June 10, 2016

SELECTIVELY DELAYING LOCATION-BASED QUERIES

Fredrik Bergenlid

Nils Grimsmo

SELECTIVELY DELAYING LOCATION-BASED QUERIES

Introduction

Many Internet-capable mobile devices (smart phones, tablets, laptops, etc.) implement geo-location techniques using Global Positioning Systems (GPS), for example. Software applications executing on such devices allow users to submit queries related to the current geographic locations of the devices ("location-based queries") by typing or speaking. Higher precision in determining the location generally results in more pertinent results in response to a location-based query. However, regularly and frequently updating the location of the mobile device using GPS, or another sensor-based positioning technique, may not be practical because of the limited battery life and the bandwidth.

As one alternative to regular and frequent updates, a software application can determine the current location at the time of query. The location update request, reporting the new location of the device, may be streamed with the query up to a server. However, the time delay before the GPS can provide a good and accurate location fix may be longer than the time required to complete the query. Even if the mobile device uses a technique that is not GPS-based, in general there are many different components involved in determining an accurate user location, and some external signals (e.g., WLAN signals), may take many seconds to stabilize. Further, always trying to acquire the current user location at the time of a query is inefficient because many queries are not location-based.

More generally, constraints such as device hardware, bandwidth, signal strength, battery power, etc., limit the ability software applications to quickly and efficiently execute location-based queries.

Delayed location-based queries

A system of this disclosure determines, based on characteristics of a query submitted via a mobile device, whether the quality of the search results is likely to benefit from a more precise location of the mobile device. The query can be spoken or typed, for example. If the system determines that the results are likely to improve with a better location fix, the system delays the query until a more accurate location is determined, or until a certain deadline expires. The deadline can be determined based on the level of precision required by the query, and/or how

poor the quality of the search results for the query are expected to be without a more exact location. Thus, the mobile device in some cases is allotted some time to acquire a better location for inclusion in the query. As a result, the search results in these cases are likely to be more pertinent.

On the other hand, this technique does not necessitate an increase latency of non-location-based queries, nor does it require a new location update when the location is already current and accurate. Another benefit of determining whether a query is location-based is that a server can be implemented to leverage search technology and delay certain queries more than others based on the determined context of the query and other signals from normal search. A longer delay typically will result in a more accurate user location.

To determine whether a certain query can benefit from a better location fix, the system can look at the context of the query itself (e.g. keywords) and/or the search results of the given query. When looking at the query itself, the system can quickly identify terms or phrases that indicate that the query is location-based. For example, words such as “here” or “near me” have little meaning without a position fix, and responses to queries that include such words are likely to improve significantly with a more precise position fix. However, the query alone may not provide enough contextual information to precisely determine the required accuracy of the position fix.

On the other hand, when analyzing search results of the query, the system can more precisely determine the accuracy needed for the location fix. For example, search results relating to a store within a few meters of user may benefit more from an accurate position fix than search results relating to a store across town. Compared to analyzing the context of the query, analyzing the search results generally provides greater information about the level of accuracy needed for the position of the user, but generally takes longer to complete.

To maximize both the precision and speed of determining the required accuracy of the location fix, the system can use a combination of analyzing the query itself and the search results for the given query. By combining both methods, the system can analyze keywords of a query as the query as received and then complete the determination once search results are received. Queries that potentially can be improved by determining the location of the querying device with a certain degree of precision can be referred to a “location-seeking” queries.

In some embodiments, the determining whether a query is 'location-seeking" may be based on a combination of categories, the categories including one or more of result types, distances to the user and/or user intent. A result type refers to the classification of the query. For example, if the query is related to a location of a business, then the query belongs to a result type that requires an accurate location fix. On the other hand, if the query is related to a celebrity, then the query belongs to a result type that is not location-based.

The distance to the user also helps indicate the level of accuracy required for the location fix to return appropriate search results. For example, if a user searches for the business hours of a store, the distance of the user to the store can impact the required accuracy of the location fix. Generally, the closer a user is to the location, the greater accuracy required for the location fix. Similarly, if a user is only near one store, the location accuracy might not require the same level of precision as a user that is near two or more stores.

User intent may also influence the need for an accurate location fix. For example, if a user searches for the phone number of a local business, an updated location fix might not be necessary. On the other hand, if the user searches for an address and/or directions to a local business, an accurate location fix may be necessary to return appropriate search results.

