(\omega)$ is a 2D entity in CLR space, comprising a set of disjoint 2D polygons $< ..., \tilde{O}_u^q, ... >$ in CLR space, where each $\tilde{O}_u^q$ corresponds to $O_u^q \in STB^q(\omega)$ in (x, y, t) space. We prove that $\widetilde{STB}^q(\omega)$ is an equivalent one-to-one representation of $STB^q(\omega)$ as follows.

Proposition 8. The network space-time buffer $\widetilde{STB}^q(\omega)$ in CLR space is an equivalent

representation of $STB^q(\omega)$ in (x, y, t) space.
Proof. See Proposition A7 in the appendix. □

To construct $\widetilde{STB}^q(\omega)$ for a some $\tilde{P}^q$ in CLR space, the proposed geo-computational algorithm can be modified by simply replacing $O^f_{u,ij}$, $O^f_{v,ij}$, $O^b_{u,ij}$ and $O^b_{v,ij}$ in (x, y, t) space with $\tilde{O}^f_{u,ij}$, $\tilde{O}^f_{v,ij}$, $\tilde{O}^b_{u,ij}$ and $\tilde{O}^b_{v,ij}$ in CLR space, with the same computational complexity.

In CLR space, the network distance between two network locations, $z_i = (l_u, m_i)$ and $z_j = (l_v, m_j)$, is calculated using the same network proximity operator as in (x, y, t) space, i.e., $D_N\left(z_i, z_j\right) = D_N\left((l_u, m_i), (l_v, m_j)\right)$. We show that the spatiotemporal proximity relationships defined in (x, y, t) space remain valid for any two space-time paths, $\tilde{P}^q$ and $\tilde{P}^b$, in CLR space, as follows.

Proposition 9. Spatiotemporal proximity relationships (i.e., $\delta_{\text{Intersect}}$, δ_{Bundle}, M_{CPD}, M_{FD}, M_{APD} and M_{LCSS}) between $\tilde{P}^q$ and $\tilde{P}^b$ in CLR space are equal to those between P^q and P^b in (x, y, t) space.
Proof. See Proposition A8 in the appendix. □

We also shown that $\widetilde{STB}^q(\omega)$ can delimit all space-time points $\tilde{p}_k = (z^v_k, t_k)$ in CLR space where $D_N\left(\tilde{p}_k, \tilde{P}^q(t_k)\right)$ is less than the given spatial tolerance ω as follows.

Proposition 10. The network space-time buffer $\widetilde{STB}^q(\omega)$ of space-time path $\tilde{P}^q$ delimits all possible space-time points $\tilde{p}_k = (z^v_k, t_k)$ in CLR space satisfying $D_N\left(\tilde{p}_k, \tilde{P}^q(t_k)\right) \le \omega, t^q_s \le \forall t_k \le t^q_e$.
Proof. See Proposition A9 in the appendix. □

Based on the above Propositions 7–10, the proposed space-time buffering and overlapping can be easily implemented in CLR space for spatiotemporal proximity analysis of movement data. From Proposition 7, all space-time paths can be converted into CLR space, so they can be stored, indexed, and queried as 2D polyline entities in the existing spatial database. From Proposition 8, given a target space-time path $\tilde{P}^q$ in CLR space, $\widetilde{STB}^q(\omega)$, can be constructed and represented as a 2D multi-polygon in CLR space. Consequently, efficient 2D spatial queries (i.e., polygon-polyline-intersection and polygon-polyline-contain queries) provided by the spatial database can be directly employed to implement complicated 3D spatiotemporal queries (i.e., buffer-path-intersection and buffer-path-contain queries, respectively). Propositions 9 and 10 show how these polygon-polyline-intersection (or polygon-polyline-contain) queries can retrieve all space-time paths $\forall \tilde{P}^b$ intersected with (or contained by) $\widetilde{STB}^q(\omega)$. Therefore, the proposed space-time integrated approach can be efficiently implemented in spatial databases using the CLR technique.

7. Case study

This section presents a case study using a real-world movement dataset to demonstrate the applicability of the proposed space-time integrated approach. We adopted ArcSDE 10.2 and Oracle 11g as the spatial database to store, index, and query space-time paths in CLR space, with Grid-file employed to index space-time paths. The proposed geo-computational algorithm to construct network space-time buffers in CLR space was coded in Visual C#, with the F-heap data structure (Fredman and Tarjan, 1987) as the priority queue. The proposed five space-time overlapping operations were implemented using the ArcEngine 10.2 development kit in Visual C#. All experiments were conducted on a desktop computer with Intel dual core 3.1 GHz CPU (only a single core was used), 8 GB RAM, running Windows 7, 64-bit operating system.

7.1. Data collection

Two real-world datasets from Wuhan, a mega city in China, were collected. Figure 13 summarizes the Wuhan road network, which consists of 19,306 nodes and 46,757 directed links. Network link identities were set as unique integers from 1 to 46,757. The road network provides the CLR space reference framework for converting locations in (x, y, t) space to CLR space, and vice versa. The Wuhan floating car data was collected during 8–9 a.m. (i.e., morning peak) on a typical Thursday (3 September, 2009), and consisted of GPS observations of 10,000 taxis with (low) 30 s sampling frequency.

Due to GPS positioning and digital road network errors, GPS observations of taxis may not lie exactly on network links, and a taxi may have passed through several network links between consecutive observations due to the relatively low sampling frequency. Therefore, a multicriteria dynamic programming map matching (MDP-MM) algorithm (Chen et al., 2014) was employed to accurately map GPS observations onto the road network and reconstruct taxi space-time paths. All space-time paths were subsequently converted into CLR space and stored as a 2D polyline layer in the spatial database.

Three spatiotemporal proximity metrics, M_{CPD}, M_{APD} and M_{FD}, were evaluated and compared in both network and planar spaces. Figure 14 compares the calculated metrics between a random selected space-time path and all other paths in the dataset. To calculate these metrics, space-time paths were discretized into a set of space-time points every 30 s to ensure space-time points of all paths were sampled at the same time instances. The low sampling frequency (30 s) was adopted to reduce intensive computational cost for processing this large scale dataset.

