US 20100179759A1

(19) **United States**

(12) **Patent Application Publication**

Zheng et al.

(10) **Pub. No.: US 2010/0179759 A1**

(43) **Pub. Date: Jul. 15, 2010**

(54) **DETECTING SPATIAL OUTLIERS IN A LOCATION ENTITY DATASET**

(75) Inventors: **Yu Zheng**, Beijing (CN); **Jianqiao Feng**, Beijing (CN); **Xing Xie**, Beijing (CN); **Wei-Ying Ma**, Beijing (CN)

Correspondence Address:
LEE & HAYES, PLLC
601 W. RIVERSIDE AVENUE, SUITE 1400
SPOKANE, WA 99201 (US)

(73) Assignee: **MICROSOFT CORPORATION**, Redmond, WA (US)

(21) Appl. No.: **12/353,940**

(22) Filed: **Jan. 14, 2009**

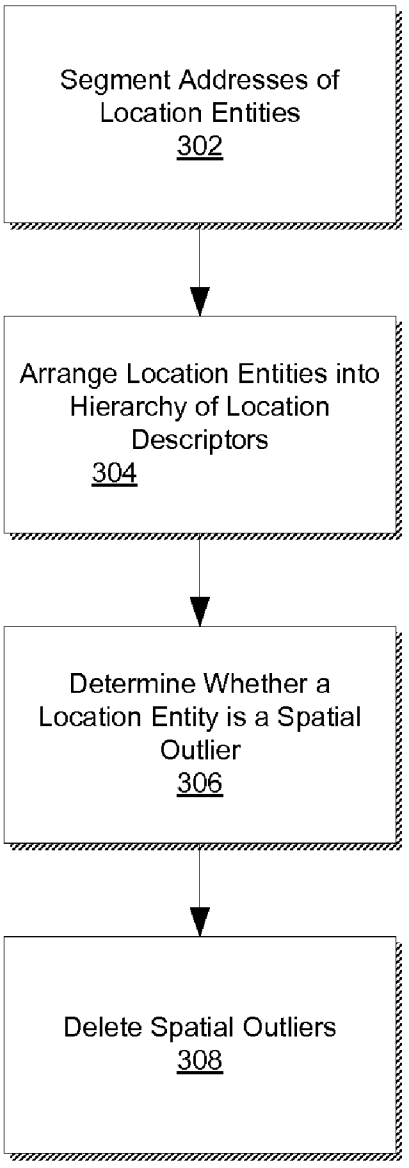
Publication Classification

(51) **Int. Cl.**
G06F 17/00 (2006.01)

(52) **U.S. Cl.** **701/300**

(57) **ABSTRACT**

Disclosed herein are one or more embodiments that arrange a plurality of location entities into a hierarchy of location descriptors. One or more of the disclosed embodiments may determine whether one of the location entities is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity. Also, the other location entities and the one location entity may share a location descriptor.