Using a combination of analyzing both the query itself and the search results for the given query may help maximize the usefulness of the categories above. Elements of each category can be detected in each phase of the analysis. In turn, the system may run more effectively than a system implementing only one phase of analysis. The result is quicker and more precise determination of the need for an updated location fix.

To analyze the queries, any combination of n-grams (sequences of words that are commonly grouped together) and/or Knowledge Graphs can be implemented. For example, if a user has entered the words "butcher shops near me" into the query, the system may identify the n-gram "near me" to determine that query is location-seeking. Alternatively, the system may implement a Knowledge Graph to analyze, for example, the query "when does Bob's open on Monday," to determine that "Bob's" is a restaurant in the area and that the query is location seeking. In another example, the system may combine n-grams and Knowledge Graphs to determine that a query containing a restaurant name and the word "reviews" is not location-seeking.

Depending on the implementation, the system can include a component executing on a mobile device that overwrites a portion of a user-submitted query with location data. For example, a software application can receive the query "what are the opening hours here" and replace "here" with a set of geographic coordinates, a street address, the name of a neighborhood, the name of a town, etc. Alternatively, the software application can leave the user-submitted query intact and submit location data along with the query. The software application can submit an indication of how precise the location is.

In some cases, the component executing on the mobile device interprets voice input, constructs a query, and forwards the query to a network server. The network server determines that a more accurate location may improve the results of the query (e.g., based on the criteria discussed above), and instruct the mobile device to attempt a location update. The mobile device completes a location update to obtain better location data and re-submits the query with the more precise location.

Example computing system

Fig. 1 illustrates examples of a computing environment in which a software application running on a mobile device determines whether a query is location-based and, in some cases, delays the query to obtain a better position fix. As illustrated in Fig. 1, the computing environment 100 includes a query database 103 connected to, or disposed within, a server 105, which, in turn, is connected to one or more client mobile devices 115 through a network 125. The network 125 can include any suitable combination of a LAN, a MAN, a WAN, a mobile, a wired or wireless network, a private network, or a virtual private network. While only one client mobile device 115 is illustrated in Fig. 1 to simplify and clarify the description, it will be understood that any number of client devices are supported and be in communication with the server 105.

The server 105 and the client 115 are computing devices that may include a CPU 130 (only shown in the client), one or more computer readable memory modules 132, one or more user interfaces 134 (keyboard, touch screen, etc.), a network interface 136, one or more peripheral interfaces, and other components. Further, other types of computers can be used that have different architectures. The client device 115 represents any suitable handheld and/or mobile device, such as a mobile phone, personal data assistant, laptop computer, tablet personal

computer, car navigation system, hand-held GPS unit, or "smart" device. More generally, the client device 115 represents any personal computing device or any other processing device having a user interface and CPU and capable of performing search queries. In at least some of the embodiments, the client device 115 is equipped with a positioning module such as a GPS receiver, a Wireless Local Area Network (WLAN) receiver configured to perform triangulation, etc. Furthermore, while in some examples, the network 125 is described as a wireless network, the network 125 may be any wired or wireless network, where the client 115 is a device on the network.

The query database 103, which may be stored in or may be separate from the server 105, contains data that can be used to determine whether a given input is location-based. Names of physical roads, businesses, parks, landmarks, and other geographic elements are example inputs that may be represented in the query data. The query database may also contain indications of precision necessary for a user's location to return the best search results. For example, some search queries will only require an approximate user location (within a few hundred feet) while other queries will require precise coordinates (within a few feet).

The data stored in the query database 103 can be obtained from several different sources. Preferentially the data stored in the query database 103 will constantly be evolving as information is gathered and certain queries can be better defined as location-based, not location-based and/or the level of precision required to accurately respond to a query. Further, the database 103 does not need to be physically located within server 105. For example, the database 103 can be stored in external storage attached to the server 105, or can be stored in a network attached storage. Additionally, there may be multiple servers 105 that connect to a single database 103. Likewise, the query database 103 may be stored in multiple different or separate physical data storage devices.

A search engine 138 may be implemented on the client 115 or on the query data server 110. A context module 140 may interpret and use query data from the database 103 to determine if the input of a query is location-based or not. Additionally or alternatively, the context module 140 can cause the client mobile device 115 to receive an updated location if an input is determined to be location-based. The context module 140 can interpret audio or text input as part of a query. In this figure, the system is implemented as a context module 140 operating in

the search engine 138. More generally, however, the system can be implemented in a user device, a network server, or both.

