TsingNUS: A Location-Based Service System Towards Live City

Guoliang Li†, Nan Zhang†, Ruicheng Zhong†, Sitong Liu†, Weihuang Huang†
Ju Fan‡, Kian-Lee Tan‡, Lizhu Zhou†, Jianhua Feng†
†Department of Computer Science, Tsinghua University, Beijing, China
‡Department of Computer Science, National University of Singapore, Singapore
{liguoliang,fengjh,dcszlz}@tsinghua.edu.cn; n-zhang10@mails.tsinghua.edu.cn
{zrc1101001,stoneliu2010,nellybamoo,fanju84}@gmail.com; tankl@comp.nus.edu.sg

ABSTRACT

We present our system towards live city, called TsingNUS, aiming to provide users with more user-friendly location-aware search experiences. TsingNUS crawls location-based user-generated content from the Web (e.g., Foursquare and Twitter), cleans and integrates them to provide users with rich well-structured data. TsingNUS provides three user-friendly search paradigms: location-aware instant search, location-aware similarity search and direction-aware search. Instant search returns relevant answers instantly as users type in queries letter by letter, which can help users to save typing efforts significantly. Location-aware similarity search enables fuzzy matching between queries and the underlying data, which can tolerate typing errors. The two features boost the search performance and improve the experiences for mobile users who often misspell the keywords due to the limitation of the mobile phone's keyboard. In addition, users have direction-aware search requirements in many applications. For example, a driver on the highway wants to find the nearest gas station or restaurant. She has a search requirement that the answers should be in front of her driving direction. TsingNUS enables direction-aware search to address this problem and allows users to search in specific directions. Moreover, TsingNUS incorporates continuous search to efficiently support continuously moving queries in a client-server system which can reduce the number of queries submitted to the server and communication cost between the client and server. We have implemented and deployed a system which has been commonly used and widely accepted.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases

Keywords

Instant Search, Location aware, Direction aware

1. INTRODUCTION

Location-based services (LBS) have been widely accepted by mobile users. Most of existing systems adopt a spatial keyword search based method to help users retrieve location-aware answers. Given a set of points-of-interest (POIs) with spatial information and textual description and a user query with location and keywords, spatial keyword search finds top-k relevant objects by considering the distance and textual relevance between the query and objects. For example, if a user wants to find a gas station nearby, she can issue a keyword query "`gas station`" to an LBS system, which returns the relevant gas stations by considering the user's location and keywords.

However, existing systems have some limitations [1]. First, their search paradigm is not user friendly. They require users to type in complete queries and cannot support fuzzy matching. For example, a user is searching for a parking place, she has to type in complete keywords, e.g., "`parking place`" or "`parking lot`". The results will return only after she clicks the search button. She has to search multiple times until the results match what she expected. This search paradigm wastes users' time and is inconvenient to use. To address these problems, we enable location-aware instant search and location-aware similarity search in spatial databases. The former provides users with instant answers as users type in queries letter by letter. The latter tolerates the fuzzy matching between the query keywords and the underlying data. Obviously these two features boost the search efficiency and improve the experiences for mobile users who often misspell the keywords due to the limitation of the mobile phone's keyboard.

In addition, in many applications, users have direction-aware search requirements. For example, a driver on the highway wants to find the nearest gas station or restaurant. She has a search requirement that the answers should be in front of her driving direction. As modern smart phones are equipped with GPS and compass, we can easily get user's location and direction. Thus we can use them to provide better search experiences. To this end, we propose direction-aware search for mobile users which can efficiently find relevant answers in the specific directions.

Second, existing systems cannot support moving queries efficiently. For example, a mobile user is moving and wants to search for restaurants. If we can use the answers of previous queries to answer the subsequent queries, we can decrease the number of submitted queries to the server and also reduce communication cost between the client and server. To this end, we enable continuous keyword search in spatial databases. In addition to finding top-k answers of a moving query, we also calculate a safe region such that if a new

SIGMOD'13, June 22–27, 2013, New York, New York, USA.

query with location falling in the safe region, we can directly use the answer set to answer the query.

Third, Web users generate large amount of data (e.g., Foursquare) and we can utilize them to enrich the POI data and provide better search experiences. We crawl location-based user-generated content from the Web, and integrate them to provide users with rich well-structured data.

