International Journal of Geo-Information

MDPI

Article

NS-DBSCAN: A Density-Based Clustering Algorithm in Network Space

Tianfu Wang [1], Chang Ren [2], Yun Luo [1] and Jing Tian [1,3,4,*]

1 School of Resource and Environmental Science, Wuhan University, 129 Luoyu Road, Wuhan 430079, China; tianfu.wang@whu.edu.cn (T.W.); Yun_Luo@whu.edu.cn (Y.L.)
2 State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China; imrc@whu.edu.cn
3 Key Laboratory of Geographic Information System, Ministry of Education, Wuhan University, 129 Luoyu Road, Wuhan 430079, China
4 Key Laboratory of Digital Mapping and Land Information Application Engineering, National Administration of Surveying, Mapping and Geoinformation, Wuhan 430079, China
* Correspondence: tianjing_sres@whu.edu.cn; Tel.: +86-1362-863-6229

Received: 27 March 2019; Accepted: 5 May 2019; Published: 8 May 2019

Abstract: Spatial clustering analysis is an important spatial data mining technique. It divides objects into clusters according to their similarities in both location and attribute aspects. It plays an essential role in density distribution identification, hot-spot detection, and trend discovery. Spatial clustering algorithms in the Euclidean space are relatively mature, while those in the network space are less well researched. This study aimed to present a well-known clustering algorithm, named density-based spatial clustering of applications with noise (DBSCAN), to network space and proposed a new clustering algorithm named network space DBSCAN (NS-DBSCAN). Basically, the NS-DBSCAN algorithm used a strategy similar to the DBSCAN algorithm. Furthermore, it provided a new technique for visualizing the density distribution and indicating the intrinsic clustering structure. Tested by the points of interest (POI) in Hanyang district, Wuhan, China, the NS-DBSCAN algorithm was able to accurately detect the high-density regions. The NS-DBSCAN algorithm was compared with the classical hierarchical clustering algorithm and the recently proposed density-based clustering algorithm with network-constraint Delaunay triangulation (NC_DT) in terms of their effectiveness. The hierarchical clustering algorithm was effective only when the cluster number was well specified, otherwise it might separate a natural cluster into several parts. The NC_DT method excessively gathered most objects into a huge cluster. Quantitative evaluation using four indicators, including the silhouette, the R-squared index, the Davis–Bouldin index, and the clustering scheme quality index, indicated that the NS-DBSCAN algorithm was superior to the hierarchical clustering and NC_DT algorithms.

Keywords: clustering analysis; DBSCAN algorithm; network spatial analysis; spatial data mining

1. Introduction

The first law of geography states that closer spatial entities are more strongly related to each other than the distant ones [1]. Therefore, detecting spatial clusters of spatial events is an important technique in spatial analysis [2]. Clustering analysis has been widely used in hot-spot detection [3–8], traffic accident analysis [9,10], climate regionalization [11,12], and earthquake clustering identification [13–15]. Clustering analysis is generally divided into two categories, one using the spatial point pattern analyses to discover aggregated points with statistical indicators and the other obtaining clusters from the perspective of data mining [16]. The spatial point pattern methods, such as the local K-function [17,18],

local Moran's I [19,20], Getis-Ord Gi [10], scan statistics [21], and local indicators of mobility association (LIMA) [22], are commonly adopted for indicating aggregated regions and discovering the density trend of spatial dataset. Some of them have already been applied to network-constraint events [10,17,23]. In contrast to spatial point pattern methods, generic clustering algorithms for multidimensional features not only delineate aggregated configuration of dataset but also precisely depict specific shapes of separated clusters.

Clustering algorithms divide a dataset into several clusters, with similar objects in the same cluster and dissimilar objects in different clusters [24–33]. They have been widely used in geoscience for spatial data [2,34]. Conventional clustering algorithms can be separated into four general categories: partitioning, hierarchical, density-based, and grid-based algorithms. The partitioning algorithms divide a dataset into several subsets by continuously optimizing an objective function. The hierarchical algorithms obtain a dendrogram by merging or splitting clusters. The density-based algorithms expand from the dense areas to obtain high-density clusters separated by low-density regions. The grid-based algorithms are a composite approach that first divides a dataset into several subregions and applies other clustering algorithms to each subregion.

Taking the Euclidean distance of points as a measure of dissimilarity, generic point clustering algorithms are mostly designated for the planar space. The partitioning algorithms can easily detect spherical clusters. The *k*-means algorithm is a traditional partitioning algorithm, and it is susceptible to outliers. The *k*-medoids algorithms [25,26,35–37] overcome this shortcoming. Among hierarchical clustering algorithms, agglomerative algorithms are commonly used. A dendrogram of hierarchical clustering algorithm cannot be modified once it is formed, making it inapplicable to incremental clustering. The BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) algorithm [27] solves this problem by maintaining a clustering feature tree and is able to distinguish noises (objects that do not belong to any cluster). The CURE (Clustering Using REpresentatives) algorithm [28] introduces the concept of representative objects to enhance efficiency. Moreover, the CHAMELEON algorithm [29] performs hierarchical clustering by constructing a k-nearest neighbor graph and is able to find clusters of arbitrary shape and variable size. The DBSCAN algorithm [30] is a famous algorithm that excels in detecting arbitrary-shaped clusters and removing noises. Many algorithms improve and extend the DBSCAN algorithm. For example, the OPTICS (Ordering Points To Identify the Clustering Structure) algorithm [31] focuses on identifying variable density clusters. The ADCN (Anisotropic Density-based Clustering for discovering spatial point patterns with Noise) algorithm [32] is able to find linear and ring clusters using anisotropic elliptical neighborhood. In addition to the aforementioned traditional spatial clustering algorithms and their improvements, some scholars also proposed new clustering paradigms, enriching the clustering methodology. The FTSC (Field-Theory based Spatial Clustering) [33] and ASCDT (Adaptive Spatial Clustering based on Delaunay Triangulation) algorithms [14] proposed by Deng draw on the basic principle of field theory and interpret the meaning of clusters from the perspective of cohesion. The ASCDT algorithm [14] breaks the long edges in Delaunay triangulation by global and local constraints to obtain clusters. The ASCDT+ algorithm [38] is outstanding in clustering points in the presence of obstacles (e.g., mountains) and facilitators (e.g., highway). In some cases, the similarities of both spatial and nonspatial attributes have to be considered based on two options. The first option deals with the spatial and nonspatial attributes separately, and the clusters meet both spatial and nonspatial criteria, such as the CLARANS (Clustering Large Applications based on RANdomized Search) [26] and DBSC (Density-Based Spatial Clustering) [11] algorithms. The second option handles the similarity for nonspatial attributes and spatial distances simultaneously to define an overall similarity between objects, such as the GDBSCAN (Generalized DBSCAN) algorithm [39] and DBRS (Density-Based Spatial clustering with Random Sampling) algorithm [40]. Moreover, the SEClu (Spatial Entropy-based Clustering) algorithm [41] also takes the spatial autocorrelation (SAR) of nonspatial attribute into account and discovers clusters with high SAR.

The aforementioned spatial clustering algorithms are mostly used in the planar space, while rarely applied to the network-constraint events [9] (e.g., shops alongside the streets, car accidents on the

roads, and earthquakes along coastlines), in spite that many phenomena in the real world, especially in city environment, are constrained by spatial networks. The network-constraint events are distributed in discrete and heterogeneous network space. If they are analyzed in continuous and homogeneous planar space, the conclusions can be unsatisfactory or even erroneous [42]. In recent years, the methods of network spatial analysis have received much more attention [17,43–48]. These methods remarkably change the distance measure from Euclidean distance in planar space to the shortest-path distance on networks. At the same time, spatial clustering algorithms also have been extended for network-constraint events. Yiu [49] proposed the k-medoids algorithm, ε-Link algorithm (extended from DBSCAN), and single-link algorithm for network events. On the basis of the algorithm proposed by Yiu, Sugihara [50] further improved the hierarchical clustering algorithm for network events. He used five distance measures to hierarchically cluster event points. It is quite a mature clustering algorithm and has been integrated into Spatial Analysis along Networks (SANET) [51], a toolbox for network spatial analysis. Stefanakis [52] extended the DBSCAN algorithm to cluster events in dynamic networks whose edges might be temporarily inaccessible. Moreover, Chen [53] proposed a framework for clustering moving objects in a spatial network based on the notion of cluster block. Deng [16] extended the DBSCAN algorithm to network space and proposed a density-based clustering algorithm that requires no input parameters. Shi [54] proposed a new framework for cluster detection in traffic networks based on the spatial–temporal flow modeling. Spatial clustering algorithms in network space were also proposed for different kinds of applications. Oliveira [55,56] extended the DBSCAN and OPTICS algorithms to the water distribution pipe breakage to locate the region of high-breakage densities. Smaltschinski [57] clustered the harvest stands on a road network using the single-linkage algorithm to reduce movements of harvesters and forwarders.