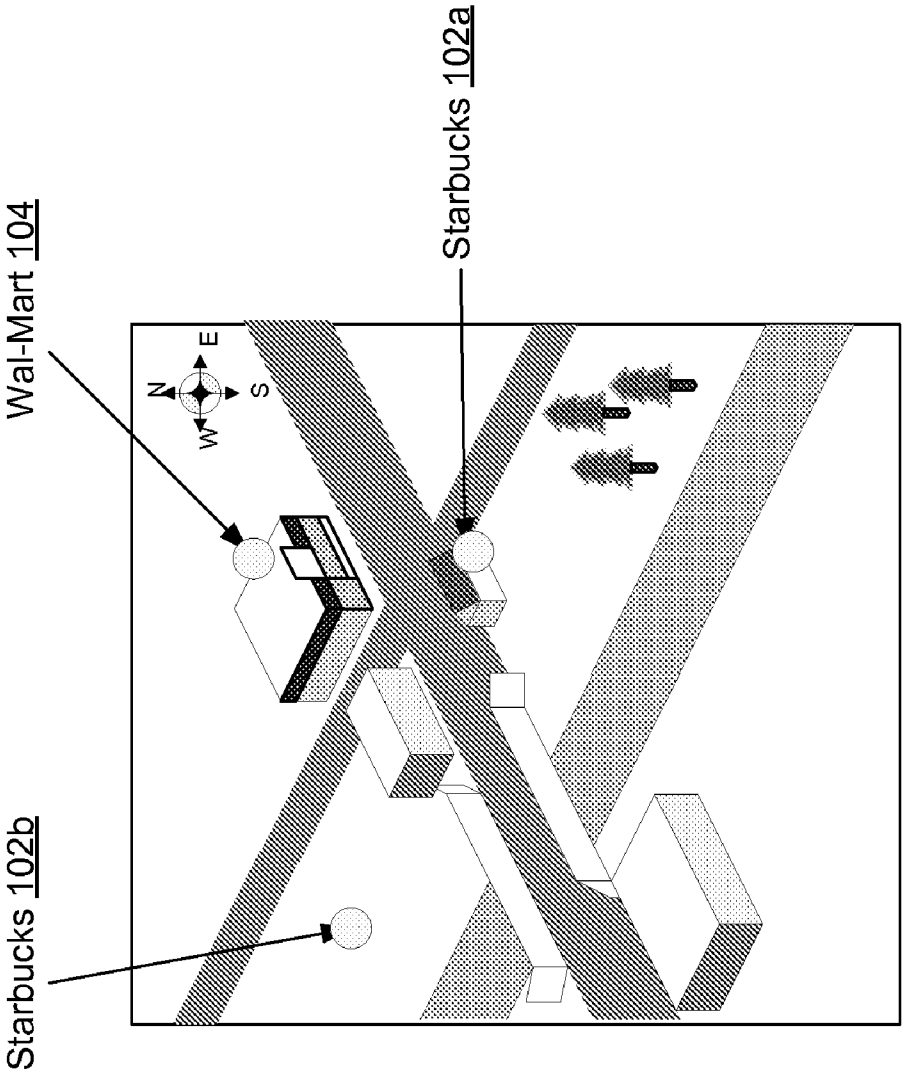


Fig. 1

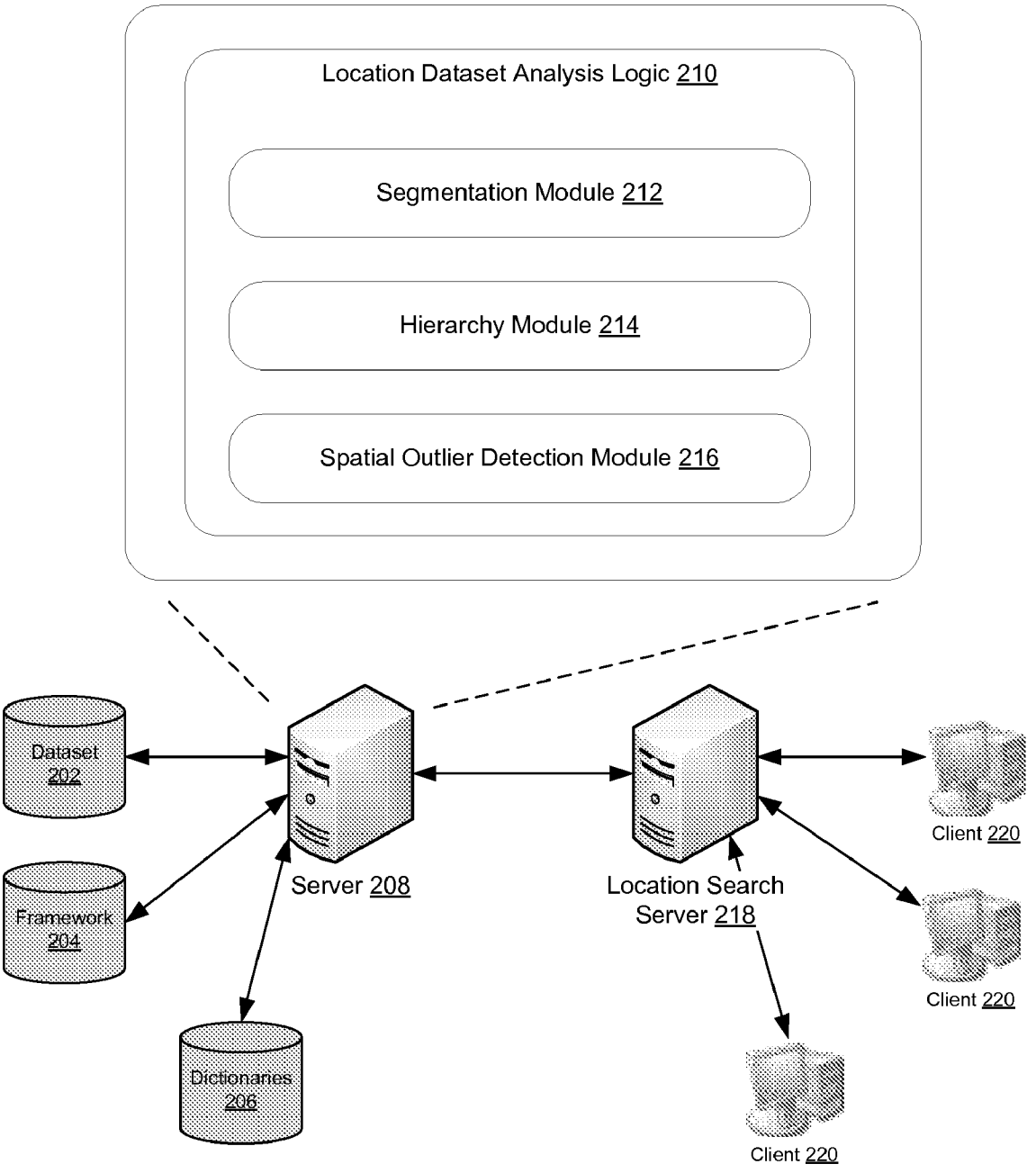

Location Dataset Analysis Logic 210
Segmentation Module 212
Hierarchy Module 214
Spatial Outlier Detection Module 216
Dataset 202
Framework 204
Dictionaries 206
Server 208
Location Search Server 218
Client 220
Client 220
Client 220

Fig. 2

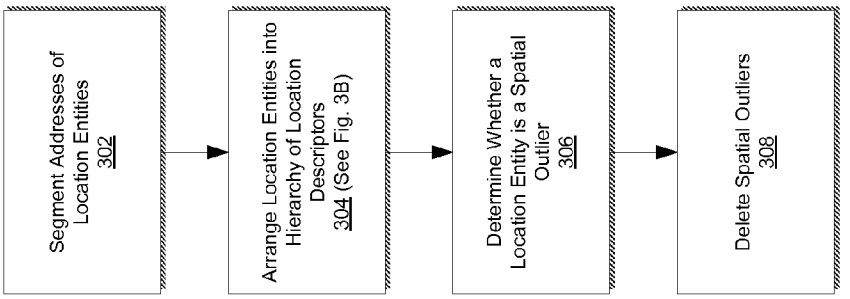

Segment Addresses of Location Entities 302
Arrange Location Entities into Hierarchy of Location Descriptors 304 (See Fig. 3B)
Determine Whether a Location Entity is a Spatial Outlier 306
Delete Spatial Outliers 308

Fig. 3a

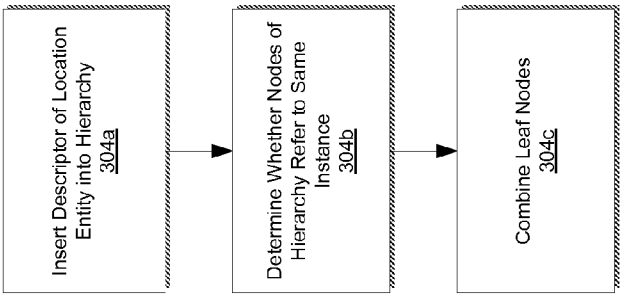

Insert Descriptor of Location Entity into Hierarchy 304a
Determine Whether Nodes of Hierarchy Refer to Same Instance 304b
Combine Leaf Nodes 304c

Fig. 3b

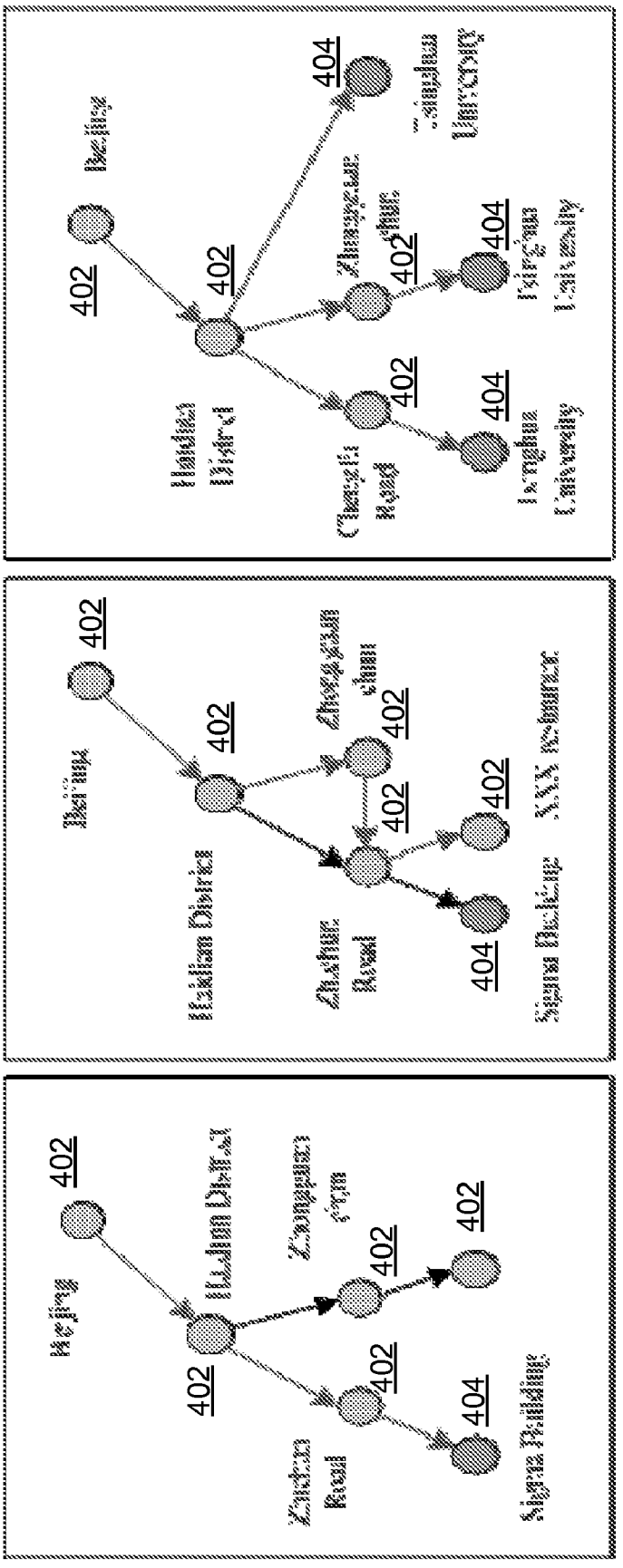

402
404
Beijing
Haidian District
Zhongguancun Road
Tsinghua University
Peking University
Beijing
Haidian District
Zhongguancun Road
Xxx Avenue
Xxx restaurant
Beijing
Haidian District
Zhongguancun Road
Sigma Building
A) Building a multi-query link
B) Building a link between users
C) Users the same landmark after multiple nodes of the framework