To address these limitations, we demonstrate a novel search system, called TsingNUS, with the following unique features. (1) TsingNUS integrates rich structured data from the Web and utilizes them to provide better search experiences. (2) TsingNUS has three user-friendly search paradigms, location-aware instant search, location-aware similarity search, and direction-aware search, which can significantly save users' typing efforts and improve the search experiences. (3) TsingNUS supports continuously moving search, which can relieve the server burden and reduce the communication cost between the client and server. Using these techniques, we have implemented and deployed a real system, available at http://tsingnus.cs.tsinghua.edu.cn, which has been commonly used and widely accepted.

2. SYSTEM IMPLEMENTATION

2.1 System Overview

The TsingNUS system includes three main components: location-based data integration, location-aware search engine, and user-friendly search interfaces. Figure 1 shows the architecture of our TsingNUS system.

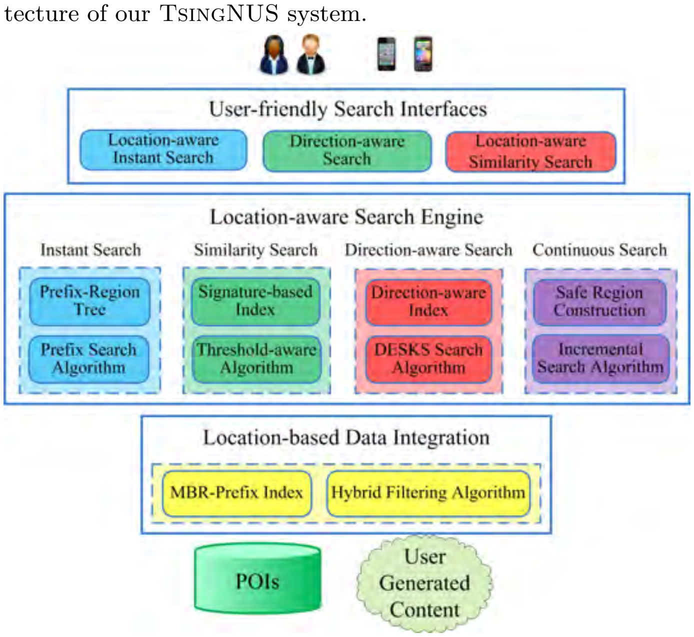

Figure 1: TsingNUS Framework

Location-based Data Integration: We first crawl and integrate location-based user-generated content (UGC) from the Web, including Foursquare, Twitter, etc. We then integrate the UGC data and existing POIs. If UGC data can match existing POIs, we extract the structured data and link them with the POIs. In this case, we can enrich existing POIs with detailed structured information. If UGC data cannot match existing POIs, we take them as new POIs. The main challenge is location-aware data integration, which links location-based data from two different sources (Section 2.2). The data integration component provides users with abundant information and better search experiences. We have more than 17 million POIs in China and 16 million POIs in USA. With these data, users do not need to search again and again in search engines to acquire details. Instead, in our system users obtain not only a map showing the POIs results, but also the detailed information when they click on the results.

User-friendly Search Interface: We provide three unique location-aware search interfaces, including location-aware instant search, location-aware similarity search, and direction-aware search, to help users easily find relevant answers.

Location-aware Search Engine: To efficiently support the three search paradigms, we devise effective indexing structures and search algorithms. Section 2.3 discusses how to support instant search and Section 2.4 describes the techniques for supporting similarity search. Direction-aware indexes and search algorithms are presented in Section 2.5. We discuss how to support continuous search in Section 2.6.

2.2 Location-aware Data Integration

With the popularity of smart phones, people nowadays generate large amounts of spatio-textual data which contain both spatial location information and textual descriptions. For example, people submit tweets with their current locations or people upload pictures of certain places with geological information. Those data are user-generated content. Since the spatio-textual data may have various representations, possibly because of deviations of GPS or different descriptions, it calls for efficient methods to integrate the data contributed by users from different sources. Thus given two sets of location-based objects (with textual descriptions and locations), the location-aware data integration identifies all the similar pairs that refer to the same entity. The similarity between two objects is quantified by combing the textual similarity and the distance of two objects. Since there may be large numbers of location-based objects, it is challenging to support similarity join by considering both textual descriptions and distance information.