Theoretically, taking the shortest-path distance as the distance measure, most of clustering algorithms in the planar space can be extended to the network space. However, in fact, the shift is not that straightforward because of the high complexity of computing the shortest-path distance between two objects. For example, the partitioning algorithms are insufficient in clustering network events because they are very time-consuming due to repeated graph traverse and are not effective at all [49]. The hierarchical and density-based algorithms perform efficiently and effectively for network events, and the density-based algorithms actually have a natural advantage in detecting arbitrarily shaped clusters and distinguishing noises. Besides, the number of clusters is the input parameter for partitioning algorithms and often the terminal condition of hierarchical algorithms, while they are actually difficult to determine. However, the density-based algorithms do not take the number of clusters as an input parameter, which is beneficial to the discovery of natural clusters [31]. Yiu, Oliveira, and Stefanakis once extended the DBSCAN algorithm to network-constraint events. These algorithms shared the same problem, that is, two global input parameters *eps* (the distance to determine a point's neighborhood) and *MinPts* (the minimal density of dense objects) were compulsory. The clustering results were very sensitive to the input parameters; therefore, obtaining good clustering results heavily depended on the user's domain knowledge [31]. To address this problem, the network space DBSCAN (NS-DBSCAN) algorithm helped improve the aforementioned drawbacks in two aspects. (1) The algorithm provided a new way to visualize the density distribution of event points, according to which two parameters (*eps* and *MinPts*) could be better determined. (2) A local shortest-path distance (LSPD) algorithm was proposed to improve efficiency.

The rest of the article is organized as follows. Section 2 briefly introduces the basic idea of the proposed algorithm and the notion of density ordering. In Section 3, two core steps of the proposed algorithm are presented. Section 4 evaluates the effectiveness of NS-DBSCAN algorithm on the points of interest (POI) in Hanyang district, Wuhan, China. Section 5 discusses the basic principles to determine two input parameters and the ways to deal with the one-way and dead-end roads. Section 6 concludes with a summary of the study, implications of the proposed algorithm to current research, and some hints for future research.

2. Basic Idea

Figure 1 is a simulated dataset used to explain the cluster detection for network events. Starting from an event point p, the part of the network that can be reached within a distance *eps* is p's *eps*-neighborhood; p is a central point, and the event points within p's *eps*-neighborhood are p's *eps*-neighbors N_*eps* (p). The number of neighbors is defined as the density of p. In Figure 1, the density of P_5 is 4.

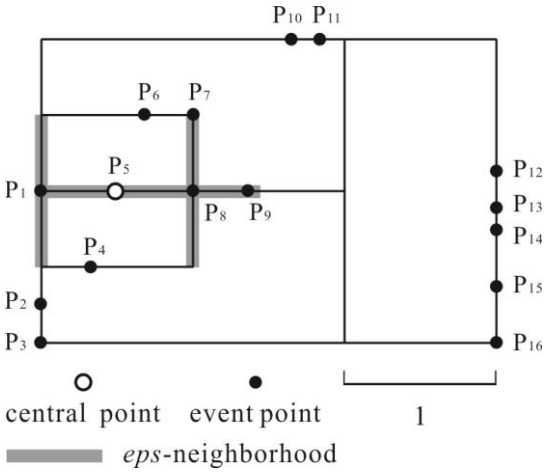

Figure 1. A simulated dataset including road network and event points. The 1-neighborhood of central point P_5 is marked as a thick gray line, covering four event points (P_1, P_7, P_8, and P_9).

The idea of density ordering is based on the fact that the densities of spatially adjacent event points are similar. Starting from an event point, the NS-DBSCAN algorithm iteratively puts the surrounding points into a density ordering table in descending order of their densities. If the event points are spatially close, they tend to have similar densities and are also close in the density ordering table. The densities of event points in the density ordering table can be visualized as a bar chart, resulting in a density ordering graph that presents hills of varying sizes corresponding to implicit clusters.

Figure 2 shows how the density ordering graph was obtained and how it reflected the density distribution of event points. The NS-DBSCAN algorithm randomly selected an event point as the central point. Irrespective of which event point was selected first, the NS-DBSCAN algorithm went straight to the peak of local density and visited every event point around it. Figure 2 shows that *cp1* was assumed to be the first one to be selected as the central point. In *cp1*'s neighborhood, *cp2* was the highest-density point and hence was chosen as the next central point. Also, *cp3* was chosen as the next central point after *cp2* and was the density peak of this region. When the peak was reached, the algorithm visited the event points around it and recorded their densities from high to low in the density ordering table. This procedure stopped when no event point was reached within the search window defined by *eps*. Moreover, the algorithm randomly selected one of the points that were not in the density ordering table as a new central point. In this way, the bars of density ordering graph usually started with a peak and went down gradually until it encountered another peak. An implicit cluster was likely to appear between the two peaks. Therefore, the density ordering graph was able to depict the density distribution of event points and even indicated the intrinsic clustering structure.

Figure 2. Network space density-based spatial clustering of applications with noise (NS-DBSCAN) algorithm reached the peak of local density from *cp1* to *cp3* (*cp3* is the local density peak).

When the density ordering graph was obtained, setting the density threshold *MinPts* accordingly led to practical clusters with the strategy of DBSCAN algorithm.

3. NS-DBSCAN Algorithm

The NS-DBSCAN algorithm is essentially the extended DBSCAN algorithm for network-constraint events. The algorithm consists of two core steps:

Step 1: Generating density ordering: In this step, the density ordering table and graph were obtained with one parameter *eps*. The step included two substeps: the first one involved obtaining *eps*-neighbors, where the LSPD algorithm is introduced. The second substep involved generating the density ordering table and graph with the densities of event points.

Step 2: Forming clusters: In this step, the second parameter *MinPts* was set according to the density ordering graph and clusters were formed by categorizing spatially adjacent and dense event points into the same cluster.

3.1. Generating Density Ordering

3.1.1. Obtaining Eps-Neighbors

The NS-DBSCAN algorithm obtained *eps*-neighbors of a central point using the LSPD algorithm. First, as depicted in Figure 3, an undirected planar graph was constructed for the network and event points in Figure 1. A new attribute, current distance to central vertex (CDCV), was assigned to each vertex, indicating the shortest-path distance of a vertex to the central vertex at any time. As interpreted in Figure 4, a basic expansion was defined as the motion from a vertex (start vertex) to its adjacent vertex (end vertex) along the edges. The path between the start vertex and the end vertex was called the expansion path. The CDCV of end vertex was updated as the sum of the CDCV of start vertex and the weight of expansion path after a basic expansion. Basic expansions aimed to minimize the CDCV

of every end vertex with the constraint of *eps*. That is, if a basic expansion failed to decrease the CDCV of an end vertex or the CDCV of an end vertex exceeded *eps*, this expansion path would be blocked. In Figure 4, if $CDCV(P_1) + W(P_1, P_2) \geq CDCV(P_2)$ or $CDCV(P_1) + W(P_1, P_2) > eps$, the expansion path $P_1 \rightarrow P_2$ would be blocked.