Fig. 4

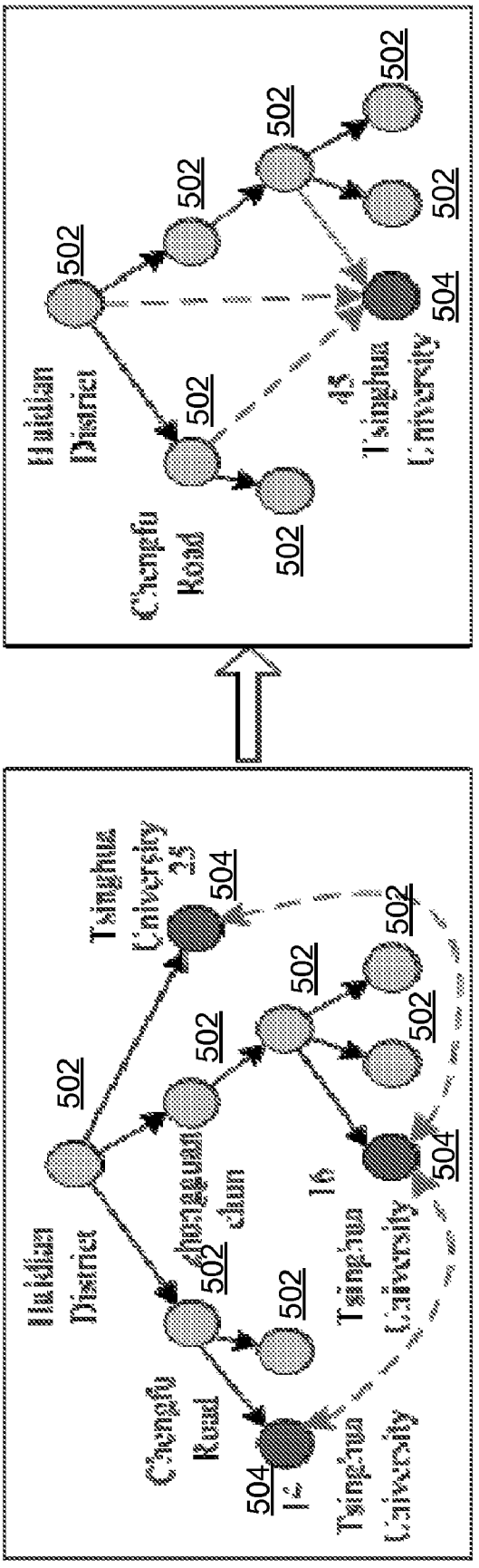
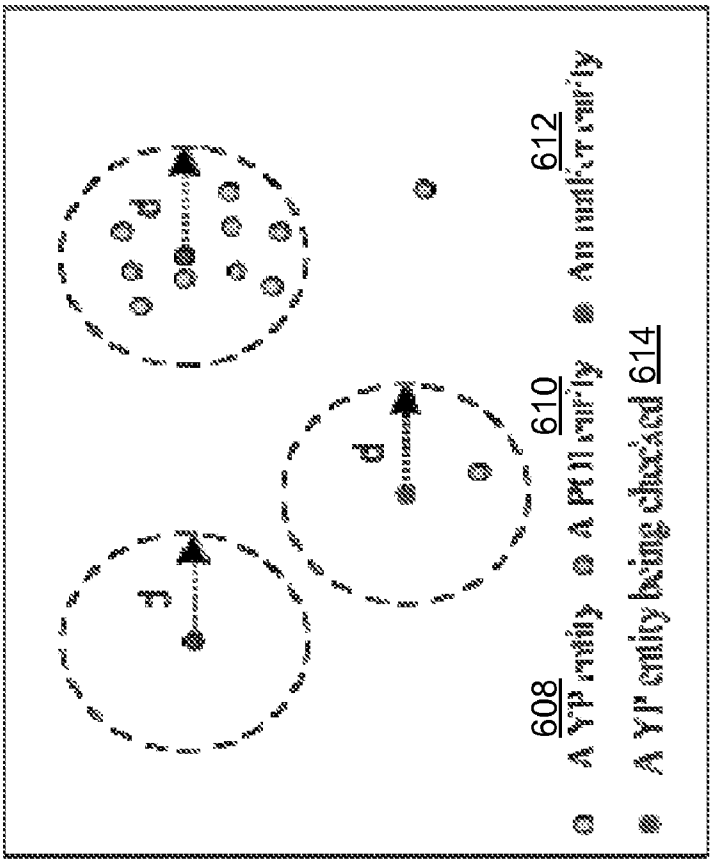

Haidian District
Zhongguancun
Tsinghua University
Renmin University
502
504

Fig. 5


A YP entity 608
A YP entity being checked 614
A PM entity 610
An outlier entity 612
B) PM-based algorithm

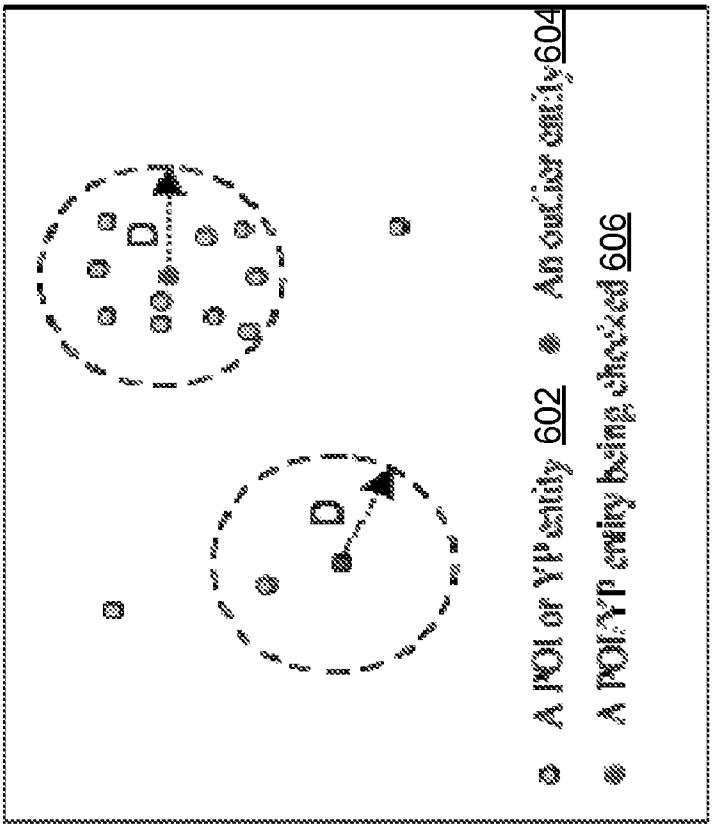

A NOI or YP entity 602
An outlier entity 604
A NOI/YP entity being checked 606
A) NP algorithm