Example method for delaying location-based queries

Fig. 2 illustrates a flow chart of an example method for updating a user's location when a location-based query is detected. The method of the current disclosure can be implemented in the client device 115 and executed by the context module 140 and/or in the server 105 (or another suitable network device). For convenience, the method is discussed below with reference to the context module 140.

The method begins at block 201, where the context module 140 receives a query through search engine 138. The query can be received through an input of the mobile device and can one or more of a text input, voice input, a received image, or any valid entry that can be interpreted by a search engine. At block 202, the context module 140 may send the received query to the query data server 105.

At block 203, either the server or the mobile device may implement the context module 140 to determine if the query is a location-based query. This determination can be made either by analyzing the context of the query itself (e.g. keywords) and/or by analyzing preliminary search results of the given query with the current position fix. If the query is location-based, the module 140 may delay the search to update the user location. Further, if the query has been determined to be location based, the context module 140 will determine the precision required of the user location to return the best possible search results. As discussed above, some location-based searches require a precise user location to return the best possible search results while other location-based searches can return satisfactory results with a more approximate user location. For example, a query of "museums" may only return a few search results within a city while a query of "restaurants" can include thousands of results. Thus, the query for "museum" would require a more approximate user location than the query for "restaurants" to return more pertinent search results. Retrieving a precise location may require a longer delay before completing the search, so it is beneficial to use proximity when possible. However, if the query is not location-based the module 140 will not delay the search and the server 105 will return search results as usual.

At block 204, the context module 140 may determine if a new user location is needed. The module 140 can check when the user location was last updated and the precision of the coordinates of the last update. Depending on the precision required by the location-based query, the module 140 can determine if the current user location information satisfies the requirements to return accurate search results. If the current user location is not satisfactory, the context module 140 will request an updated user location (block 205). However, if the current user location is sufficient, the context module will send the current user location with the query to determine search results.

At block 205, the context module 140 will request an updated user location based on the precision requirements determined above. At block 206, the context module will send the query and updated user location to the query server 105. The query data server 105 will complete the query and return the results, block 207.

ABSTRACT

To prevent superfluous updating of the location of a mobile device, a search engine can implement a tool to determine if a query is location-based before requesting an updated user location. The tool can also determine a precision of the updated location required to return accurate search results. The tool helps retrieve accurate and quick results to location-based queries while preventing unnecessary updates of a user location for non-location-based queries, in turn saving battery life of the mobile device and bandwidth on the network.