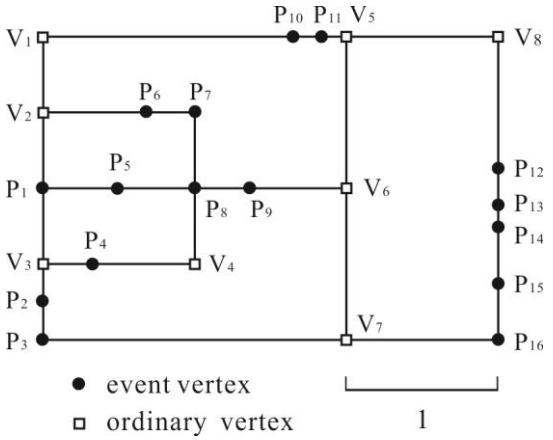

Figure 3. An undirected planar graph N = (V∪P, E, W) was generated for the simulated dataset. P is the event vertex, representing the event points. V denotes ordinary vertices, representing the location where the road segments intersect. An ordinary vertex will not be created in the segments' intersection if an event vertex already exists, such as P_1. E is the edge, representing the road segments between the two vertices. W is the weight of edges, which is defined as the length of edges in the study.

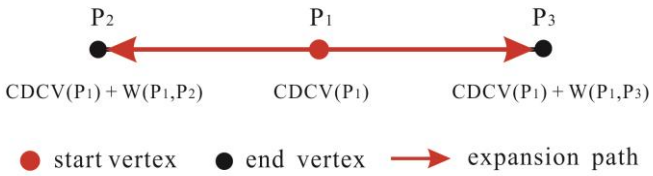

Figure 4. A basic expansion is a motion from a source (start) vertex to a target (end) vertex, the path between which is the expansion path. Current distance to central vertex (CDCV) (*p*) represents *p*'s current distance to central vertex, and W (*p*, *q*) denotes the weight (length) of expansion path between vertices *p* and *q*. The CDCV of end vertex is updated to the sum of CDCV of start vertex and the weight of expansion path between them.

The following steps explain the general steps of LSPD algorithm, and the corresponding pseudocode is presented in Table 1.

(1) Setting CDCV of central vertex to 0 and that of other vertices to ∞.
(2) Performing a basic expansion with the central vertex as the start vertex.
(3) Continuing the expansion from the new vertices, which are actually the end vertices of the last expansion.
(4) Repeating step 3 until all expansion paths are blocked. The vertices whose CDCV are neither ∞ nor 0 consist of *eps*-neighbors of the central vertex.

Table 1. Pseudocode of local shortest-path distance (LSPD) Algorithm.

Algorithm1: Local Shortest Path Distance Algorithm

Input: undirected planar graph N, central vertex *cp*, radius *eps*
Output: *cp*'s *eps*-neighbors N_*eps*(*cp*)

```
(1)   CDCV(cp) = 0, CDCV(other vertices) = ∞, cp is inserted into a newly initiated queue Q;
(2)   while Q is not empty do
(3)     p is dequeued from Q;
(4)     if p is an event vertex and not in N_eps(p) then
(5)       add p to N_eps(cp);
(6)     end if
(7)     for each vertex q adjacent to p do
(8)       if CDCV(p)+ W(p, q) < CDCV(q) and CDCV(p) + W(p, q) ≤ eps then
(9)         CDCV(q) = CDCV(p) + W(p, q);
(10)        if q is not in Q then
(11)          q is enqueued to Q;
(12)        end if
(13)      end if
(14)    end for
(15)  end while
```

In the network shown in Figure 3, *eps* = 1. P_5 was randomly selected as a central vertex. The *eps*-neighborhood of P_5 was obtained as follows. The algorithm first set CDCV(P_5) = 0 and CDCV (other vertices) = ∞. The basic expansions from P_5, that is, $P_5 \rightarrow P_1$ and $P_5 \rightarrow P_8$ took place, after which CDCV (P_1) = 0.5 and CDCV (P_8) = 0.5. P_1 and P_8 became the start vertices in the next expansion, for P_1: $P_1 \rightarrow V_2$, $P_1 \rightarrow V_3$; for P_8: $P_8 \rightarrow P_7$, $P_8 \rightarrow P_9$, and $P_8 \rightarrow V_4$. The expansion continued with V_2, V_3, P_7, P_9, and V_4 as the start vertices until all the expansion paths were blocked. The event vertices whose CDCV were neither ∞ nor 0 constituted the *eps*-neighbors of P_5, which were P_1, P_6, P_7, P_8, and P_9.

3.1.2. Generating the Density Ordering Table and Graph

The density ordering table recorded the event point, its density, and its *eps*-neighbors as follows:

$$\{Id, Density, \text{N\_}eps\}$$

where *Id* represents the identifier of event point, *Density* is its density, and N_*eps* is its *eps*-neighbors. A noteworthy phenomenon was that the density of an *eps*-neighbor was similar to its central point and it was likely to be gathered at the same cluster. Hence, if the local density peak was first visited followed by its surrounding event points, the order of event points in the density ordering table indicated implicit clusters. The density ordering graph of road network in Figure 3 is depicted in Figure 5.

The pseudocode of generating density ordering is shown in Table 2. For the simulated dataset, let *eps* = 1. An empty table was initialized. P_1 was first selected (by the order of event point identifier) and N_*eps* (P_1) = {P_5, P_2, P_4, P_8, P_3} and the queue Q ={P_1}. As Q was not empty, P_1 was dequeued from Q. P_1's density and *eps*-neighbors were written into the density ordering table to get a record {*Id*: P_1; *Density*: 5; N_*eps*: [P_5, P_2, P_4, P_8, P_3]}. N_*eps*(P_1) was inserted into Q and the density of each point in N_*eps*(P_1) was calculated. After that, the points in Q were sorted according to the order of their densities from high to low, that is Q = {P_8, P_5, P_2, P_4, P_3} and their densities were 5, 4, 3, 3, and 3, respectively. Then, the first point P_8 dequeued from Q was added to the density ordering table. Q was updated as {P_5, P_2, P_4, P_3, P_7, P_9, P_6}, where the densities of P_7, P_9, and P_6 were unknown. The densities of these points were calculated to be 4, 3, and 3, respectively, and Q was updated as {P_5, P_7, P_2, P_4, P_3, P_9, P_6}. Afterward, the first point in Q was sequentially dequeued to perform the

aforementioned operations until Q was empty. Eventually, the order of event points in the density ordering table was P_1, P_8, P_5, P_7, P_2, P_4, P_3, P_9, and P_6.

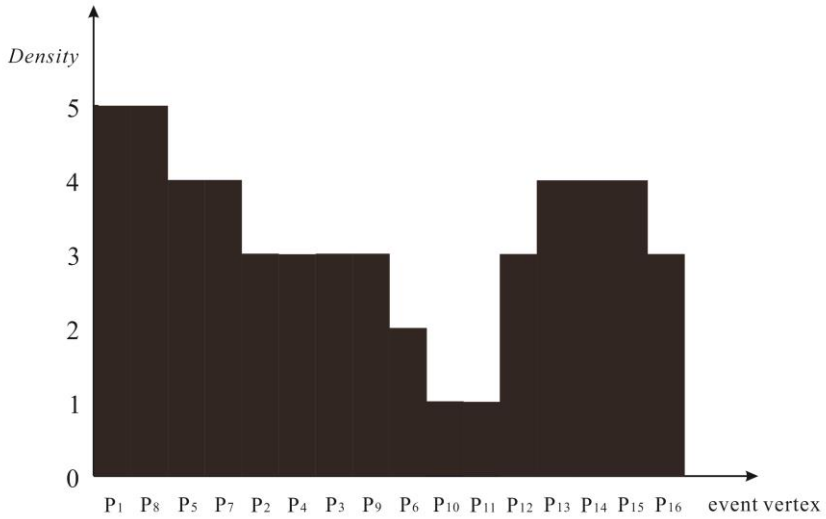

Figure 5. A density ordering graph of simulated dataset. It is a bar chart where the horizontal axis is the identifier of event points and the ordinate is the density of each event point.

Table 2. Pseudocode of Generating Density Ordering Table and Graph.