As shown in the figure, the $M_{CPD} < M_{APD} < M_{FD}$ relationship held for both network and planar spaces. For example, the average M_{CPD} value in network space was 10.5 km, average $M_{APD} = 11.2$ km, and average $M_{FD} = 11.9$ km. However, distinct spatiotemporal proximity patterns were observed for the metrics in network space and their counterparts in planar space. For example, average M_{CPD} in planar space = 5.9 km, only 56.2% of the network space value. This highlights significant underestimation of spatiotemporal proximity between network-constrained space-time paths using such metrics in the planar space. Therefore, spatiotemporal proximity analyses should be performed for network constrained space-time paths in network space rather than planar space.

7.2. Computational performance of space-time buffering and overlapping operations

The computational performance of space-time buffering and overlapping operations were investigated. Table 1 shows computational times required by the proposed geo-computational algorithm to construct the network space-time buffer using different parameter settings in terms of spatial tolerance and target path length. Target path length was defined as the number of links covered by the path; and average link length in the Wuhan road network is approximately 269 meters. The reported computational times were the average of 10 runs of random selected target space-time paths.

Computational performance of the proposed algorithm degrades with increasing target space-time path length. For example, the proposed algorithm consumed 7 ms when $\omega = 500$ m and the target path covered 100 links (approximately 27 km), which increased 4.1 times to 36 ms when target path length increased to 300 links (approximately 81 km). This is because the number of shortest path searches performed by the proposed algorithm (i.e., ForwardNetBuffer and BackwardNetBuffer procedures) linearly increases with the length of the target path. The proposed algorithm computational performance also degrades with increasing spatial tolerance, ω. This is expected, since larger spatial tolerance increases the network space the proposed algorithm explores for every shortest path search. However, as shown in the table, the proposed algorithm can efficiently construct space-time buffers under various input spatial tolerances and target path lengths, in the large scale road network with satisfactory computational time (i.e., less than 40 ms).

The efficiency was a consequence of employing the incremental shortest path search and avoiding label initialization techniques. To further examine the proposed techniques' effectiveness, two algorithms were implemented. Algorithm 1 utilized only the avoiding label initialization technique, while Algorithm 2 did not use either techniques. Table 1 shows that the avoiding label initialization technique reduced computational times by 95.8% (i.e., 1 – 62 / 1478), when the target path covered 300 links and $\omega = 500$ m. This computational improvement for the avoiding label initialization technique shows the considerable computational cost of the label initialization step for numerous shortest path searches at all control points. Computational performances of the proposed algorithm and Algorithm 1 verify the effectiveness of the incremental shortest path search technique, providing 41.9% (i.e., 1 – 9 / 62) reduced computational times for the same target path length and spatial tolerance. This is expected, since the proposed incremental shortest path search technique determines forward (backward) network spatial buffers for all control points in one single forward (backward) shortest path search process, improving computational performance for constructing the network space-time buffer. The sampling frequency was relatively low for this case study (i.e., approximately 30 s), and the effectiveness of the incremental shortest path search technique would be further enhanced for target space-time paths with higher sampling frequency.

Table 1. Computational performance for constructing network space-time buffers

Spatial tolerance (m)	Target path length (number of links)	Proposed algorithm (ms)	Algorithm 1 (ms)	Algorithm 2 (ms)
100	100	3	5	473
200	100	4	6	474
300	100	5	8	476
400	100	6	11	479
500	100	7	13	485
500	150	12	20	713
500	200	21	32	968
500	250	27	46	1,197
500	300	36	62	1,478

Table 2 compares computational performance of the five space-time overlapping operations using the same target space-time path length and spatial tolerance settings. The reported computational times were also the average of 10 runs using randomly selected target space-time paths. All overlapping operations performed well, within 60 s, for retrieving spatiotemporal proximal space-time paths in the large movement dataset of 10,000 space-time paths, even when the target path covered 300 links (i.e., about 81 km). Computational performance of all overlapping operations degraded with increasing target space-time path lengths and spatial tolerances. For example, the CPD overlapping operation consumed 0.75 s, when $\omega = 100$ m and target path length = 100 links, which increased 6.7 times to 5.77 s when $\omega = 500$ m, and increased further 6.8 times to 45.14 s when target path length was set as 300 links. This is expected, since network space-time buffer sizes increased with increasing target space-time path lengths and spatial tolerances.

Table 2. Computational performance of five space-time overlapping operations

Spatial tolerance (m)	Target path length (number of links)	CPD overlapping (s)	FD overlapping (s)	APD overlapping (s)	LCSS overlapping (s)	Bundling (s)
100	100	0.752	0.763	0.784	0.808	0.818
200	100	0.967	0.997	1.020	1.069	1.081
300	100	1.338	1.392	1.445	1.534	1.596
400	100	3.913	4.053	4.115	4.213	4.289
500	100	5.771	6.010	6.121	6.268	6.381
500	150	15.836	16.531	18.818	19.189	19.330
500	200	21.292	22.383	24.858	25.245	25.517
500	250	30.628	31.684	34.527	35.343	35.692
500	300	45.140	47.532	56.099	57.856	58.279

7.3. Comparison with current spatiotemporal proximity analysis approaches

To demonstrate the effectiveness of the proposed space-time integrated approach, two conventional approaches, space-time separated and brute force approaches, were also implemented for comparison. Only the M_{CPD} measure was adopted to evaluate the spatiotemporal proximity between space-time paths (i.e., only the CPD operation), since the space-time overlapping operations have similar computational performance (see Table 2). The brute-force approach was implemented by calculating M_{CPD} between the target path and every other space-time path in the dataset, and by determining resultant space-time proximal paths with $M_{CPD} \leq \omega$. The space-time separated approach was implemented using the following two steps. Step 1 constructed a conventional spatial buffer using ω spatial tolerance along the target space-time path in planar space, then performed a spatial overlapping operation to retrieve all paths within the constructed spatial buffer, and checked their temporal coverage to exclude paths outside the time period of interest. Step 2 calculated M_{CPD} for the paths obtained in the first step, and paths violating $M_{CPD} \leq \omega$ were removed.

For these conventional approaches, space-time paths were stored as spatial polyline features in the spatial database and the time dimension information was stored as Z values along the polyline features. To approximate M_{CPD} metrics, all space-time paths were discretized into a set of space-time points every 30 s, to reduce the computational burden. In contrast, the proposed space-time integrated approach can exactly determine all spatiotemporal proximal paths in the continuous form.