Fig. 6

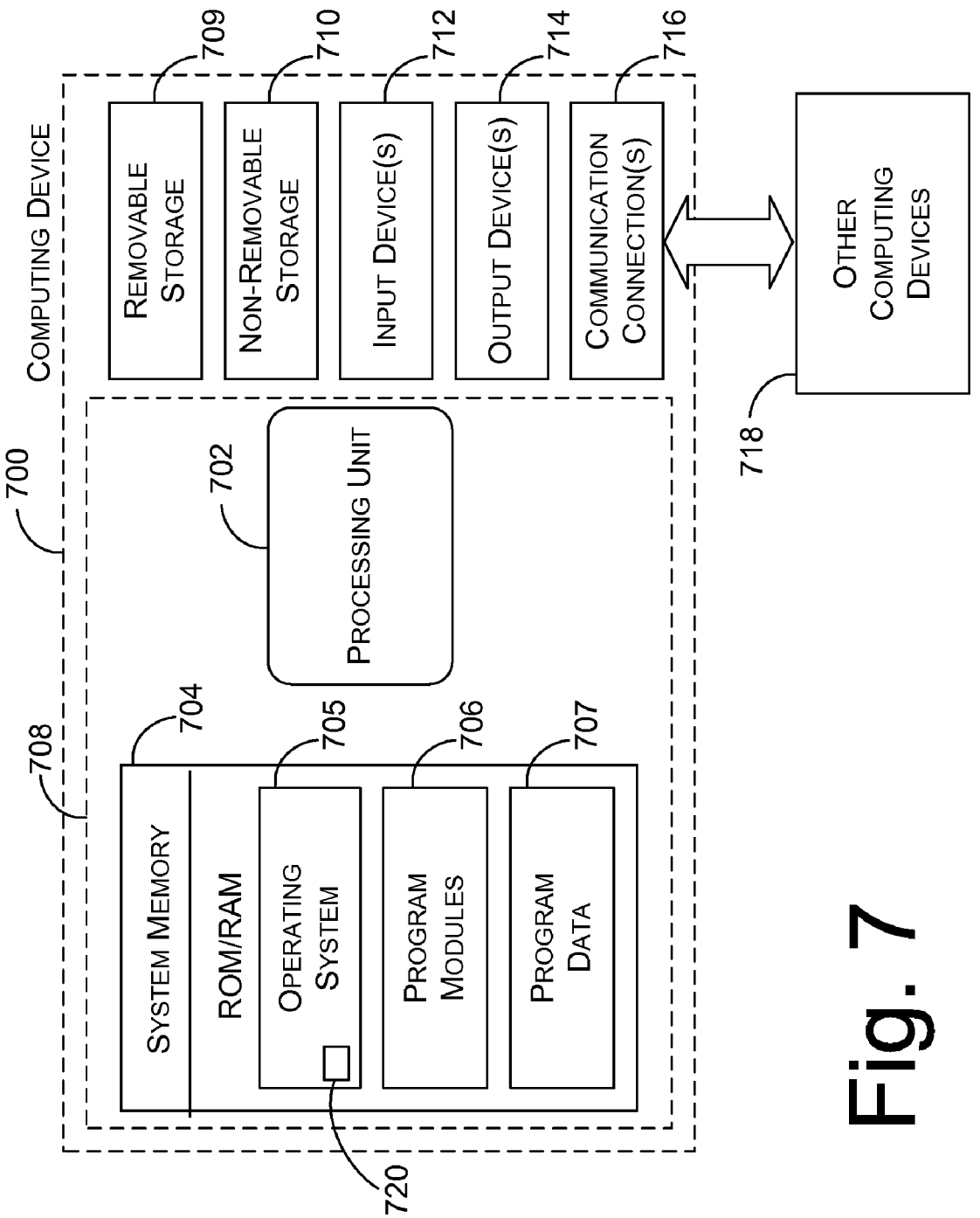

COMPUTING DEVICE
700
REMOVABLE STORAGE
709
NON-REMOVABLE STORAGE
710
INPUT DEVICE(S)
712
OUTPUT DEVICE(S)
714
COMMUNICATION CONNECTION(S)
716
OTHER COMPUTING DEVICES
718
PROCESSING UNIT
702
708
SYSTEM MEMORY
704
ROM/RAM
OPERATING SYSTEM
705
PROGRAM MODULES
706
PROGRAM DATA
707
720

Fig. 7

DETECTING SPATIAL OUTLIERS IN A LOCATION ENTITY DATASET

BACKGROUND

[0001] With the wide availability of wireless and satellite connections to online services, users are increasingly relying on location search services to find destinations. Services such as Windows Local Live Search™ often provide users with traditional maps of locations, aerial photographs of those same locations, and/or combinations of photographs and maps.

[0002] In addition, the services often annotate these maps and photographs with identifiers for landmarks, businesses, and/or other points of interest. These annotations are often drawn from large datasets of location entities. The location entities are in turn often classified as "point of interest" (POI) entities or "yellow page" (YP) entities. POI entities are often created by users with mobile, GPS-enabled devices. Accordingly, the GPS coordinates for such entities tend to have a high degree of accuracy. Other fields of POI entities (e.g., name, address, etc.), however, tend to be less accurate as the entity-creating user may not enter those fields with a great degree of care. YP entities are often created by the businesses or locations that they identify, and may be captured for the dataset by, for example, crawling the Internet. Because YP entities are often created by businesses or locations having a strong desire to be found, name and address fields of the entities may be highly accurate. GPS coordinates for YP entities are then geo-coded based on the address field and vary in quality based on the accuracy of the address field.

[0003] These large datasets often include a number of entities with erroneous location information, resulting in location identifiers being placed on maps at the wrong locations. While location entities with erroneous location information may be manually located and deleted, doing so can be time and labor intensive.

SUMMARY

[0004] In various embodiments, a computing device is configured to arrange a plurality of location entities into a hierarchy of location descriptors. The computing device may further process determine whether one of the location entities is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity. Also, the other location entities and the one location entity may share a location descriptor.

[0005] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

DESCRIPTION OF DRAWINGS

[0006] Non-limiting and non-exhaustive examples are described with reference to the following figures:

[0007] FIG. **1** illustrates an overview of location dataset analysis, in accordance with various embodiments;

[0008] FIG. **2** an exemplary operating environment including a computing device programmed with location dataset analysis logic, in accordance with various embodiments;

[0009] FIGS. **3**A-**3**B are flowchart views of exemplary operations of a location dataset analysis, in accordance with various embodiments;

[0010] FIG. **4** illustrates a number of exemplary hierarchies of location descriptors, in accordance with various embodiments;

[0011] FIG. **5** illustrates the merging of location descriptors for location entities determined to refer to the same instance, in accordance with various embodiments;

[0012] FIG. **6** illustrates exemplary methods of determining spatial outliers, in accordance with various embodiments; and

[0013] FIG. **7** is a block diagram of an exemplary computing device.

DETAILED DESCRIPTION

Overview

[0014] FIG. **1** illustrates an overview of location dataset analysis, in accordance with various embodiments. As shown, a map or aerial photograph may include identifiers for a plurality of locations, such as identifiers **102***a*, **102***b*, and **104**. The identifiers may correspond to location entities retrieved or received from a location dataset. The location entities may include either or both of "point of interest" (POI) and "yellow page" (YP) entities.

[0015] Unless the context indicates otherwise, a POI entity, as used herein, refers to a location entity having a GPS coordinate field, the GPS coordinates having been captured by a GPS-enabled device. Also, unless the context indicates otherwise, a YP entity, as used herein, refers to a location entity for which at least an address field has been manually entered (or copied from a manually entered address) and GPS-coordinates have been geo-coded based on the address. A POI entity may also have a manually entered address, but a YP entity is often created by a business, thus providing an incentive for a more accurate recording of the address.

[0016] In FIG. **1**, both identifiers **102***a* and **102***b* refer to location entities named "Starbucks." Identifier **104** refers to a location entity named "Wal-mart." Visual inspection of the map seems to confirm the accuracy of identifiers **102***a* and **104**, and call into question the accuracy of identifier **102***b*, which appears to point to an empty field despite being associated with the name "Starbucks." Accordingly, Starbucks may be a "spatial outlier." A spatial outlier, as used herein, is a location entity having measurably erroneous location information.