Algorithm2: Generating Density Ordering

Input: undirected planar graph N, radius *eps*
Output: density ordering table, density ordering graph

```
(1)  initialize density ordering table;
(2)  for each point p do
(3)    if p is not in density ordering table then
(4)      p is inserted into Q and get N_eps(p) by LSPD algorithm;
(5)    end if
(6)    while Q is not empty do
(7)      the first point q is dequeued from Q, write the density of q and N_eps(q) into the density
       ordering table;
(8)      for each point q in N_eps(q) do
(9)        if q's density is unknown then
(10)         calculate its density by LSPD algorithm;
(11)       end if
(12)       if q is not in density ordering table nor in Q then
(13)         add q into Q to make the densities of the points in Q are from high to low.
(14)       end if
(15)     end for
(16)   else
(17)     go to (2);
(18)   end while
(19)  draw a density ordering graph according to the density ordering table;
```

The priority queue Q determined the order of event points in the density ordering table because it ensured that the high-density event points preferentially entered the density ordering table. This made the density ordering graph appear like hills starting with local density peaks.

3.2. Forming Clusters

The density ordering table indicates implicit clusters. To explicitly obtain clusters, the density threshold *MinPts* was set according to the density ordering graph. The event points with a density greater than *MinPts* were core points, and its *eps*-neighbors were border points. The proposed algorithm constructed a cluster by initiating a cluster with a core point and gradually bringing border points of core points in the cluster until no more points were added. The pseudocode of forming clusters is shown in Table 3.

Table 3. Pseudocode of Forming Clusters.

Algorithm3: Forming Clusters

Input: density ordering table, density threshold *MinPts*
Output: density-based clusters

```
(1)   for each event point p in density ordering table do
(2)     if p does not belong to any cluster or noises then
(3)       if the density of p is less than MinPts then
(4)         mark p as noise;
(5)       else
(6)         create a new cluster C containing p;
(7)       end if
(8)       for each point q in cluster C do
(9)         if q is a core point then
(10)          for each point s in q's border points do
(11)            if s is not a member of cluster C then
(12)              add s to cluster C;
(13)            end if
(14)          end for
(15)        end if
(16)      end for
(17)    end if
(18)  end for
```

For the simulated dataset, *eps* = 1, *MinPts* = 4, and the general process of forming clusters is explained as follows. Point P_1 was first selected as a core point and did not belong to any cluster; therefore, a cluster C1 containing P_1 was initiated. The border points of P_1, {P_5, P_8, P_2, P_4, P_3}, were added into C1, as shown in Figure 6a. The next core point in C1 was P_5, and its border points {P_1, P_8, P_7, P_9} were added to C1, as shown in Figure 6b. Thereafter, the border points of P_8, {P_5, P_9, P_1, P_7, P_6}, were added into C1, as shown in Figure 6c. Next, P_2, P_4, P_3, P_7, P_9 and P_6 did not bring any other points to C1 because they were not core points. A cluster C1 = {P_1, P_5, P_2, P_4, P_8, P_3, P_7, P_9, P_6} was eventually formed. The algorithm then marked P_{10} and P_{11} as noises and assigned P_{12}–P_{16} to another cluster C2. The final clusters of the simulated dataset are shown as Figure 6d.

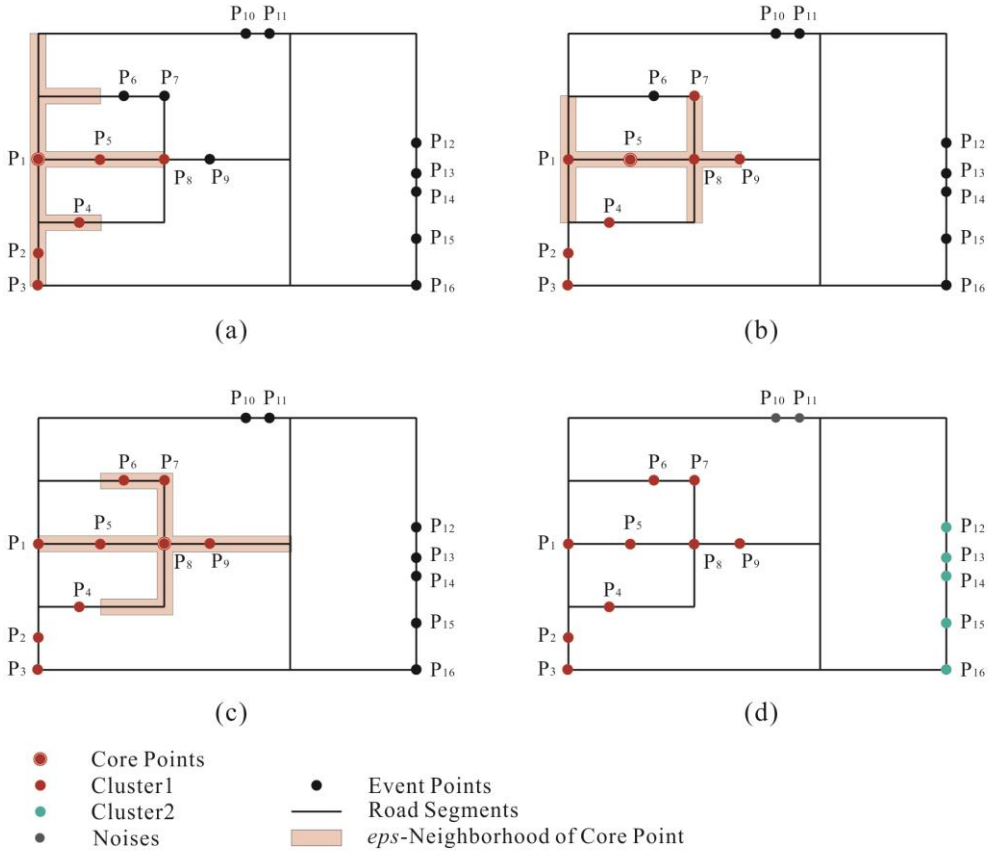

Figure 6. The core points in cluster gradually brought the border points into the cluster: (**a**) P_1 brought P_5, P_2, P_4, P_8, and P_3 to cluster 1; (**b**) P_5 brought P_7 and P_9 to cluster 1; (**c**) P_8 brought P_6 to cluster 1; (**d**) The Final description of the core points and the border points. The points of the simulated dataset eventually formed two clusters, and two points became noises.

4. Experiment

This section evaluates the effectiveness of NS-DBSCAN algorithm both qualitatively and quantitatively by testing it on the real dataset and comparing with the classical hierarchical clustering algorithm proposed by Sugihara [50] and the density-based clustering algorithm with network-constrained Delaunay triangulation (NC_DT algorithm) recently proposed by Deng [16]. Section 4.1 gives a brief introduction to the dataset used in the experiment. Section 4.2 presents the preprocessing of dataset. Experiment and comparisons of three algorithms are presented in Section 4.3.

4.1. Dataset

The Hanyang district is one of the important industrial areas in Wuhan, Hubei Province. It is a less-developed area where several large lakes are embedded. Therefore, highly populated regions are quite separated. As shown in Figure 7, the POIs in Hanyang district are mainly distributed in or around the residential areas (I & III), colleges (II), industrial parks (IV & V), and auto part shops (VI). The Wangjiawan and Longyang villages are the main residential areas, with lots of commercial housings, schools, banks, and shops. The Wuhan Technician College is an important college where thousands of technicians are trained every year. The Huangjinkou Urban Industrial Park contributes the largest share of tax among urban industrial parks in Wuhan. The Wantong Industrial Park and Shengyuan Yuanhua Industrial Park are also large industrial parks. The Huangjinkou Auto Parts Market is the largest auto parts market in central China, where more than 3000 shops sell car accessories. These highly populated regions are also regions where POIs easily form clusters.

Figure 7. Points of interest (POIs) of Hanyang district included six aggregated regions: I, Wangjiawan; II, Wuhan Technician College; III, Longyang Village; IV, Huangjinkou Urban Industrial Park and Wantong Industrial Park; V, Shengyuan Yuanhua Industrial Park; and VI, Huangjinkou Auto Parts Market.

The POI data were downloaded from Baidu Map (http://lbsyun.baidu.com/) on January 14, 2018. They included 4062 points, and were classified into 11 categories according to their functions for citizens, as shown in Figure 7. The road network dataset was downloaded from Open Street Map (https://www.openstreetmap.org) on January 9, 2018. It included 944 road segments after preprocessing.