Table 3 shows the computational times for the three space-time proximity analysis approaches for different numbers of space-time paths in the database, with $\omega = 100$ m, and target path length was set as 100 links. The brute force approach performed worst for all space-time path amounts, e.g. 1371.524 seconds in the database with 10,000 space-time paths. This outcome occurred because evaluating spatiotemporal proximity relationships between the target path and all other paths is computationally intensive in road networks. The space-time separated approach performed better than brute force, e.g. 80.129 s (16.1 times faster) for the database with 10,000 paths. The number of space-time paths for evaluation was significantly reduced by Step 1, e.g. 1,417 in the same database with 10,000 paths (see Table 4). However, as discussed in Section 4, the space-time paths retrieved by Step 1 were not guaranteed to be actual spatiotemporal proximal paths, and Step 2 was required for further verification. Only 85 paths were determined by Step 2 as actual spatiotemporal proximal paths for the database with 10,000 paths, i.e., approximately 6.0% of those retrieved by Step 1. Verification of all paths retrieved by Step 1 required considerable computational time, e.g. 80.118 s for the database with 10,000 paths. The proposed integrated space-time approach significantly outperformed the other two methods for all the database sizes. For example, the proposed approach consumed only 0.752 seconds for the database with 10,000 paths, i.e., 105.6 times faster than the conventional space-time separated approach. This is because the proposed space-time overlapping operations exactly determine spatiotemporal proximal paths and hence saves the considerable computational costs required for verification.

Table 3. Computational times for three space-time proximity analysis approaches in seconds

Number of space-time paths in database	Space-time integrated approach	Space-time separated approach (Step 1 + Step 2)	Brute force approach
1,000	0.286	9.163 (0.009 + 9.154)	121.863
5,000	0.575	47.831 (0.010 + 47.821)	681.096
10,000	0.752	80.129 (0.011 + 80.118)	1371.524

Table 4. Space-time paths obtained by each step of the separated space-time approach

Number of space-time paths in database	Step 1	Step 2
1,000	180	12
5,000	824	46
10,000	1,417	85

8. Conclusions and further studies

This study proposed a space-time integrated approach based on the space-time buffering concept to efficiently determine spatiotemporal proximal paths by considering space and time dimensions simultaneously. Several directions for future research are worth considering. First, due to space limitations, the case study presented in this article was by no means comprehensive; ArcSDE and Oracle were adopted as the spatial database and the grid index provided by ArcSDE was employed as the spatiotemporal index for space-time paths in CLR space. Development of effective indexes considering the geometry characteristics of space-time paths in CLR space may improve space-time overlapping performance. Integration of emerging non-SQL database techniques (e.g. Mongo DB) may also provide another approach to improve space-time overlapping performance. Second, space-time overlapping operations were introduced based on five widely used measures (M_{CPD}, M_{FD}, M_{APD}, and M_{LCSS}, and δ_{Bundle}). Extension of the proposed space-time overlapping operations based on other proximity should be further investigated. Last but not least, space-time proximity analysis is a critical step for many movement data analysis methods, such as individual-group pattern recognition, space-time path clustering, classification and outlier detection, etc. Subsequent studies should investigate incorporating the proposed space-time integrated approach in these additional movement data analysis methods.

Disclosure statement
No potential conflict of interest was reported by the authors.

Funding
The work described in this article was supported by research grants from the National Key Research and Development Program (No. 2017YFB0503600), the National Natural Science Foundation of China (Nos. 41231171 and 41571149), the Research Grants Council of the Hong Kong Special Administrative Region, China (No. PolyU 152095/17E), the Natural Science Foundation of Hubei Province (2016CFB568), and Chongqing Natural Science Foundation and Frontier Research Planning Project (No. CSTC2014JCYJA00011).

References

Benkert, M., Gudmundsson, J., Hübner, F. and Wolle, T., 2008, Reporting flock patterns. *Computational Geometry*, 41, pp. 111-125.

Agrawal, R., Faloutsos, C. and Swami. A., 1993, Efficient similarity search in sequence databases. Lecture Notes in Computer Science, 730, 69–84.

Charleux, L., 2015, A modification of the time-geographic framework to support temporal flexibility in 'fixed' activities. *International Journal of Geographical Information Science*, 29, pp. 1125-1143.

Chen, L. and Ng., R., 2004. On the marriage of lp-norms and edit distance. In: Proceedings of the 30th International Conference on Very Large Data Bases. VLDB Endowment, 792–803.

Chen, L., Ozsu, M. T. and Oria. V., 2005, Robust and fast similarity search for moving object trajectories. In: Proceedings of the 24th ACMSIGMOD International Conference on Management of Data. ACM, 491-502.

Chen, J., Shaw, S.L., Yu, H.B., Lu, F., Chai, Y.W. and Jia, Q.L., 2011, Exploratory data analysis of activity diary data: A space-time GIS approach. *Journal of Transport Geography*, 19, pp. 394-404.

Chen, B.Y., Li, Q.Q., Wang, D.G., Shaw, S.L., Lam, W.H.K., Yuan, H. and Fang, Z.X., 2013, Reliable space-time prisms under travel time uncertainty. *Annals of the Association of American Geographers*, 103, pp. 1502-1521.

Chen, B.Y., Yuan, H., Li, Q.Q., Lam, W.H., Shaw, S.-L. and Yan, K., 2014, Map-matching algorithm for large-scale low-frequency floating car data. *International Journal of Geographical Information Science*, 28, pp. 22-38.

Chen, B.Y., Yuan, H., Li, Q.Q., Shaw, S.-L., Lam, W.H. and Chen, X., 2016a, Spatiotemporal data model for network time geographic analysis in the era of big data. *International Journal of Geographical Information Science*, 30, pp. 1041-1071.

Chen, H.-P., Chen, B.Y., Wang, Y.F. and Li, Q.Q., 2016b, Efficient geo-computational algorithms for constructing space-time prisms in road networks. *ISPRS International Journal of Geo-Information*, 5, pp. 214.

Chen, B.Y., Wang, Y., Wang, D., Li, Q., Lam, W.H.K. and Shaw, S.-L., 2018, Understanding the impacts of human mobility on accessibility using massive mobile phone tracking data. *Annals of the American Association of Geographers*, DOI:10.1080/24694452.2017.1411244.

de Almeida, V.T. and Güting, R.H., 2005, Indexing the trajectories of moving objects in networks. *Geoinformatica*, 9, pp. 33-60.