[0017] To determine whether identifier **102***b* is a spatial outlier, a computing device programmed as shown in FIG. **2** and described below may perform a location dataset analysis on a plurality of location entities, including the location entity corresponding to identifier **102***b*. The computing device may arrange the location entities into a hierarchy of location descriptors and determine whether the location entity for identifier **102***b* is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the location entity for identifier **102***b*. The other location entities and the location entity for identifier **102***b* may share a location description. If the location entity for identifier **102***b* is determined to be a spatial outlier,

that location entity may be deleted and identifier **102***b* may not be rendered to a user as part of the map or aerial photograph.

Exemplary Operating Environment

[0018] FIG. **2** is a block diagram illustrating an exemplary operating environment, in accordance with various embodiments. More specifically, FIG. **2** shows a computing device **208** that is programmed to perform a location dataset analysis for location entities of a dataset **202**. In some embodiments, the computing device **208** may further retrieve or generate a framework **204** and/or one or more dictionaries **206**. The framework **204** may be used to segment addresses of location entities and to arrange the location entities into a hierarchy of location descriptors. The dictionaries **206** may also be used to segment the addresses of the location entities. To perform the location dataset analysis, computing device **208** may have location dataset analysis logic **210**. The location dataset analysis logic **210** may in turn include a plurality of modules, such as segmentation module **212**, hierarchy module **214**, and spatial outlier detection module **216**. As mentioned above, the location dataset analysis logic **210** may detect and delete spatial outlier location entities from the dataset **202**. A location search server **218** may then be receive the resulting dataset, the dataset to be served with maps to clients **220**.

[0019] In various embodiments, dataset **202** may be any sort of file storing a plurality of location entities. For example, dataset **202** may be a database file, a text file, or an XML file. In some embodiments, dataset **202** may be stored on a database server (not shown) that is separate and distinct from the computing device **202**, or on some other server or computing device. In other embodiments, dataset **202** may be stored on computing device **208**. Additionally, dataset **202** may comprise POI and YP location entities. An exemplary location entity may include fields for a location name, an address, a GPS position, a phone number, a category, and/or a type (e.g., POI or YP). A dataset **202** having two location entities is illustrated in Table 1:

TABLE 1

Name	Address	GPS Position	Phone Num.	Category	Type
Starbucks	7001 5th Ave Seattle, WA	116.325, 35.364	1-56987452	Café	YP
Silver Cloud Inn	3014 7th Ave Redmond, WA	116.451, 35.209	1-25698716	Restaurant	POI

[0020] In various embodiments, framework **202** may be a hierarchical tree structure of location descriptors. For example, framework **202** may have a location descriptor for a city as the root node, location descriptors for districts as the nodes for the next level, location descriptors for streets as the nodes for the third level, and location descriptors for buildings in the fourth level. In some embodiments, levels of the framework **202** may include multiple descriptor types (i.e., the second level may contain district nodes and a landmark node). FIGS. **4** and **5** illustrate exemplary frameworks **204** after the insertion of nodes for location entities. For example, picture A) in FIG. **4** illustrates a framework with a root node labeled "Beijing", a second level node labeled "Haidian District", and third level nodes labeled "Zhichun Road" and "Zhongguan chun".

[0021] In various embodiments, computing device **208** may create the framework **204** based on layout information for a city. For example, some cities may publish a file having layout information, and computing device **208** may build the hierarchical tree of framework **204** based on the layout information. In such embodiments, the framework **204** may be stored on computing device **208**. In other embodiments, framework **204** may be generated by another computing device and/or stored on another computing device, such as a storage server.

[0022] As further illustrated in FIG. **2**, computing device **208** may also utilize one or more dictionaries **206**. In some embodiments, dictionaries **206** may include a breaker words dictionary and a category words dictionary. Either dictionary **206** may be of any file format known in the art, such as a database file, a text file, or an XML file. The breaker words dictionary **206** may include words or phrases indicating a break between location descriptors comprising an address. For example, in the address "Chow restaurant, 200 meters from the Sigma building, Yi Ping Road, Beijing", "200 meters from" may be a breaker word, indicating a break between the location descriptors "Chow Restaurant" and "Sigma Building". Other exemplary breaker words may include "turn left", "on the north side of", etc. In some embodiments, one or more users of one or more computing devices may manually generate the breaker words dictionary **206**. The breaker words dictionary may be stored on computing device **208** or on some other computing device, such as a storage server.

[0023] In various embodiments, category words dictionary **206** may include words or phrases derived from name or category fields of location entities. For example, category words may include such words or phases as "restaurant", "company", or "shopping mall." In some embodiments, category words dictionary **206** may be manually generated, automatically generated based on names or categories of location entities, or both. If manually generated, one or more users of one or more computing devices may create or contribute to the document including the category words. If automatically generated, computing device **208** or some other computing device may process the location entities **208** to extract category words. In some embodiments, the computing device **208** or other device may extract a plurality of n-grams from the name and/or category fields of the location entities and evaluate those n-grams utilizing n-gram algorithms known in the art, selecting n-grams as category words that occur with a pre-defined frequency within dataset **202**.

[0024] As shown in FIG. **2**, computing device **208** may be any sort of computing device or devices known in the art, such as personal computers (PCs), laptops, servers, phones, personal digital assistants (PDAs), set-top boxes, and data centers. In some embodiments, the computing device **208** may be a particular machine configured to perform some or all of the location dataset analysis operations described above and below. As shown, computing device **208** may be programmed with location dataset analysis logic **210** and may thus be capable of detecting and deleting spatial outlier location entities from dataset **202** and providing the modified dataset **202** to a location search server **218**. Computing device **208** may further be configured to receive, retrieve, or generate any or all of dataset **202**, framework **204**, and/or dictionaries **206**, either as they are generated, at pre-determined times, or in response to a user command or request. FIG. **7** and its corresponding description below illustrate an exemplary computing device **208** in greater detail.

[0025] Also, in some embodiments, computing device **208**, location search server **218**, clients **220**, and/or device(s) storing any or all of dataset **202**, framework **204**, or dictionaries **206** may be connected by at least one networking fabric (not shown). For example, the device **208** and server **218** may be connected by a local access network (LAN), a public or private wide area network (WAN), and/or by the Internet. In some embodiments, the devices may implement between themselves a virtual private network (VPN) to secure the communications. Also, the devices may utilize any communications protocol known in the art, such as the Transmission Control Protocol/Internet Protocol (TCP/IP) set of protocols. In other embodiments, rather than being coupled by a networking fabric, the devices may be locally or physically coupled.

[0026] As is further illustrated in FIG. **2**, computing device **208** may include and be programmed with location dataset analysis logic **210** (hereinafter "logic **210**"). Logic **210** may be any set of executable instructions capable of performing the operations described below with regard to modules **212-216**. Logic **210** may reside completely on computing device **208**, or may reside at least in part on one or more other computing devices and may be delivered to computing device **208** via the above-described networking fabric. While logic **210** is shown as comprising concept segmentation module **212**, hierarchy module **214**, and spatial outlier determination module **216**, logic **210** may instead comprise more or fewer modules collectively capable of performing the operations described below with regard to modules **212-216**. Thus, modules **212-216** are shown and described simply for the sake of illustration, and all operations performed by any of the modules **212-216** are ultimately operations of logic **210** that may be performed by any sort of module of logic **210**.