4.2. Preprocessing

The road network (Figure 8a) and POI were preprocessed before they were used in the experiment. The preprocessing included the following four main steps:

(1) Deleting all flyovers and tunnels to make the road network planar.
(2) Extracting the skeletons of divided highways and splitting the road segments where they intersect (Figure 8b).
(3) Moving the point alongside roads to its nearest road segment to create event vertices and establish a correspondence between event vertices and point, followed by creating ordinary vertices where road segments intersect (Figure 8c).
(4) Splitting road segments at event vertices (Figure 8d).

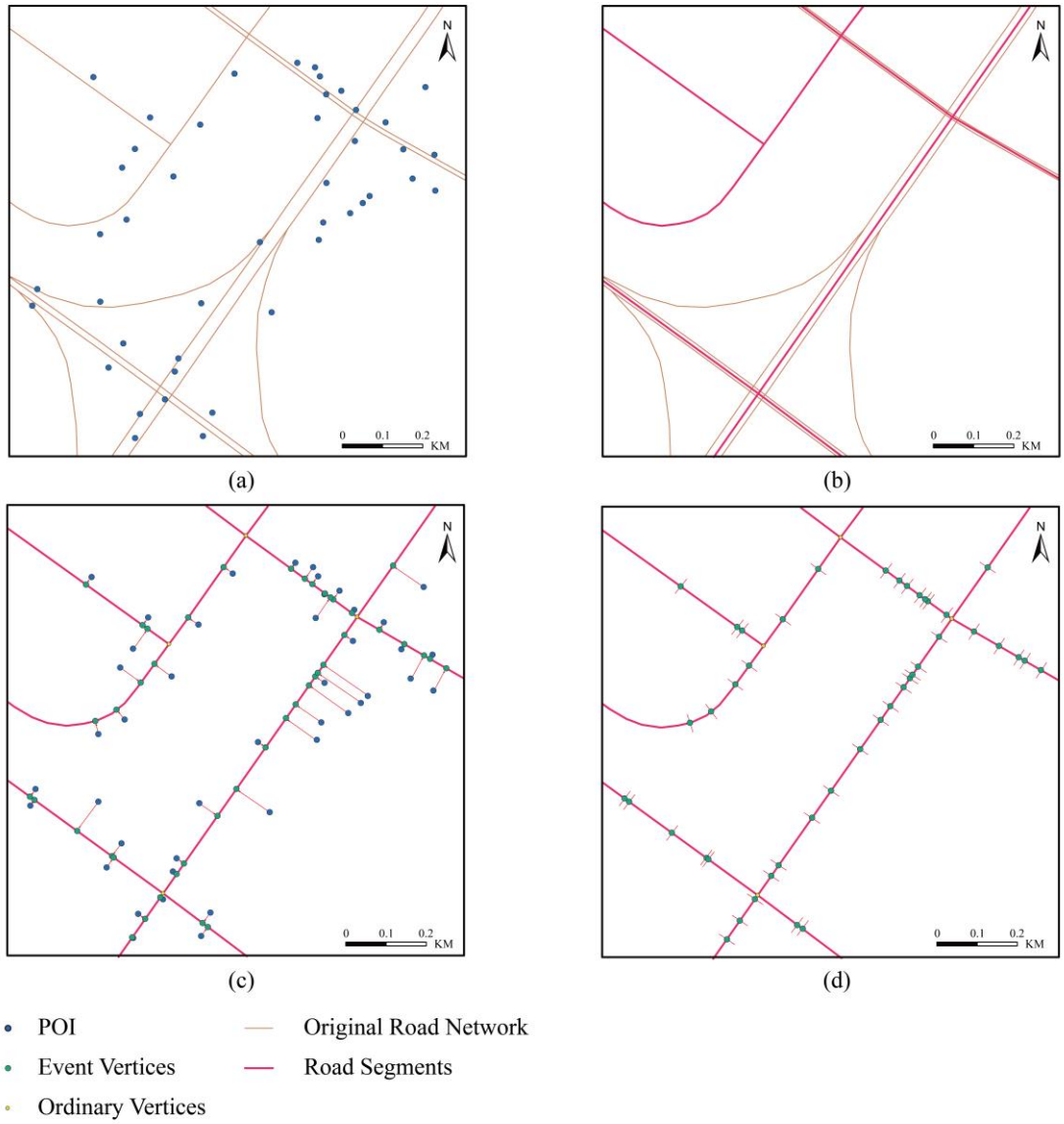

Figure 8. Steps of preprocessing: (**a**) original dataset; (**b**) extraction of skeletons; (**c**) movement of POI to the nearest road segment; and (**d**) splitting of road segments at event vertices.

4.3. Results

The three algorithms were tested in the same dataset of Hanyang district. The NS-DBSCAN and NC_DT algorithms were both implemented in ArcGIS SDK. Further, the hierarchical clusters were obtained using SANET version 4.1 [51], a plug-in program which statistically analyzes spatial patterns of events that occur on/alongside networks.

Figure 9 shows the clustering result of NS-DBSCAN algorithm. The six highly populated regions were all accurately detected and the less aggregated POIs became noise. The very high density regions (I, III, and IV) were portrayed as a large cluster, while those with subregions (II, V, and VI) were depicted with several separated clusters. The NS-DBSCAN algorithm was capable of distinguishing the separated highly populated regions, and the shape of clusters approximately portrayed the shape of these regions. In addition, some other regions with high local density were also detected and small clusters were formed there.

Figure 9. Clusters of NS-DBSCAN algorithm (*eps* = 200, *MinPts* = 20) accurately delineated the six highly populated regions of aggregated POI.

The hierarchical clustering algorithm for network-constraint events proposed by Sugihara [50] is a classical algorithm in literature. The algorithm evaluates five typical variants of distances between clusters, including the closest-pair distance (single-linkage distance in planar space), the farthest-pair distance (complete-linkage distance in planar space), the average distance (average-linkage distance in planar space), the median-pair distance (median distance in planar space), and the radius distance (centroid distance in planar space) on network events. Moreover, these distance measures were also evaluated in this study. Hierarchical clustering requires cluster number as an input parameter, which has a significant impact on the clustering result. The optimal cluster number was set as the one maximizing the silhouette (an evaluation indicator for clustering algorithms; the larger the silhouette, the better the clustering result).

Clusters of hierarchical clustering algorithm in Figure 10 delineated some high-density regions (II, III, IV, V, and VI) and the sparse POIs were also gathered to clusters. As amplified in Figure 10, region I was divided into two separate parts, although it was highly aggregated visually. The algorithm divided the dataset into several clusters according to their distances, without considering their density distribution. Although the clustering results were sensitive to the cluster number, the algorithm was still capable of delineating the density distribution of dataset as long as the cluster numbers were suitably set.

The NC_DT algorithm is a density-based algorithm for network-constraint events recently proposed by Deng [16]. The algorithm considered the road segments as areas with a specific width, not a geographical line entity. It constructed the Delaunay triangulation for all the event points and deleted the edges that were not within the road areas. The remaining triangulation was defined as network-constraint Delaunay triangulation (NC_DT), in which the shortest-path distances between event points were obtained. The major merit of this algorithm was no input parameters required. It determined neighborhood size (*eps*) by network kernel density estimation and potential entropy. Moreover, statistical tests of each event point under a null hypothesis were introduced to decide which point finally became the core point. Similar to the DBSCAN algorithm, the NC_DT algorithm obtained density-based clusters by expanding from core points and the less aggregated points finally became noises.

Figure 10. Clusters of hierarchical clustering algorithm (farthest-pair distance, cluster number = 36) basically delineated the regions of aggregated POI.