Dodge, S., Laube, P. and Weibel, R., 2012, Movement similarity assessment using symbolic representation of trajectories. *International Journal of Geographical Information Science*, 26, pp. 1563-1588.

Eiter, T. and Mannila, H., 1994. Computing discrete Fréchet distance. Wien: Technical University of Wien, CD-TR 94/64.

Erwig, M., Güting, R.H., Schneider, M. and Vazirgiannis, M., 1999, Spatio-temporal data types: An approach to modeling and querying moving objects in databases. *Geoinformatica*, 3, pp. 269-296.

Fredman, M.L. and Tarjan, R.E., 1987, Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the Acm*, 34, pp. 596-615.

Goodchild, M.F., 1987, A spatial analytical perspective on geographical information systems. *International Journal of Geographical Information Systems*, 1, pp. 327-334.

Goodchild, M.F., 2013, Prospects for a Space–Time GIS. *Annals of the Association of American Geographers*, 103, pp. 1072-1077.

Güting, R.H. and Schneider, M., 2005, *Moving objects databases*, San Francisco: Morgan Kaufmann.

Güting, R.H., de Almeida, V.T. and Ding, Z.M., 2006, Modeling and querying moving objects in networks. *The VLDB Journal*, 15, pp. 165-190.

Hägerstraand, T., 1970, What about people in regional science? *Papers in regional science*, 24, pp. 7-24.

Jeung, H.Y., Yiu, M.L., Zhou, X.F. and Jensen, C.S., 2010, Path prediction and predictive range querying in road network databases. *The VLDB Journal*, 19, pp. 585-602.

Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S. and Shen, H.T., 2010, Discovery of convoys in trajectory databases. *Proceedings of the Vldb Endowment*, 1, pp. 1068-1080.

Kim, H.-M. and Kwan, M.-P., 2003, Space-time accessibility measures: A geocomputational algorithm with a focus on the feasible opportunity set and possible activity duration. *Journal of Geographical Systems*, 5, pp. 71-91.

Kuijpers, B. and Othman, W., 2017, The geometry of space-time prisms with uncertain anchors. *International Journal of Geographical Information Science*, In press.

Kwan, M.P., 2000, Interactive geovisualization of activity-travel patterns using three-dimensional geographical information systems: A methodological exploration with a large data set. *Transportation Research Part C*, 8, pp. 185-203.

Kwan, M.-P. and Weber, J., 2003, Individual accessibility revisited: Implications for geographical analysis in the twenty-first Century. *Geographical Analysis*, 35, pp. 341-353.

Kwan, M.-P. and Neutens, T., 2014, Space-time research in GIScience. *International Journal of Geographical Information Science*, 28, pp. 851-854.

Kwan, M.-P., 2016, Algorithmic geographies: Big Data, algorithmic uncertainty, and the production of geographic knowledge. *Annals of the American Association of Geographers*, 106, pp. 274-282.

Laube, P., Imfeld, S. and Weibel, R., 2005, Discovering relative motion patterns in groups of moving point objects. *International Journal of Geographical Information Science*, 19, pp. 639-668.

Li, Q.Q., Chen, B.Y., Wang, Y. and Lam, W.H.K., 2015, A hybrid link-node approach for finding shortest paths in road networks with turn restrictions. *Transactions in GIS*, 19, pp. 915-929.

Li, X. and Lin, H., 2006, Indexing network-constrained trajectories for connectivity-based queries. *International Journal of Geographical Information Science*, 20, pp. 303-328.

Liao, F., Rasouli, S. and Timmermans, H., 2014, Incorporating activity-travel time uncertainty and stochastic space-time prisms in multistate supernetworks for activity-travel scheduling. *International Journal of Geographical Information Science*, 28, pp. 928-945.

Liu, Y., Liu, X., Gao, S., Gong, L., Kang, C., Zhi, Y., Chi, G. and Shi, L., 2015, Social sensing: A new approach to understanding our socioeconomic environments. *Annals of the Association of American Geographers*, 105, pp. 512-530.

Long, J.A. and Nelson, T.A., 2013, A review of quantitative methods for movement data. *International Journal of Geographical Information Science*, 27, pp. 292-318.

Miller, H.J. and Shaw, S.-L., 2001, Geographic information systems for transportation: principles and applications. New York: Oxford University Press

Miller, H.J., 2005, A measurement theory for time geography. *Geographical analysis*, 37, pp. 17-45.

Miller, H.J. and Bridwell, S.A., 2009, A field-based theory for time geography. *Annals of the Association of American Geographers*, 99, pp. 49-75.

Miller, H.J. and Goodchild, M.F., 2015, Data-driven geography. *GeoJournal*, 80, pp. 449-461.

Neutens, T., Van de Weghe, N., Witlox, F. and De Maeyer, P., 2008, A three-dimensional network-based space-time prism. *Journal of Geographical Systems*, 10, pp. 89-107.

Pei, T., Gong, X., Shaw, S.-L., Ma, T. and Zhou, C., 2013, Clustering of temporal event processes. *International Journal of Geographical Information Science*, 27, pp. 484-510.

Shaw, S.-L. and Yu, H.B., 2009, A GIS-based time-geographic approach of studying individual activities and interactions in a hybrid physical-virtual space. *Journal of Transport Geography*, 17, pp. 141-149.

Shaw, S.-L., Tsou, M.-H. and Ye, X., 2016, Editorial: human dynamics in the mobile and big data era. *International Journal of Geographical Information Science*, 30, pp. 1687-1693.

Turdukulov, U., Oswaldo, A., Huisman, C.R.O. and Retsios, V., 2014, Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach. *International Journal of Geographical Information Science*, 28, pp. 2013-2029.

Vlachos, M., Gunopulos, D., and Kollios, G., 2002. Robust similarity measures for mobile object trajectories. In: 5th international workshop on mobility in databases and distributed systems (MDDS), 2–6 September Aix-en-Provence, France, 721–726. Los Alamitos, CA: IEEE Computer Society Press.

Wolfson, O., Xu, B., Chamberlain, S. and Jiang, L., 1998. Moving object databases: issues and solutions. In: Proceedings of the 10th international conference on scientific and statistical database management, Capri. New York, NY: IEEE, 111–122.

Xiang, L., Wu, T. and Ettema, D., 2017, An intersection-based trajectory-region movement study. *Transactions in GIS*. In press.