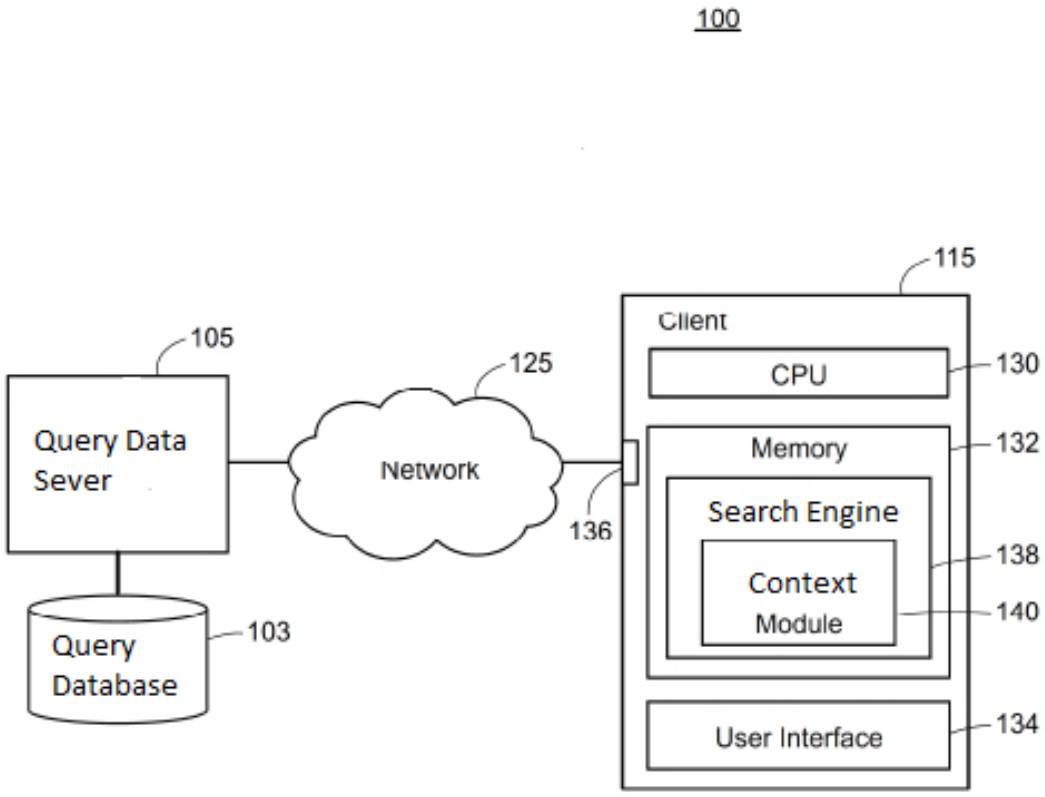
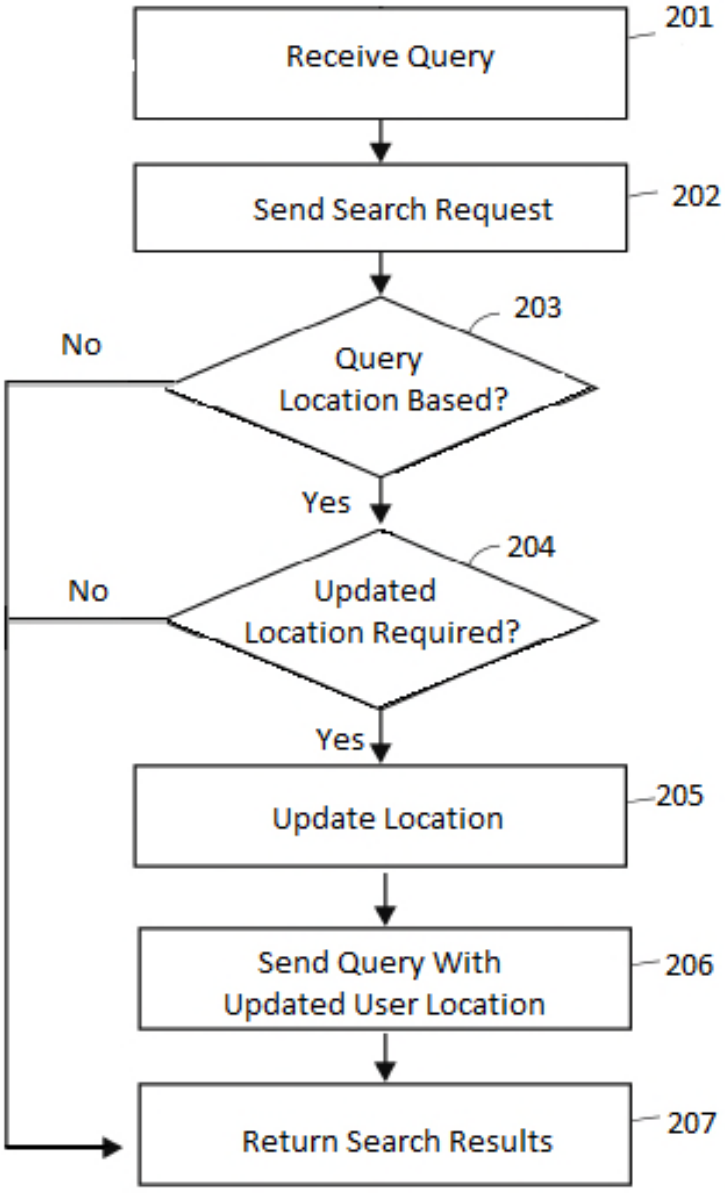

100
115
Client
105
Query Data Sever
125
Network
CPU
130
Memory
132
Search Engine
Context Module
138
140
136
Query Database
103
User Interface
134

FIG. 1

FIG. 2


Receive Query
201
Send Search Request
202
Query Location Based?
203
No
Yes
Updated Location Required?
204
No
Yes
Update Location
205
Send Query With Updated User Location
206
Return Search Results
207