Figure 11 shows that the NC_DT algorithm did not perform effectively for this dataset; it gathered 89% of POIs into a giant cluster and many small clusters appeared mostly in the less-dense regions. It was not suitable for the dataset because the distances between some event points might be exaggerated due to the disconnectedness of NC_DT. As a consequence, the parameter determined by the statistical method based on the distances of event points might be invalid. For example, the POIs in the amplified region of Figure 11 should be in the same cluster. However, they were separated into two different clusters because the points in different clusters were not connected by NC_DT. In fact, most small clusters were formed where NC_DT was disconnected. A possible remedy is to add all the interaction and inflection points of road segments into a distance matrix and distinguish these points from event points while clustering. However, this inevitably adds to the high time and space complexity of algorithm. Therefore, the NC_DT algorithm was recommended in very dense event points, in which case the effect of disconnected NC_DT could be largely avoided.

The quantitative evaluation of the three clustering algorithms is listed in Table 4. Four indicators, including the silhouette [25], the R-squared index (RS) [58], the Davis–Bouldin index (DB) [59], and the clustering scheme quality index (SD) [60], were used to evaluate the effectiveness of clustering algorithms. All the indicators showed that the NC_DT algorithm was not effective for the dataset. Among the five distance measures of hierarchical algorithm, the closest-pair distance performed badly, and the rest were acceptable. The NS-DBSCAN algorithm obtained several clustering results with different groups of input parameters. Although not all the input parameters led to a good clustering result, all the indicators showed that the clustering result was relatively better compared with the other two algorithms in general.

Figure 11. Clusters of NC_DT algorithm does not work effectively for the dataset.

Table 4. Indicators for Evaluating the Effectiveness of Clustering Algorithms.

Clustering Algorithms	Parameters	Cluster Number	silhouette	RS	DB	SD
NC_DT Algorithm		37	−0.5054	0.2215	0.8167	0.0364
Heirarchical Algorithm	closest-pair distance	10	0.0164	0.5517	0.4878	0.0887
	farthest-pair distance	36	0.3949	0.9739	0.9251	0.0264
	average-pair distance	10	0.4466	0.9031	0.7822	0.0866
	median-pair distance	23	0.3420	0.9446	0.8121	0.0393
	radius distance	16	0.4292	0.9426	0.6821	0.0518
NS-DBSCAN Algorithm	*eps* = 100, *MinPts* = 10	60	0.4470	0.9968	0.4047	0.0131
	eps = 100, *MinPts* = 15	38	0.4296	0.9978	0.5004	0.0166
	eps = 100, *MinPts* = 20	21	0.6740	0.9999	0.3605	0.0180
	eps = 200, *MinPts* = 15	38	0.2002	0.9719	0.5375	0.0095
	eps = 200, *MinPts* = 20	31	0.4187	0.9913	0.4407	0.0088
	eps = 200, *MinPts* = 25	28	0.4703	0.9952	0.4456	0.0113
	eps = 300, *MinPts* = 20	24	0.2902	0.9628	0.5473	0.0145
	eps = 300, *MinPts* = 25	23	0.2608	0.9666	0.5213	0.0121
	eps = 300, *MinPts* = 30	20	0.3220	0.9681	0.4860	0.0119
	eps = 400, *MinPts* = 25	14	0.3424	0.9307	0.5812	0.0262
	eps = 400, *MinPts* = 30	16	0.3560	0.9564	0.5610	0.0194
	eps = 400, *MinPts* = 35	17	0.3777	0.9625	0.5098	0.0158

[1] For silhouette and RS, larger is better; For DB and SD, smaller is better.

5. Discussion

5.1. Parameterization

To obtain satisfactory clusters, two input parameters *eps* and *MinPts* need to be carefully set, and the density ordering graph can be the guidance. The density ordering graph exhibited some characteristics. First, it was able to delineate the overall density distribution of event points and indicate implicit clusters. For example, with the dataset used in the experiment, Figure 12 depicts the density ordering graphs of four situations (*eps* = 100, 200, 300, and 400). The hills within the orange rectangle in Figure 12b delineate Wangjiawan (region I in Figure 7), a highly populated region where clusters were most likely to appear because the densities of spatially adjacent event points were quite

high. Second, with the increase in *eps*, the density ordering graph lost more details about density distribution, and large clusters were more likely to come out. Finally, the hills passed through by the line at *MinPts* are likely to form clusters. Besides, the number of clusters approximately equaled the number of hills crossed by *MinPts*. For example, the red line in Figure 12b passed through 27 hills, and 31 clusters came into being eventually, as shown in Figure 9.

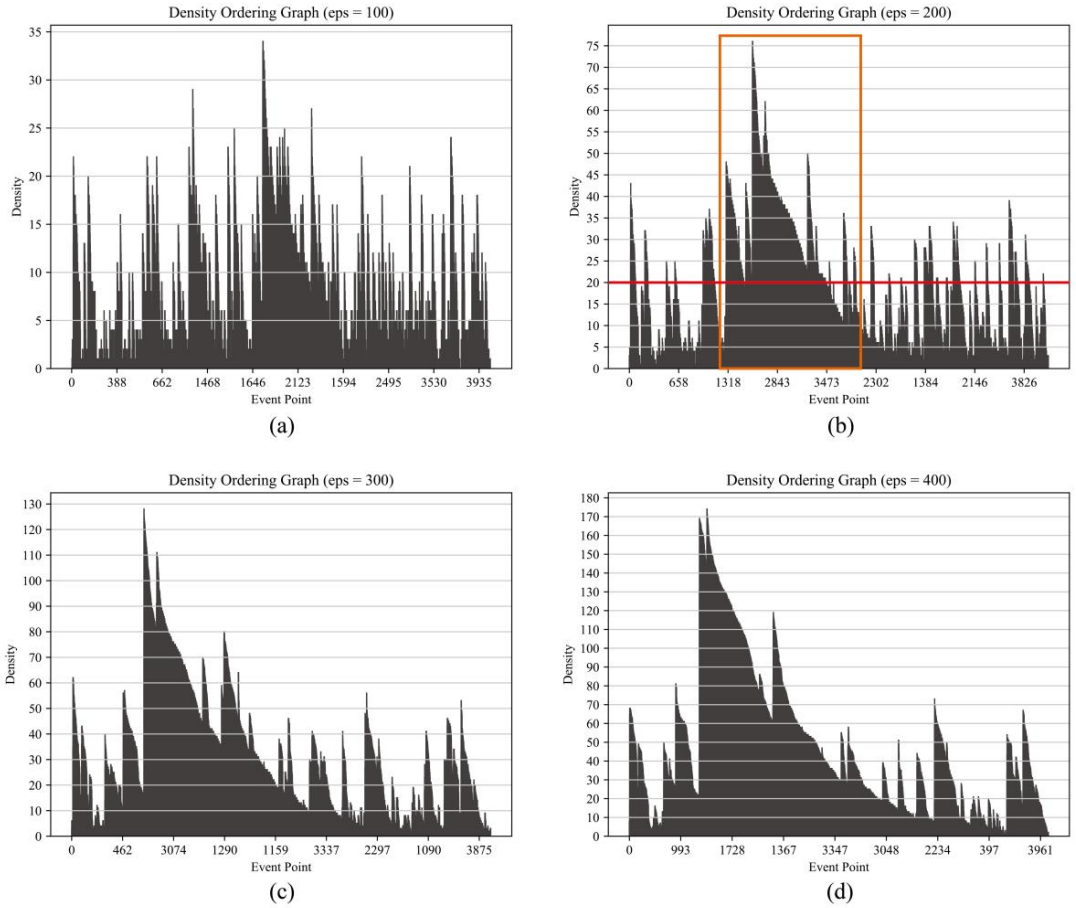

Figure 12. Density ordering graphs of (**a**) *eps* = 100; (**b**) *eps* = 200; (**c**) *eps* = 300; and (**d**) *eps* = 400.

Based on the characteristics of density ordering graph, the parameter setting followed the basic principles. First, appropriate *eps* should be set to get a density ordering graph with hills of medium size. This helped to obtain medium-sized clusters. For example, the hills in Figure 10b were preferable than those in (a) (too slim) and (c) and (d) (too fat). Second, *MinPts* was set below the peaks where clusters were expected. Setting two input parameters in accordance with the aforementioned principle finally led to a relatively practical clustering result.