Xu, J.Q. and Güting, R.H., 2013, A generic data model for moving objects. *Geoinformatica*, 17, pp. 125-172.

Yang, C., Huang, Q., Li, Z., Liu, K. and Hu, F., 2017, Big Data and cloud computing: innovation opportunities and challenges. *International Journal of Digital Earth*, 10, pp. 13-53.

Yu, H. and Shaw, S.-L., 2008, Exploring potential human activities in physical and virtual spaces: a spatio-temporal GIS approach. *International Journal of Geographical Information Science*, 22, pp. 409-430.

Žalik, B., Zadravec, M. and Clapworthy, G.J., 2003, Construction of a non-symmetric geometric buffer from a set of line segments. *Computers & geosciences*, 29, pp. 53-63.

Zhang, H.C., Lu, F. and Xu, J.Q., 2016, Modeling and querying moving objects with social relationships. *ISPRS International Journal of Geo-Information*, 5, p. 121.

Zheng, Y., 2015, Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, 6, pp. 1-41.

Appendix

Definition A1. Given a polyline L^q and a spatial tolerance ω, the spatial buffer $SB^q(\omega)$ delimits all possible spatial points (x_k, y_k) satisfying $SB^q(\omega) = \{(x_k, y_k) \mid D((x_k, y_k),(x_l, y_l)) \le \omega, \forall(x_l, y_l) \in L^q\}$.

Proposition A1. Given a space-time path P^q, the projection of its space-time buffer $STB^q(\omega)$ onto the geographical space is equivalent to the spatial buffer of its projected polyline L^q.

Proof. Let $SB^q(\omega)$ be the spatial buffer of projected polyline L^q, and $PSTB^q(\omega)$ be the projection of space–time buffer $STB^q(\omega)$ onto the geographical space. Given any point $(x_k, y_k) \in SB^q(\omega)$, according to the definition of spatial buffering operation (i.e., Definition A1), there exists a point $(x_l, y_l) \in L^q$ satisfying $D((x_k, y_k),(x_l, y_l)) \le \omega$. Since spatial point $(x_l, y_l) = P^q(t_k)$ is the projection of space-time point $(x_l, y_l, t_k) \in P^q$, we have

$D\big((x_k,y_k),P^q(t_k)=(x_l,y_l,t_k)\big)\le\omega$. According to the definition of space-time buffering operation (i.e., Definition 1), we have $(x_k,y_k,t_k)\in STB^q(\omega)$. Therefore, we have $(x_k,y_k)\in PSTB^q(\omega)$ and $SB^q(\omega)\subset PSTB^q(\omega)$. Then given any point $(x_i,y_i,t_i)\in STB^q(\omega)$, we have $(x_i,y_i)\in PSTB^q(\omega)$. According to Definition 1, there exist a point $P^q(t_i)=(x_j,y_j)$ satisfying $D\big((x_i,y_i),P^q(t_i)\big)\le\omega$. Since spatial point (x_j,y_j) is the projection of space-time point (x_j,y_j,t_i) on space-time path P^q, we have $D((x_i,y_i),(x_j,y_j))\le\omega$ on the geographical space. According to Definition A1, we have $(x_i,y_i)\in SB^q(\omega)$ and $PSTB^q(\omega)\subset SB^q(\omega)$. Thus, we have $PSTB^q(\omega)=SB^q(\omega)$. W

Proposition A2. The CPD operation can retrieve all space-time paths with M_{CPD} to P^q $\le\omega$.
Proof. Let $\bar{Q}_{CPD}=\{...,P^c,...\}$ be the retrieved path set of the CPD operation; and $Q_{CPD}=\{...,P^b,...\}$ be the complete set of all space-time paths with M_{CPD} to P^q $\le\omega$. Given any path $P^c\in\bar{Q}_{CPD}$, $STB^q(\omega)$ intersects P^c, and there exists at least one space-time point $P^c(t_k)$ within $STB^q(\omega)$. From Definition 1, we have $D\big(P^q(t_k),P^c(t_k)\big)\le\omega$. According to Eq. (5), we have $M_{CPD}(P^q,P^c)=\min\limits_{\forall t_k}\Big(D\big(P^q(t_k),P^c(t_k)\big)\Big)\le D\big(P^q(t_k),P^c(t_k)\big)\le\omega$. Therefore, we have $P^c\in Q_{CPD}$ and $\bar{Q}_{CPD}\subset Q_{CPD}$. Then, given any path $P^b\in Q_{CPD}$, we have $M_{CPD}(P^q,P^b)\le\omega$. Without loss of generality, suppose that the minimum distance $M_{CPD}(P^q,P^b)$ achieves at time stamp t_m. Thus, $D\big(P^q(t_m),P^b(t_m)\big)\le\omega$ holds. From Definition 1, we have $P^b(t_m)$ within $STB^q(\omega)$. Therefore, P^b intersects $STB^q(\omega)$, and we have $P^b\in\bar{Q}_{CPD}$ and $Q_{CPD}\subset\bar{Q}_{CPD}$. Therefore, $\bar{Q}_{CPD}=Q_{CPD}$ holds. □

Proposition A3. The FD operation can retrieve all space-time paths with M_{FD} to P^q $\le\omega$.
Proof. Let $\bar{Q}_{FD}=\{...,P^c,...\}$ be the retrieved path set of the FD operation; and $Q_{FD}=\{...,P^b,...\}$ be the complete set of all space-time paths with M_{FD} to $P^q\le\omega$. Given any path $P^c\in\bar{Q}_{FD}$, we have P^c within $STB^q(\omega)$. Thus, $P^c(t_k)$ is within $STB^q(\omega)$ for any time stamp. From Definition 1, we have $D\big(P^q(t_k),P^c(t_k)\big)\le\omega$ for any time stamp t_k. From Eq. (6), we have $\max\limits_{\forall t_k}\Big(D\big(P^q(t_k),P^c(t_k)\big)\Big)=M_{FD}(P^q,P^c)\le\omega$. Therefore, we have $P^c\in Q_{FD}$ and $\bar{Q}_{FD}\subset Q_{FD}$. Then, given any path $P^b\in Q_{FD}$, we have $M_{FD}(P^q,P^b)=\max\limits_{\forall t_k}\Big(D\big(P^q(t_k),P^b(t_k)\big)\Big)\le\omega$ according to Eq. (6). Consequently, we have $D\big(P^q(t_k),P^b(t_k)\big)\le\omega$ for any time stamp t_k. From Definition 1, we have $P^b(t_k)$ within $STB^q(\omega)$ for any time stamp t_k. Thus, P^b is within $STB^q(\omega)$, and $P^b\in\bar{Q}_{FD}$ and $Q_{FD}\subset\bar{Q}_{FD}$. Therefore, we have $\bar{Q}_{FD}=Q_{FD}$. □