[0027] In various embodiments, segmentation module **212** may segment an address field of a location entity into a plurality of location descriptors. For example, if the address of a location entity is "4F Sigma Building, No. 49 Zhichun Road, Haidian District, Beijing", then segmentation module **212** may segment the address into four phrases/descriptors: Beijing, Haidian District, Zhichun Road, and Sigma Building. In some embodiments, segmentation module **212** may operate to determine the segmentation by finding word delimiters, such as commas, or other grammatical symbols. Each delimiter or symbol may be regarded as separating two location descriptors. For an address with N delimiters or symbols, there may be N+1 location descriptors/phrases. In some embodiments, in addition to extracting phrases/descriptors, segmentation module **212** may filter out information such as street or suite numbers. Thus, for example, segmentation module **212** may filter "4F" from "4F Sigma Building", leaving "Sigma Building" as the location descriptor. Also, in some embodiments, segmentation module **212** may filter out words that match entries in the breaker words or category words dictionaries **206** from the location descriptors.

[0028] In other embodiments, segmentation module **212** may instead segment addresses of location entities based at least in part on framework **204** and dictionaries **206**. For example, in other languages, such as Chinese, delimiters or symbols do not separate the terms of an address. Thus, some other mechanism of separating the terms into location descriptors is required. In various embodiments, segmentation module **212** may separate an address into words/descriptors by comparing the address to the contents of the framework **204** and dictionaries **206**. For example, framework **204** may contain a root node associated with the location descriptor "Beijing." The address of the location entity under evaluation by segmentation module **212** may also include the word Beijing. Upon finding a match, the segmentation module **212** may consider the word "Beijing" a location descriptor for the location entity. Also, if segmentation module **212** finds a match between a portion of the address and the breaker words dictionary (e.g., turn left), it may consider the words on either side of the breaker word to be candidate words/descriptors. Further, the segmentation module **212** may compare the address of a location entity to words contained in the category words dictionary **206**. If a match is found, such as "building", the segmentation module **212** may consider the matching word to be a delimiter, as category words are often the last word or words in any portion of an address.

[0029] In some embodiments, the segmentation module **212** may then filter out words that match entries in the breaker words or category words dictionaries **206** from the determined location descriptors, as well as street numbers, etc. After filtering, for any portion of the address that has not yet been matched and is over a certain threshold length (e.g., 5 Chinese characters), the segmentation module **212** may attempt to split that portion. Other unmatched portions that are smaller than the threshold length may be considered location descriptors. To split an address portion with a length exceeding the threshold length, the segmentation module **212** may again compare the portion to the framework **204** to determine of any sub-portion matches the framework **204**. If a match is found, the segmentation module **212** may consider the sub-portion a location descriptor and may again filter and split the remaining portion, if necessary. If a match is not found, then the segmentation module **212** may consider the portion of the address a location descriptor.

[0030] In various embodiments, hierarchy module **214** may arrange the location entities into a hierarchy of location descriptors, such as framework **204**, as mentioned above. To arrange the descriptors derived from the address by the segmentation module **212** into the framework **204**, hierarchy module **214** may start with the broadest descriptor, such as a city name, and determine if it is present in the framework. The broadest descriptor may be the last descriptor that appears in an address, if the address is in English, or the first descriptor in an address, if the address is in Chinese. If not descriptors are found in the framework **204** (i.e., the framework **204** is empty), then the broadest descriptor may be added as a root node. The hierarchy module **214** may then repeat the determining of whether each descriptor is present in the framework **204** until the narrowest descriptor is reached. For each descriptor not found in the framework **204**, hierarchy module **214** may add it as a child node of a parent node that corresponds to the next broadest descriptor in the same address. For example, if an address include the fragment "Zhichun Road, Haidian District", and "Haidian District" is present in the framework and "Zhichun Road" is not, then hierarchy module **214** may add "Zhichun Road" as a child node of "Haidian District." If the narrowest descriptor is already present in the framework **204**, then the hierarchy module **214** may associate the node for the narrowest descriptor with the location entity containing that descriptor, in some embodiments by a pointer or index to the location entity. In various embodiments, the narrowest descriptor may often correspond to a leaf node of framework **204**.

[0031] In various embodiments, FIG. **4** illustrates several exemplary insertions of location descriptors into frameworks

204. In picture A), a location entity having the descriptors "Sigma Building, Zhichun Road, Haidian District, Beijing" is inserted into the framework **204**. Hierarchy module **214** may compare the descriptors of the address of the location entity to the framework **204** and determine that only the narrowest descriptor, "Sigma Building", is not present in the framework **204**. Hierarchy module **214** may then add a Sigma Building node **404** as a child node of the node **402** for the next broadest descriptor, Zhichun Road, and associate the location entity being inserted with the Sigma Building node **404**.

[0032] Further illustrated in FIG. **4** in picture B), sometimes a location entity may contain two descriptors at the same level in the framework **204**. For example, "Zhichun Road" and "Zhongguan chun" appear as nodes **402** at the same level of the framework **204**. They are peers rather than parent and child. An address of a location entity, however, because of an error or other reason, may include them in sequence, suggesting a parent-child relationship. When hierarchy module **214** encounters this situation, it may add an edge between the peer nodes **402**, the edge pointing from the node **402** in the "broader" position in the address to the node **402** in the "narrower" position.

[0033] Again referring to FIG. **4**, in Picture C), sometimes the same narrowest descriptor for a plurality of different location entities may be inserted at a plurality of different locations in the framework **204**. For example, three different location entities sharing the same narrowest location descriptor, "Tsinghua University", may each have an address field comprised of different sets of descriptors. A first may have "Tsinghua University, Beijing." A second may have "Tsinghua University, Chengfu Road, Haidian District, Beijing." A third may have "Tsinghua University, Zhongguan chun, Haidian District, Beijing." In evaluating these sets of descriptors, hierarchy module **214** may insert a node **404** for the narrowest descriptor multiple times in multiple places because of the different parent-child relationships implied by the descriptors.

[0034] In various embodiments, after inserting descriptors for the location entities, the hierarchy module **214** may merge nodes/descriptors which refer to the same instance. For example, referring to FIG. **4** picture C), the three Tsinghua University nodes may actually all refer to the same, physical Tsinghua University. Before merging descriptors, however, hierarchy module **214** must determine whether the descriptors refer to the same instance. If there are ten descriptors for "Starbucks", they may well refer to several different physical Starbucks locations. To determine whether multiple location descriptors refer to the same instance, hierarchy module **214** may determine the frequency with which the location descriptor occurs as a child node of a common parent node. If the frequency exceeds a pre-defined threshold, the hierarchy module **214** may determine that the descriptors refer to the same instance. For example, in FIG. **5**, the descriptor "Tsinghua University" is shown as occurring as a child node **504** of the "Beijing" node **502** twenty-five times. If the threshold is, for example, twenty occurrences, then hierarchy module **214** will consider all descendent nodes of Beijing referring to Tsinghua University to refer to the same instance.

[0035] In some embodiments, hierarchy module **214** may then determine the number of location entities associated with each node having the descriptor to be merged. For example, eight location entities may be associated with the node "Tsinghua University" that is a child of "Chengfu Road". In other words, eight location entities may share this same set of location descriptors. Hierarchy module **214** may then select the node with the lowest level in framework **204** whose number of associated location entities exceeds a predetermined threshold. Continuing with the example above, there may be a lower level node for Tsinghua University (e.g., "Tsinghua University, Fudan Campus, Chengfu Road"), but that lower level node may only be associated with, for example, three location entities. If the threshold is seven, then hierarchy module **214** may select the node associated with the eight location entities. Hierarchy module **214** may then retain the selected node and delete the other nodes sharing the location descriptor, effectively combining the nodes. In combining the nodes, hierarchy module **214** may add edges from the parent nodes of the nodes being deleted to the node being retained, and may associate the location entities of the nodes being deleted to the node being retained. For example, FIG. **5** illustrates the lowest level node **504** being retained, and other nodes **504** being deleted. Also, edges are shown being added from each of the parent nodes **502** of deleted nodes **504** to the retained node **504**.