5.2. One-way and Dead-End Cases

All the paths of road network were assumed to be bidirectional in this study. However, one-way roads sometimes exist in a city's road network and they should be considered while clustering. In the LSPD algorithm, if an expansion path is unidirectional, it is blocked when it goes backward. For example, in Figure 4, the expansion path $P_1 \rightarrow P_2$ is blocked if the path from P_2 to P_1 is unidirectional.

Dead-end roads do not have any impact on the study. In a real dataset, dead-end roads do not intersect with other roads on one side or on both sides. Figure 13 demonstrates the dead-end side of a road and the basic expansion from P_1 to the dead-end vertex P_2 (usually an ordinary vertex). CDCV (P_2) was updated to CDCV (P_1) + W (P_1, P_2), and hence was greater than CDCV (P_1). P_2 became the

next start vertex and the basic expansion from it went nowhere because the expansion path from P_2 to P_1 was blocked. In this situation, no other vertices were influenced by the dead-end vertex.

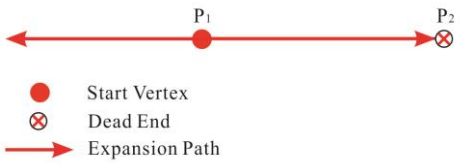

Figure 13. A basic expansion in a dead-end road segment.

6. Conclusions

This study extended the DBSCAN algorithm to network events and proposed a new clustering algorithm named NS-DBSCAN. Although two input parameters were still required, a density ordering graph was provided as guidance. An experiment in a real dataset was conducted to evaluate the effectiveness of the proposed algorithm, and it accurately detected highly populated communities. Compared with the hierarchical algorithm and NC_DT algorithm, NS-DBSCAN algorithm could better delineate the density distribution of the dataset. Besides, the four indicators showed that NS-DBSCAN algorithm, in general, performed better than two compared algorithms.

The proposed algorithm was important because of the following reasons: First, it concentrated on clustering network-constraint events, which were very common in city environments. Second, it provided an efficient algorithm (LSPD algorithm) to obtain *eps*-neighbors of event points without constructing a distance matrix. Finally, it provided a new visualization of the density distribution of spatial point events, and it could be the guidance of segmentation threshold parameterization. The proposed visual parameterization was less time-consuming than those extending DBSCAN with statistical indicators. Since clustering results should meet different needs in real-life application, the proposed visualization allowed users to obtain suitable clustering results graphically.

Future research on NS-DBSCAN should take the following issues into consideration. First, each hill exceeding *MinPts* in the density ordering graph, in general, should correspond to one cluster. However, such correspondence is not strict. Therefore, a more practical mechanism for ordering event points to increase the certainty of one-to-one correspondence between hills and clusters can be explored. Second, users may have to try different *eps* to get a suitable graph. The *eps* parameter is expected to be determined heuristically or statistically in future studies.

Author Contributions: Conceptualization, Tianfu Wang and Jing Tian; methodology, Tianfu Wang, Yun Luo; software, Tianfu Wang, Yun Luo; validation, Tianfu Wang, Chang Ren and Jing Tian; formal analysis, Tianfu Wang; investigation, Tianfu Wang; resources, Jing Tian; data curation, Tianfu Wang; writing—original draft preparation, Tianfu Wang; writing—review and editing, Jing Tian, and Chang Ren Yun Luo; visualization, Tianfu Wang; supervision, Jing Tian; project administration, Jing Tian.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 41701439. The APC was funded by the National Natural Science Foundation of China.

Acknowledgments: We would like to give my heartfelt gratefulness to Jiajia Liu and Yimin Huang. They kindly provided the material, without which the work could not have been finished. Besides, the pertinent suggestions of Weijun Yin, Zhiying Zhou and Guan Luo are really helpful when we were trying to improve the methodology and we do appreciate it.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tobler, W.R. A Computer Movie Simulating Urban Growth in the Detroit Region. *Econ. Geogr.* **1970**, *46*, 234–240. [CrossRef]
2. Waller, L.A. Detection of clustering in spatial data. In *The SAGE Handbook of Spatial Analysis*; SAGE Publications: Thousand Oaks, CA, USA, 2009; pp. 299–320.

3. Estivill-Castro, V.; Lee, I. Multi-level clustering and its visualization for exploratory spatial analysis. *Geoinformatica* **2002**, *6*, 123–152. [CrossRef]
4. Shiode, S. Street-level Spatial Scan Statistic and STAC for Analysing Street Crime Concentrations. *Trans. GIS* **2011**, *15*, 365–383. [CrossRef]
5. He, L.; Páez, A.; Liu, D. Persistence of Crime Hot Spots: An Ordered Probit Analysis. *Geogr. Anal.* **2017**, *49*, 3–22. [CrossRef]
6. Guo, D.; Zhu, X.; Jin, H.; Gao, P.; Andris, C. Discovering Spatial Patterns in Origin-Destination Mobility Data. *Trans. GIS* **2012**, *16*, 411–429. [CrossRef]
7. Chen, Y.; Qin, K.; Wang, Y.; Zhao, P.; Ye, X. A trajectory clustering approach based on decision graph and data field for detecting hotspots. *Int. J. Geogr. Inf. Sci.* **2016**, *31*, 1101–1127. [CrossRef]
8. Pei, T.; Wang, W.; Zhang, H.; Ma, T.; Du, Y.; Zhou, C. Density-based clustering for data containing two types of points. *Int. J. Geogr. Inf. Sci.* **2015**, *29*, 175–193. [CrossRef]
9. Yamada, I.; Thill, J.C. Local Indicators of Network—Constrained Clusters in Spatial Point Patterns. *Geogr. Anal.* **2007**, *39*, 268–292. [CrossRef]
10. Nie, K.; Wang, Z.; Du, Q.; Ren, F.; Tian, Q. A network-constrained integrated method for detecting spatial cluster and risk location of traffic crash: A case study from Wuhan, China. *Sustainability* **2015**, *7*, 2662–2677. [CrossRef]
11. Liu, Q.; Deng, M.; Shi, Y.; Wang, J. A density-based spatial clustering algorithm considering both spatial proximity and attribute similarity. *Comput. Geosci.* **2012**, *46*, 296–309. [CrossRef]
12. Nojarov, P. Genetic climatic regionalization of the Balkan Peninsula using cluster analysis. *J. Geogr. Sci.* **2017**, *27*, 43–61. [CrossRef]
13. Pei, T.; Jasra, A.; Hand, D.J.; Zhu, A.X.; Zhou, C. DECODE: A new method for discovering clusters of different densities in spatial data. *Data Min. Knowl. Discov.* **2009**, *18*, 337–369. [CrossRef]
14. Deng, M.; Liu, Q.; Cheng, T.; Shi, Y. An adaptive spatial clustering algorithm based on delaunay triangulation. *Comput. Environ. Urban Syst.* **2011**, *35*, 320–332. [CrossRef]
15. Liu, Q.; Tang, J.; Deng, M.; Shi, Y. An iterative detection and removal method for detecting spatial clusters of different densities. *Trans. GIS* **2015**, *19*, 82–106. [CrossRef]
16. Deng, M.; Yang, X.; Shi, Y.; Gong, J.; Liu, Y.; Liu, H. A density-based approach for detecting network-constrained clusters in spatial point events. *Int. J. Geogr. Inf. Sci.* **2018**, *33*, 466–488. [CrossRef]
17. Okabe, A.; Yamada, I. The K-Function Method on a Network and its computational implementation. *Geogr. Anal.* **2001**, *33*, 270–290. [CrossRef]
18. Liu, Q.; Tang, J.; Deng, M.; Shi, Y.; Li, Z.; Liu, Q.; Tang, J.; Deng, M. An adaptive method for clustering spatio-temporal events. *Trans. GIS* **2018**, *22*, 82–106. [CrossRef]
19. Anselin, L. Local indicators of spatial analysis—LISA. *Geogr. Anal.* **1995**, *27*, 93–115. [CrossRef]
20. Jackson, M.C.; Huang, L.; Xie, Q.; Tiwari, R.C. A modified version of Moran's I. *Int. J. Health Geogr.* **2010**, *9*. [CrossRef] [PubMed]
21. Martin, K.; Neville, N. Spatial disease clusters: Detection and inference. *Stat. Med.* **2018**, *14*, 799–810. [CrossRef]
22. Rey, S.J. Space-time patterns of rank concordance: Local indicators of mobility association with application to spatial income inequality dynamics. *Ann. Am. Assoc. Geogr.* **2016**, *106*, 788–803. [CrossRef]
23. Fan, Y.; Zhu, X.; She, B.; Guo, W.; Guo, T. Network-constrained spatio-temporal clustering analysis of traffic collisions in Jianghan District of Wuhan, China. *PLoS ONE* **2018**, *13*, e0195093. [CrossRef]
24. Han, J.; Kamber, M.; Pei, J. *Data Mining: Concept and Techniques*, 2nd ed.; Elsevier Pte Ltd.: Singapore, 2012.
25. Kaufman, L.; Rousseeuw, P.J. *Finding Groups in Data: An. Introduction to Cluster Anaalysis*; Wiley: London, UK, 1990.
26. Ng, R.T.; Han, J. Efficient and effective clustering methods for spatial data mining. In Proceedings of the VLDB Conference, Santiago, Chile, 12–15 September 1994.
27. Zhang, T.; Ramakrishnan, R.; Livny, M. BIRCH: An efficient data culatering method for very large databases. In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, QC, Canada, 4 –6 June 1996; Volume 1, pp. 103–114.
28. Guha, S.; Rastogi, R.; Shim, K. CURE: An Efficient Clustering Algorithm for Large Databases. In *ACM SIGMOD Record*; Elsevier: Seattle, WA, USA, 1998; Volume 27, pp. 73–84.