Proposition A4. The APD operation retrieves all space-time paths with M_{APD} to $P^q\le\omega$.
Proof. Let $\bar{Q}_{APD}=\{...,P^c,...\}$ be the retrieved path set of the APD operation; and $Q_{APD}=\{...,P^b,...\}$ be the complete set of all space-time paths with M_{APD} to P^q $\le\omega$. Given any path $P^c\in\bar{Q}_{APD}$, we

have $M_{APD}(P^q, P^c) \le \omega$, since P^c was not removed in the second step of the APD operation. Thus, we have $P^c \in Q_{APD}$ and $\bar{Q}_{APD} \subset Q_{APD}$. Then, given any path $P^b \in Q_{APD}$, we have $M_{APD}(P^q, P^b) = \underset{\forall t_k}{mean}\left(D\left(P^q(t_k), P^b(t_k)\right)\right) \le \omega$ according to Eq. (7). Thus, we have $M_{CPD}(P^q, P^b) = \min\limits_{\forall t_k}\left(D\left(P^q(t_k), P^b(t_k)\right)\right) \le \underset{\forall t_k}{mean}\left(D\left(P^q(t_k), P^b(t_k)\right)\right) \le \omega$. Therefore, from Proposition 2, P^b was retrieved in the CPD operation (i.e., the first step). Because $M_{APD}(P^q, P^b) \le \omega$ holds, the space-time path P^b was not removed in the second step. Thus, we have $P^b \in \bar{Q}_{APD}$ and $Q_{APD} \subset \bar{Q}_{APD}$. Therefore, $\bar{Q}_{APD} = Q_{APD}$ holds. □

Proposition A5. The LCSS operation retrieves all space-time paths with M_{LCSS} to P^q ≥ 0.

Proof. Let $\bar{Q}_{LCSS} = \{..., P^c, ...\}$ be the retrieved path set of the LCSS operation; and $Q_{LCSS} = \{..., P^b, ...\}$ be the complete set of all space-time paths with M_{LCSS} to P^q ≥ 0. Given any $P^c \in \bar{Q}_{LCSS}$, we have space-time buffer $STB^q(\omega)$ intersecting P^c. Without loss of generality, suppose that $STB^q(\omega)$ intersects P^c during period $[t_s, t_e]$. From Definition 1, we have $D\left(P^q(t_k), P^c(t_k)\right) \le \omega$ for any $t_k \in [t_s, t_e]$. From Eq. (7), we have $M_{LCSS}(P^q, P^c) = t_e - t_s > 0$. Thus, we have $P^c \in Q_{LCSS}$ and $\bar{Q}_{LCSS} \subset Q_{LCSS}$. Then, given any path $P^b \in Q_{LCSS}$, we have $M_{LCSS}(P^q, P^b) > 0$. Without loss of generality, suppose that $D\left(P^q(t_k), P^b(t_k)\right) \le \omega$ for any $t_k \in [t_{s'}, t_{e'}]$. From Definition 1, $STB^q(\omega_d)$ intersects P^b during $[t_{s'}, t_{e'}]$. Thus, we have $P^b \in \bar{Q}_{LCSS}$ and $Q_{LCSS} \subset \bar{Q}_{LCSS}$ $Q_{LCSS} \subset \bar{Q}_{LCSS}$. Therefore, $\bar{Q}_{LCSS} = Q_{LCSS}$ holds. □

Proposition A6. The bundling operation retrieves all space-time paths bundled with P^q during period $[t_s, t_e]$.

Proof. Let $\bar{Q}_{Bundling} = \{..., P^c, ...\}$ be the retrieved path set of the bundling operation; and $Q_{Bundling} = \{..., P^b, ...\}$ be the complete set of all space-time paths bundled with P^q. Given any path $P^c \in \bar{Q}_{Bundling}$, we have its sub-path contained by $STB^q(\omega)$ during $[t_s, t_e]$. From Definition 1, we have $D\left(P^q(t_k), P^c(t_k)\right) \le \omega$ for any $t_k \in [t_s, t_e]$. From Eq. (4), we have $\delta_{\mathrm{Bundle}}(\mathrm{P^q}, \mathrm{P^c}) = 1$. Thus, we have $P^c \in Q_{Bundling}$ and $\bar{Q}_{Bundling} \subset Q_{Bundling}$. Then, given any path $P^b \in Q_{Bundling}$, we have $D\left(P^q(t_k), P^b(t_k)\right) \le \omega$ for any time stamp $t_k \in [t_s, t_e]$, according to Eq. (4). Thus, from Definition 1, P^b intersected $STB^q(\omega)$ during $t_k \in [t_s, t_e]$. Thus, we have $P^b \in \bar{Q}_{Bundling}$ and $Q_{Bundling} \subset \bar{Q}_{Bundling}$. Therefore, $\bar{Q}_{Bundling} = Q_{Bundling}$ holds. □

Proposition A7. The network space-time buffer $\widetilde{STB}^q(\omega)$ in CLR space is an equivalent representation of $STB^q(\omega)$ in (x, y, t) space.