[0036] As is further illustrated in FIG. **2**, the spatial outlier determination module **216** (hereinafter "outlier module **216**") may determine whether a given entity is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity, the other location entities and the one location entity sharing a location descriptor. To determine whether a location entity is a spatial outlier, outlier module **216** may apply at least one of a D-P algorithm or a POI-based algorithm.

[0037] In various embodiments, if performing the D-P algorithm, the outlier module **216** may first determine a reference set for a location entity. The reference set may include other location entities associated with the same node of framework **204**, the same parent node, and/or a same ancestor node. Based on the set selected, outlier module **216** may further determine or select a distance d and a number p of other location entities expected to be within that distance. In one embodiment, the outlier module **216** may calculate the distance d by determining a box which includes the geographic area of the common node for the set. If the common node is the same node, the box may be small, and if the common node is an ancestor node, the box may be large. The outlier module **216** may then multiply a diagonal of the box by a predetermined percentage (e.g., 10%), and may assign the resulting value to the distance d. In some embodiments, outlier module **216** may calculate p by determining the total number of location entities in the reference set and multiplying that total number by a predetermined fraction (e.g., ¼). Once d and p have been selected or calculated, the outlier module **216** may determine the number of location entities from the reference set that are within the distance d of the location entity being evaluated. In performing this determining, the outlier module **216** may utilize the GPS coordinates of each location entity. If that number does not meet or exceeds p, the outlier module **216** may deem the location entity a spatial outlier.

[0038] FIG. **6** illustrates the D-P algorithm in further detail. As shown, the outlier module **216** may compare a location entity being checked **606** (which may be either a POI or YP entity) to other location entities **602** comprising a reference set. If the number of location entities **602** within the distance d of entity **606** does not meet or exceed p, then outlier module **216** may deem entity **606** a spatial outlier **604**.

[0039] In various embodiments, the outlier module **216** may only perform the POI-based method if the entity being checked is a YP entity. If performing the POI-based algorithm, the outlier module **216** may first determine a reference set for the YP entity, the reference set including only POI entities. The reference set may include POI location entities associated with the same node of framework **204**, the same parent node, and/or a same ancestor node as the YP entity being checked. Based on the set selected, outlier module may further determine or select a distance d. In one embodiment, outlier module **216** may calculate the distance d by determining a box which includes the geographic area of the common node for the set. If the common node is the same node, the box may be small, and if the common node is an ancestor node, the box may be large. The outlier module **216** may then multiply a diagonal of the box by a predetermined percentage (e.g., 10%), and may assign the resulting value to the distance d. Once d is selected or calculated, the outlier module **216** may determine whether any POI entities of the reference set are within the distance d of the YP entity. In performing this determining, the outlier module **216** may utilize the GPS coordinates of each location entity. If no POI entities are within d of the YP entity, then outlier module **216** may deem the YP entity to be a spatial outlier.

[0040] FIG. **6** illustrates the D-P algorithm in further detail. As shown, the outlier module **216** may compare a YP entity being checked **614** to POI entities **610** comprising a reference set. If no POI entities **610** are within distance d of the YP entity **614**, then the outlier module **216** may deem the YP entity **614** a spatial outlier **612**. Also, the presence or absence of other YP entities **608** within the distance d of YP entity **614** may make no difference in the outcome of the POI-based algorithm.

[0041] In various embodiments, after determining that a location entity is a spatial outlier, the outlier module **216** may delete the location entity from the dataset **202**, or create a new modified dataset **202** which does not include the spatial outlier. The outlier module **216** may then repeat the determination of whether an entity is a spatial outlier for some or all of the other entities of the dataset **202**. In some embodiments, outlier module **216** may perform both the D-P algorithm and the POI-based algorithm for a location entity. In one embodiment, the outlier module **216** may then only delete the location entity if both algorithms deem it a spatial outlier. In another embodiment, the outlier module **216** may delete the location entity so long as it is deemed a spatial outlier by one of the algorithms.

[0042] As is further illustrated by FIG. **1**, a location search server **218** may receive the modified dataset **202**, with the spatial outliers deleted, from the computing device **208**. Location search server **218** may be any sort of computing device or devices known in the art, such as personal computers (PCs), laptops, servers, phones, personal digital assistants (PDAs), set-top boxes, and data centers. In one embodiment, location search server **218** and computing device **208** may be the same physical computing device. Location search server **218** may be configured to provide location search services, such as Windows Local Live Search™, to a plurality of client **220** over a networking fabric, such as the networking fabric described above. The location search services may include providing the clients **220** with maps or photographs annotated with identifiers corresponding to location entities of the modified dataset **202**, in some embodiments.

Exemplary Operations

[0043] FIGS. **3**A-**3**B are flowchart views of exemplary operations of a location dataset analysis, in accordance with various embodiments. As illustrated in FIG. **3**A, a computing device may first segment address fields of a plurality of location entities into location descriptors, block **302**. In some embodiments, the location entities may comprise yellow page entities and point of interest entities. Also, at least one of the location entities may comprise a location name, a location address, and a global positioning system (GPS) position. In various embodiments, the segmenting, block **302**, may comprise segmenting based on commas and/or other characters indicating a separation between two or more terms. In other embodiments, the segmenting, block **302**, may comprise segmenting based at least in part on one or more frameworks and/or dictionaries. In some embodiments, the framework may be a tree structure of location descriptors generated from a published description of a geographic area. Also, the dictionaries may include either a collection of breaker words used to separate location descriptors or a collection of categories derived from name fields of the location entities.

[0044] In various embodiments, the computing device may then arrange the plurality of location entities into a hierarchy of location descriptors, block **304**. The arranging shown in block **304** is illustrated in greater detail in FIG. **3**B and is described further herein.

[0045] As further illustrated in FIG. **3**A, the computing device may then determine whether one of the location entities is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity, block **306**. In some embodiments, the other location entities and the one location entity may share a location descriptor. In various embodiments, the determining, block **306**, may also comprise determining whether the number of other location entities within the predetermined distance of the one location entity exceeds a threshold and, in response, determining that the one location entity is a spatial outlier. Further, in some embodiments, when the one location entity is a yellow page entity and the other location entities are point of interest entities, the determining, block **306**, may comprise determining whether at least one of the point of interest entities is present within a predetermined distance of the yellow page entity and, in response, determining that the yellow page entity is a spatial outlier.

[0046] In various embodiments, in response to determining that the one location entity is a spatial outlier, the computing device may delete the one location entity, block **308**.

[0047] FIG. **3**B illustrates the arranging of block **304** in further detail. As illustrated, the arranging may include inserting, by the computing device, a descriptor of each location entity derived from an address field of each location entity as a leaf node in a tree of location descriptors, block **304***a*. Next, the computing device may determine that at least two leaf nodes refer to a same instance if the nodes share the same descriptor and if the same descriptor is shared by a number of descendant nodes of a same parent, block **304***b*, the number exceeding a first threshold. Then, the computing device may combine the at least two leaf nodes, block **304***c*, the combining including retaining one of the leaf node at a lowest level in

the hierarchy in which a number of occurrences of the at least two leaf nodes exceeds a second threshold.

Exemplary Computing Device

[0048] FIG. **7** illustrates an exemplary computing device **700** that may be configured to determine whether a location entity is a spatial outlier.

[0049] In a very basic configuration, computing device **700** may include at least one processing unit **702** and system memory **704**. Depending on the exact configuration and type of computing device, system memory **704** may be volatile (such as RAM), non-volatile (such as ROM, flash memory, etc.) or some combination of the two. System memory **704** may include an operating system **705**, one or more program modules **706**, and may include program data **707**. The operating system **705** may include a component-based framework **720** that supports components (including properties and events), objects, inheritance, polymorphism, reflection, and provides an object-oriented component-based application programming interface (API), such as that of the .NET™ Framework manufactured by Microsoft Corporation, Redmond, Wash. The device **700** may be of a configuration demarcated by a dashed line **708**.