29. Karypis, G.; Han, E.H.; Kumar, V. Chameleon: Hierarchical clustering using dynamic modeling. *Computer* **1999**, *32*, 68–75. [CrossRef]
30. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
31. Ankerst, M.; Breunig, M.M.; Kriegel, H.-P.; Sander, J. OPTICS: Ordering Points to Identify the Clustering Structure. In *ACM SIGMOD'99 International Conference on Management of Data*; ACM: New York, NY, USA, 1999; pp. 49–60.
32. Mai, G.; Janowicz, K.; Hu, Y.; Gao, S. ADCN: An anisotropic density-based clustering algorithm for discovering spatial point patterns with noise. *Trans. GIS* **2018**, *22*, 348–369. [CrossRef]
33. Deng, M.; Liu, Q.; Li, G.; Cheng, T. Field-theory based spatial clustering method. *J. Remote Sens.* **2010**, *14*, 694–709.
34. Marek, L.; Pászto, V.; Tucek, P. Using clustering in geosciences: Examples and case studies. In Proceedings of the 15th International Multidisciplinary Scientific GeoConference-SGEM, Albena, Bulgaria, 18–24 June 2015.
35. Park, H.S.; Jun, C.H. A simple and fast algorithm for K-medoids clustering. *Expert Syst. Appl.* **2009**, *36*, 3336–3341. [CrossRef]
36. Lucasius, C.B.; Dane, A.D.; Kateman, G. On k-medoid clustering of large data sets with the aid of a genetic algorithm: Background, feasiblity and comparison. *Anal. Chim. Acta* **1993**, *282*, 647–669. [CrossRef]
37. van der Laan, M.J.; Pollard, K.S. A New Partitioning Around Medoids Algorithm. *Biostatistics* **2002**, *73*, 575–584. [CrossRef]
38. Liu, Q.; Deng, M.; Shi, Y. Adaptive spatial clustering in the presence of obstacles and facilitators. *Comput. Geosci.* **2013**, *56*, 104–118. [CrossRef]
39. Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Min. Knowl. Discov.* **1998**, *2*, 169–194. [CrossRef]
40. Wang, X.; Hamilton, H.J. DBRS: A Density-Based Spatial Clustering Method with Random Sampling. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2637, pp. 563–575.
41. Wang, B.; Wang, X. Spatial entropy-based clustering for mining data with spatial correlation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin/Heidelberg, Germany, 2011.
42. Yamada, I.; Thill, J.C. Comparison of planar and network K-functions in traffic accident analysis. *J. Transp. Geogr.* **2004**, *12*, 149–158. [CrossRef]
43. Okabe, A.; Boots, B.; Sugihara, K. Nearest neighbourhood operations with generalized voronoi diagrams: A review. *Int. J. Geogr. Inf. Syst.* **1994**, *8*, 43–71. [CrossRef]
44. Erwig, M. The graph Voronoi diagram with applications. *Networks* **2000**, *36*, 156–163. [CrossRef]
45. Okabe, A.; Yomono, H.; Kitamura, M. Statistical Analysis of the Distribution of Points on a Network. *Geogr. Anal.* **1995**, *27*, 152–175. [CrossRef]
46. Flahaut, B.; Mouchart, M.; Martin, E.S.; Thomas, I. The local spatial autocorrelation and the kernel method for identifying black zones: A comparative approach. *Accid. Anal. Prev.* **2003**, *35*, 991–1004. [CrossRef]
47. Whiteaker, T.L.; Maidment, D.R.; Gopalan, H.; Patino, C.; McKinney, D.C. Raster-network regionalization for watershed data processing. *Int. J. Geogr. Inf. Sci.* 2007. [CrossRef]
48. Tong, D.; Murray, A.T. Spatial Optimization in Geography. *Ann. Assoc. Am. Geogr.* **2012**, *102*, 1290–1309. [CrossRef]
49. Yiu, M.L.; Mamoulis, N. Clustering Objects on a Spatial Network. In *SIGMOD Conference*; ACM: Paris, France, 2004; pp. 443–454.
50. Sugihara, K.; Okabe, A.; Satoh, T. Computational method for the point cluster analysis on networks. *Geoinformatica* **2011**, *15*, 167–189. [CrossRef]
51. Okabe, A.; Okunuki, K.; Shiode, S. A Toolbox for Spatial Analysis on a Network. *GIS Based Stud.* **2005**, *38*, 57–66. [CrossRef]
52. Stefanakis, E. NET-DBSCAN: Clustering the nodes of a dynamic linear network. *Int. J. Geogr. Inf. Sci.* **2007**, *21*, 427–442. [CrossRef]
53. Chen, J.; Lai, C.; Meng, X.; Xu, J.; Hu, H. Clustering Moving Objects in Spatial Networks. In Proceedings of the 12th International Conference on Database Systems for Advanced Applications, Bangkok, Thailand, 9–12 April 2007; pp. 611–623.

54. Shi, Y.; Deng, M.; Gong, J.; Lu, C.-T.; Yang, X.; Liu, H. Detection of clusters in traffic networks based on spatio-temporal flow modeling. *Trans. GIS* **2019**, *23*, 312–333. [CrossRef]
55. Oliveira, D.; Garrett, J.; Soibelman, L. Spatial clustering analysis of water main break events. In Proceedings of the International Workshop on Computing in Civil Engineering 2009, Austin, TX, USA, 24–27 June 2009; pp. 338–347. [CrossRef]
56. Oliveira, D.; Garrett, J.; Soibelman, L. A density-based spatial clustering approach for defining local indicators of drinking water distribution pipe breakage. *Adv. Eng. Inform.* **2011**, *25*, 380–389. [CrossRef]
57. Smaltschinski, T.; Seeling, U.; Becker, G. Clustering forest harvest stands on spatial networks for optimised harvest scheduling. *Ann. For. Sci.* **2012**, *69*, 651–657. [CrossRef]
58. Halkidi, M.; Batistakis, Y.; Vazirgiannis, M. Cluster Validity Methods: Part I. *SIGMOD Rec.* **2002**, *31*, 40–45. [CrossRef]
59. Davies, D.L.; Bouldin, D.W. A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Mach. Intell.* **1979**, *PAMI-1*, 224–227. [CrossRef]
60. Halkidi, M.; Vazirgiannis, M.; Batistakis, Y.; Ri, H.S.W.; Wkhqv, Q.; Frqrplfv, R.I.; Hoodv, W.U.; Pyd, P.; Dqqlv, L.U.J.; Ju, D. Quality scheme assessment in the clustering process. In *Principles of Data Mining and Knowledge Discovery*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 265–276.