Proof. According to the transformation rule, we prove that $\widetilde{STB}^q_{ij}(\omega)$ of any path segment $s^q_{ij} \in P^q$ is an equivalent representation of $STB^q_{ij}(\omega)$ as below. $\widetilde{STB}^q_{ij}(\omega)$ is the union of

$\widetilde{STB}_{ij}^{f}(\omega)=<\tilde{O}_{u,ij}^{f},\tilde{O}_{v,ij}^{f},...>$ and $\widetilde{STB}_{ij}^{b}(\omega)=<\tilde{O}_{u,ij}^{b},\tilde{O}_{v,ij}^{b},...>$. The $\tilde{O}_{u,ij}^{f}=<...,\tilde{c}_{k}^{u},...>$ entity is transformed from $O_{u,ij}^{f}=<...,c_{k}^{u},...>$ (shown in Figure 8) by obtaining any point $\tilde{c}_{k}^{u}=(z_{k}^{u}=l_{u}+m_{k},t_{k})$ from $c_{k}^{u}=(l_{u},m_{k},t_{k})$. Then, $\tilde{O}_{u,ij}^{f}$ and $O_{u,ij}^{f}$, upon the same link a_{u}, have same scale and shape; and can convert into each other, i.e., with the one-to-one relationship. Thus, $\tilde{O}_{u,ij}^{f}$ is an equivalent representation of $O_{u,ij}^{f}$. The $\tilde{O}_{v,ij}^{f}=<...,\tilde{c}_{k}^{v},...>$ entity is transformed from $O_{v,ij}^{f}=<...,c_{k}^{v},...>$ (shown in Figure 9) by obtaining any point $\tilde{c}_{k}^{v}=(z_{k}^{v}=l_{v}+m_{k},t_{k})$ from $c_{k}^{v}=(l_{v},m_{k},t_{k})$. Then, $\tilde{O}_{v,ij}^{f}$ and $O_{v,ij}^{f}$, upon the same nearby link a_{v}, also have same scale and shape; and can convert into each other. Thus, $\tilde{O}_{v,ij}^{f}$ is an equivalent representation of $O_{v,ij}^{f}$. Therefore, $\widetilde{STB}_{ij}^{f}(\omega)=<\tilde{O}_{u,ij}^{f},\tilde{O}_{v,ij}^{f},...>$ is an equivalent representation of $STB_{ij}^{f}(\omega)=<O_{u,ij}^{f},O_{v,ij}^{f},...>$.

Similarly, the $\tilde{O}_{u,ij}^{b}=<...,\tilde{c}_{k}^{u},...>$ entity is transformed from $O_{u,ij}^{b}=<...,c_{k}^{u},...>$ (shown in Figure 10) by obtaining any point $\tilde{c}_{k}^{u}=(z_{k}^{u}=l_{u}+m_{k},t_{k})$ from $c_{k}^{u}=(l_{u},m_{k},t_{k})$. Then, $\tilde{O}_{u,ij}^{b}$ and $O_{u,ij}^{b}$, upon the same link a_{u}, have same scale and shape; and can convert into each other, i.e., with the one-to-one relationship. Thus, $\tilde{O}_{u,ij}^{b}$ is an equivalent representation of $O_{u,ij}^{b}$. The $\tilde{O}_{v,ij}^{b}=<...,\tilde{c}_{k}^{v},...>$ entity is transformed from $O_{v,ij}^{b}=<...,c_{k}^{v},...>$ (shown in Figure 11) by obtaining any point $\tilde{c}_{k}^{v}=(z_{k}^{v}=l_{v}+m_{k},t_{k})$ from $c_{k}^{v}=(l_{v},m_{k},t_{k})$. Then, $\tilde{O}_{v,ij}^{b}$ and $O_{v,ij}^{b}$, upon the same nearby link a_{v}, also have same scale and shape; and can convert into each other. Thus, $\tilde{O}_{v,ij}^{b}$ is an equivalent representation of $O_{v,ij}^{b}$. Therefore, $\widetilde{STB}_{ij}^{b}(\omega)=<\tilde{O}_{u,ij}^{b},\tilde{O}_{v,ij}^{b},...>$ is an equivalent representation of $STB_{ij}^{b}(\omega)=<O_{u,ij}^{b},O_{v,ij}^{b},...>$; and $\widetilde{STB}_{ij}^{q}(\omega)=\widetilde{STB}_{ij}^{f}(\omega)\cup\widetilde{STB}_{ij}^{b}(\omega)$ is also an equivalent representation of $STB_{ij}^{q}(\omega)=STB_{ij}^{f}(\omega)\cup STB_{ij}^{b}(\omega)$. □

Proposition A8. Spatiotemporal proximity relationships (i.e., $\delta_{\mathrm{Intersect}}$, δ_{Bundle}, M_{CPD}, M_{FD}, M_{APD} and M_{LCSS}) between $\tilde{P}^{q}$ and $\tilde{P}^{b}$ in CLR space are equal to those between P^{q} and P^{b} in (x,y,t) space.

Proof. For any time stamp t_{k}, locations on two paths are $\tilde{P}^{q}(t_{k})=(z_{i}=l_{u}+m_{i},t_{k})$ and $\tilde{P}^{b}(t_{k})=(z_{j}=l_{v}+m_{j},t_{k})$ in CLR space. According to Proposition 7, the corresponding locations in (x,y,t) space are $P^{q}(t_{k})=(l_{u},m_{i},t_{k})$ and $P^{b}(t_{k})=(l_{v},m_{j},t_{k})$. Thus, we have $D_{N}\left(\tilde{P}^{q}(t_{k}),\tilde{P}^{b}(t_{k})\right)=D_{N}\left(z_{i},z_{j}\right)=D_{N}\left((l_{u},m_{i}),(l_{v},m_{j})\right)=D_{N}\left(P^{q}(t_{k}),P^{b}(t_{k})\right)$ for any time stamp t_{k}. Therefore, according to Eqs. (3–9), we have $\delta_{\mathrm{Intersect}}(\tilde{P}^{q},\tilde{P}^{b})=\delta_{\mathrm{Intersect}}(P^{q},P^{b})$, $\delta_{\mathrm{Bundle}}(\tilde{P}^{q},\tilde{P}^{b})=\delta_{\mathrm{Bundle}}(P^{q},P^{b})$, $M_{CPD}(\tilde{P}^{q},\tilde{P}^{b})=M_{CPD}(P^{q},P^{b})$, $M_{FD}(\tilde{P}^{q},\tilde{P}^{b})=M_{FD}(P^{q},P^{b})$, $M_{APD}(\tilde{P}^{q},\tilde{P}^{b})=M_{APD}(P^{q},P^{b})$ and $M_{LCSS}(\tilde{P}^{q},\tilde{P}^{b})=M_{LCSS}(P^{q},P^{b})$. □

Proposition A9. The network space-time buffer $\widetilde{STB}^{q}(\omega)$ of space-time path $\tilde{P}^{q}$ delimits all possible space-time points $\tilde{c}_k^v = (z_k^v, t_k)$ in CLR space satisfying $D_N\left(\tilde{c}_k^v, \tilde{P}^q(t_k)\right) \le \omega, t_s^q \le \forall t_k \le t_e^q$.