[0050] Computing device **700** may also have additional features or functionality. For example, computing device **700** may also include additional data storage devices (removable and/or non-removable) such as, for example, magnetic disks, optical disks, or tape. Such additional storage is illustrated in FIG. **7** by removable storage **709** and non-removable storage **710**. Computer storage media may include volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data. System memory **704**, removable storage **709** and non-removable storage **710** are all examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computing device **700**. Any such computer storage media may be part of device **700**. Computing device **700** may also have input device(s) **712** such as keyboard, mouse, pen, voice input device, touch input device, etc. Output device(s) **714** such as a display, speakers, printer, etc. may also be included. These devices are well know in the art and need not be discussed at length here.

[0051] Computing device **700** may also contain communication connections **716** that allow the device to communicate with other computing devices **718**, such as over a network. Communication connections **716** are one example of communication media. Communication media may typically be embodied by computer readable instructions, data structures, program modules, etc.

Closing Notes

[0052] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms of implementing the claims.

[0053] References are made in the detailed description to the accompanying drawings that are part of the disclosure and which illustrate embodiments. Other embodiments may be utilized and structural or logical changes may be made without departing from the scope of the disclosure. Therefore, the detailed description and accompanying drawings are not to be taken in a limiting sense, and the scope of embodiments is defined by the appended claims and equivalents.

[0054] Various operations may be described, herein, as multiple discrete operations in turn, in a manner that may be helpful in understanding embodiments; however, the order of description should not be construed to imply that these operations are order-dependent. Also, embodiments may have fewer operations than described. A description of multiple discrete operations should not be construed to imply that all operations are necessary.

[0055] The description may use perspective-based descriptions such as up/down, back/front, and top/bottom. Such descriptions are merely used to facilitate the discussion and are not intended to restrict the scope of embodiments.

[0056] The terms "coupled" and "connected," along with their derivatives, may be used herein. These terms are not intended as synonyms for each other. Rather, in particular embodiments, "connected" may be used to indicate that two or more elements are in direct physical or electrical contact with each other. "Coupled" may mean that two or more elements are in direct physical or electrical contact. However, "coupled" may also mean that two or more elements are not in direct contact with each other, but yet still cooperate or interact with each other.

[0057] The description may use the phrases "in an embodiment," or "in embodiments," which may each refer to one or more of the same or different embodiments. Furthermore, the terms "comprising," "including," "having," and the like, as used with respect to embodiments, are synonymous.

[0058] For the purposes of the description, a phrase in the form "A/B" means A or B. For the purposes of the description, a phrase in the form "A and/or B" means "(A), (B), or (A and B)". For the purposes of the description, a phrase in the form "at least one of A, B, and C" means "(A), (B), (C), (A and B), (A and C), (B and C), or (A, B and C)". For the purposes of the description, a phrase in the form "(A)B" means "(B) or (AB)" that is, A is an optional element.

1. A method comprising:

arranging, by a computing device, a plurality of location entities into a hierarchy of location descriptors; and

determining, by the computing device, whether one of the location entities is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity, the other location entities and the one location entity sharing a location descriptor.

2. The method of claim **1**, wherein the location entities comprise yellow page entities and point of interest entities.

3. The method of claim **1**, wherein at least one of the location entities comprises a location name, a location address, and a global positioning system (GPS) position.

4. The method of claim **1** further comprising segmenting address fields of the location entities into location descriptors.

5. The method of claim **4**, wherein the segmenting comprises segmenting based on commas and/or other characters indicating a separation between two or more terms.

6. The method of claim 4, wherein the segmenting comprises segmenting based at least in part on one or more frameworks and/or dictionaries.

7. The method of claim 6, wherein the framework is a tree structure of location descriptors generated from a published description of a geographic area.

8. The method of claim 6, wherein the dictionaries include either a collection of breaker words used to separate location descriptors or a collection of categories derived from name fields of the location entities.

9. The method of claim 1, wherein the arranging further comprises inserting a descriptor of each location entity derived from an address field of each location entity as a leaf node in a tree of location descriptors.

10. The method of claim 9, wherein the arranging further comprises determining that at least two leaf nodes refer to a same instance if the nodes share the same descriptor and if the same descriptor is shared by a number of descendant nodes of a same parent, the number exceeding a first threshold.

11. The method of claim 10, wherein the arranging further comprises combining the at least two leaf nodes, the combining including retaining one of the leaf node at a lowest level in the hierarchy in which a number of occurrences of the at least two leaf nodes exceeds a second threshold.

12. The method of claim 1, wherein the determining further comprises:

determining whether the number of other location entities within the predetermined distance of the one location entity exceeds a threshold; and

in response to determining that the number does not exceed the threshold, determining that the one location entity is a spatial outlier.

13. The method of claim 1, wherein the one location entity is a yellow page entity and the other location entities are point of interest entities, and the determining further comprises:

determining whether at least one of the point of interest entities is present within a predetermined distance of the yellow page entity; and

in response to determining that no point of interest entity is present within the predetermined distance, determining that the yellow page entity is a spatial outlier.

14. The method of claim 1 further comprising, in response to determining that the one location entity is a spatial outlier, deleting the one location entity.

15. An article of manufacture comprising:

a storage medium; and

a plurality of executable instructions stored on the storage medium which, when executed by a computing device, perform operations including:

arranging a plurality of location entities into a hierarchy of location descriptors, the location entities including yellow page entities and point of interest entities; and

determining whether one of the yellow page entities is a spatial outlier based at least in part on presence of at least one of the point of interest entities within a predetermined distance of the one yellow page entity, the one point of interest entity and the one yellow page entity sharing a location descriptor.

16. The article of claim 15, wherein the arranging further comprises inserting a descriptor of each location entity derived from an address field of each location entity as a leaf node in a tree of location descriptors.

17. The article of claim 15, wherein the determining further comprises, in response to determining that no point of interest entity is present within the predetermined distance, determining that the yellow page entity is a spatial outlier.

18. The article of claim 17, wherein the executable instructions, when executed by the computing device, further perform operations including, in response to determining that the yellow page entity is a spatial outlier, deleting the yellow page entity.

19. A system comprising:

a processor; and

logic configured to be executed by the processor to perform operations including:

segmenting address fields of a plurality of location entities into location descriptors, the segmenting including either or both of:

segmenting based on commas and/or other characters indicating a separation between two or more terms; and

segmenting based at least in part on one or more frameworks and/or dictionaries;

arranging the location entities into a hierarchy of location descriptors, the arranging including:

inserting a descriptor of each location entity derived from an address field of each location entity as a leaf node in a tree of location descriptors;

determining that at least two leaf nodes refer to a same instance if the nodes share the same descriptor and if the same descriptor is shared by a number of descendant nodes of a same parent, the number exceeding a first threshold; and

combining the at least two leaf nodes, the combining including retaining one of the leaf node at a lowest level in the hierarchy in which a number of occurrences of the at least two leaf nodes exceeds a second threshold;

determining whether one of the location entities is a spatial outlier based at least in part on presence of one or more other location entities within a predetermined distance of the one location entity, the other location entities and the one location entity sharing a location descriptor; and

in response to determining that the one location entity is a spatial outlier, deleting the one location entity.

20. The system of claim 19, wherein determining whether the one location entity is a spatial outlier further comprises:

determining whether the number of other location entities within the predetermined distance of the one location entity exceeds a threshold; and

in response to determining that the number does not exceed the threshold, determining that the one location entity is a spatial outlier.

* * * * *