Proof. Let $STB^q(\omega)$ be the network space-time buffer of the space-time path P^q. According to Proposition 7, at any time stamp t_k, location $P^q(t_k) = (l_u, m_k, t_k)$ in (x, y, t) space can be transformed into CLR space as $\tilde{P}^q(t_k) = (z_k^u = l_u + m_k, t_k)$. According to the definition of $STB^q(\omega)$, any location $c_k^v = (l_v, m_k, t_k)$ satisfying $D_N\left(c_k^v, P^q(t_k)\right) \le \omega$ is within $STB^q(\omega)$. We can transform c_k^u into CLR space as $\tilde{c}_k^v = (z_k^v = l_v + m_k, t_k)$, then $D_N\left(\tilde{c}_k^v, \tilde{P}^q(t_k)\right) = D_N\left(c_k^v, P^q(t_k)\right) \le \omega$. According to Proposition 9, $\tilde{c}_k^v$ is within $\widetilde{STB}^q(\omega)$. Therefore, for any t_k, any location $\tilde{c}_k^v = (z_k^v, t_k)$ satisfying $D_N\left(\tilde{c}_k^v, \tilde{P}^q(t_k)\right) \le \omega$ is within $\widetilde{STB}^q(\omega)$.□

Lemma A1. $\bar{A}_j^f(\omega) = \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$.

Proof. Given any network location $(l_v, m_k) \notin a_u$, we have $D_{SP}(P^q(t_i), (l_v, m_k)) = m_j d_u - m_i d_u + D_{SP}(P^q(t_j), (l_v, m_k))$, because U-turns are allowed at only the ending node, and the shortest path between $P^q(t_i)$ and (l_v, m_k) must pass through the sub-link from $P^q(t_i)$ to $P^q(t_j)$. We can prove $\bar{A}_j^f(\omega) \subset \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$ as follows. Given any network location $(l_w, m_w) \in \bar{A}_j^f(\omega)$, we have $D_{SP}(P^q(t_j), (l_w, m_w)) \le \omega$. Then, we have $D_{SP}(P^q(t_i), (l_w, m_w)) = D_{SP}(P^q(t_j), (l_w, m_w)) + m_j d_u - m_i d_u \le \omega + m_j d_u - m_i d_u$. Therefore, (l_w, m_w) is within $\bar{A}_i^f(\omega + m_j d_u - m_i d_u)$, and $\bar{A}_j^f(\omega) \subset \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$.
Subsequently, we can prove $\bar{A}_i^f(\omega + m_j d_u - m_i d_u) \subset \bar{A}_j^f(\omega)$ as follows. Given any network location $(l_r, m_r) \in \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$, we have $D_{SP}(P^q(t_i), (l_r, m_r)) \le \omega + m_j d_u - m_i d_u$. Then, we have $D_{SP}(P^q(t_j), (l_r, m_r)) = D_{SP}(P^q(t_i), (l_r, m_r)) - (m_j d_u - m_i d_u) \le \omega$. Thus, (l_r, m_r) is within $\bar{A}_j^f(\omega)$; and $\bar{A}_i^f(\omega + m_j d_u - m_i d_u) \subset \bar{A}_j^f(\omega)$.
Therefore, $\bar{A}_j^f(\omega) = \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$ holds. □

Lemma A2. $\bar{A}_i^f(\omega) \subset \bar{A}_j^f(\omega)$.

Proof. According to Lemma A1, we have $\bar{A}_j^f(\omega) = \bar{A}_i^f(\omega + m_j d_u - m_i d_u)$. Since $m_j d_u - m_i d_u > 0$, we have $\bar{A}_i^f(\omega) \subset \bar{A}_j^f(\omega)$. □

Lemma A3. $\bar{A}_i^b(\omega) = \bar{A}_j^b(\omega + m_j d_u - m_i d_u)$.

Proof. Given any network location $(l_v, m_k) \notin a_u$, we have $D_{SP}((l_v, m_k), P^q(t_j),) = D_{SP}((l_v, m_k), P^q(t_i)) + m_j d_u - m_i d_u$, because U-turns are allowed at only the ending node and the shortest path between (l_v, m_k) and $P^q(t_j)$ must pass through the sub-link from $P^q(t_i)$ to $P^q(t_j)$. Firstly, we can prove $\bar{A}_i^b(\omega) \subset \bar{A}_j^b(\omega + m_j d_u - m_i d_u)$ as follows. Given any network location $(l_w, m_w) \in \bar{A}_i^b(\omega)$, we have $D_{SP}((l_w, m_w), P^q(t_i)) \le \omega$. Then, we have $D_{SP}((l_w, m_w), P^q(t_j)) = D_{SP}((l_w, m_w), P^q(t_i)) + m_j d_u - m_i d_u \le \omega + m_j d_u - m_i d_u$. Therefore, (l_w, m_w) is within $\overline{A}_j^b(\omega + m_j d_u - m_i d_u)$; and thus $\overline{A}_i^b(\omega) \subset \overline{A}_j^b(\omega + m_j d_u - m_i d_u)$. Subsequently, we can prove $\overline{A}_j^b(\omega + m_j d_u - m_i d_u) \subset \overline{A}_i^b(\omega)$ as follows. Given any network location $(l_r, m_r) \in \overline{A}_j^b(\omega + m_j d_u - m_i d_u)$, we have $D_{SP}((l_r, m_r), P^q(t_j)) \leq \omega + m_j d_u - m_i d_u$. Then, we have $D_{SP}((l_r, m_r), P^q(t_i)) = D_{SP}((l_r, m_r), P^q(t_j)) - (m_j d_u - m_i d_u) \leq \omega$. Thus, (l_r, m_r) is within $\overline{A}_i^b(\omega)$; and $\overline{A}_j^b(\omega + m_j d_u - m_i d_u) \subset \overline{A}_i^b(\omega)$. Therefore, $\overline{A}_i^b(\omega) = \overline{A}_j^b(\omega + m_j d_u - m_i d_u)$ holds.□

Lemma A4. $\overline{A}_j^b(\omega) \subset \overline{A}_i^b(\omega)$.

Proof. According to Lemma A3, we have $\overline{A}_i^b(\omega) = \overline{A}_j^b(\omega + m_j d_u - m_i d_u)$. Since $m_j d_u - m_i d_u > 0$, we have $\overline{A}_j^b(\omega) \subset \overline{A}_i^b(\omega)$. □