To address this challenge, we develop a filter-and-refine framework and devise several efficient algorithms [5]. We first generate spatial and textual signatures for the objects and build inverted index on top of these signatures to avoid redundant computations. Then we generate candidate pairs whose signatures are similar enough using the inverted lists. Finally, we refine the candidates and generate the final result. To generate high-quality signatures for spatial information, we develop an MBR-prefix based signature to prune large numbers of dissimilar object pairs. For each object, it selects a subregion of the object as a spatial signature to substitute the entire region. We devise effective hybrid filtering algorithms to fully utilize the textual pruning and spatial pruning to improve the performance.

2.3 Location-aware Instant Search

Instant search is a recent trend and recently Google has been deployed to support instant search. However, it cannot support location-aware instant search, which given a query location and a prefix query, finds top-k objects with a keyword having the prefix and nearest to the query location. Roy et al. [6] proposed location-aware type-ahead search. However, their method cannot support multi-keyword queries and also consumes a large amount of memory. The main challenge of location-aware instant search is to achieve high interactive speed on spatial databases by considering both spatial information and the prefixes of POIs. Since there may be large numbers of objects that satisfy the prefix matching constraint, it is rather expensive to extend existing instant search methods to spatial databases.

To address this challenge, we propose a novel index structure, prefix-region tree (called PR-Tree), to efficiently support location-aware instant search [8]. PR-Tree is a tree-based index structure which seamlessly integrates the textual descriptions and spatial information to index the spatial data. Each node on the PR-tree has both a rectangle for the spatial information and a prefix for the textual information. It can divide the space using both spatial and textual information. Using the PR-Tree, we develop efficient pruning algorithms to support single prefix queries and multi-keyword queries. PR-Tree can help to achieve high efficiency.

2.4 Location-aware Similarity Search

Similarity search is an effective way to tolerate the inconsistencies between user's queries and the underlying data. For textual data, we can use existing similarity functions, e.g., Jaccard and Edit Distance, to quantify the similarity between query keywords and the data. For location-based data, we need to define new functions to quantify the similarity. For POIs, we combine the distance information between two POIs and the textual similarity to define the location-aware similarity. Nowadays, modern LBS applications generate a new kind of spatio-textual data, regions-of-interest (ROIs), containing region-based spatial information and textual descriptions, e.g., mobile user profiles with active regions and interest tags. For ROIs, we combine spatial overlap and textual overlap to define the similarity. Thus, given a set of spatio-textual objects and a query, location-aware similarity search finds the similar objects whose similarity to the query is no smaller than a given threshold. For this feature, it calls for an efficient search method to support large scales of spatio-textual data in LBS systems. The challenge is to efficiently find relevant answers by considering the hybrid similarity. To address this challenge, we propose a filter-and-verification framework to compute the answers [2]. In the filter step, we generate signatures for the data and the query, and utilize the signatures to generate candidates whose signatures are similar to that of the query. In the verification step, we verify the candidates and identify the final answers. To achieve high performance, we generate effective high-quality signatures, and devise efficient filtering algorithms as well as pruning techniques.

2.5 Direction-Aware Search

Direction-aware search finds relevant objects in a specific search direction, where the direction can be gotten from compass of mobile phones. A straightforward method first finds candidates without considering the direction constraint, and then generates the answers by pruning those candidates which invalidate the constraint. However, this method is rather expensive as it involves large amount of useless computation on many unnecessary directions.

To address this problem, we develop effective indexing structures and efficient algorithms to find top-k POIs that satisfy the search requirement [4]. We devise novel direction-aware indexing structures to maintain both the spatial and textual information. We develop effective distance pruning and direction pruning techniques to prune unnecessary regions. Given a query, we can deduce a direction range with a lower direction bound and an upper direction bound, pruning the objects outside the direction bounds. Based on the index and pruning strategies, we devise efficient search algorithms to achieve high performance. For the cases of changing search directions, we devise an incremental search algorithm to effectively and incrementally answer a query.

2.6 Continuous Search

Although we can extend existing methods to support moving queries by repeatedly issuing multiple queries, these methods have the following limitations. First, it increases the communication cost between the client and the server, and also wastes the bandwidth in transmission. Second, it aggravates the system burden due to issuing multiple repeated queries. To address these problems, we propose an effective method to support moving top-k spatial keyword queries [3]. An effective way to support moving queries is that in addition to finding top-k answers of a moving query, we also calculate a safe region such that if a new query with a location falling in the safe region, we can directly use the answer set to answer the query. Before the client issues a new query at another location, the system will first check whether the new location is still in the safe region. If yes, it can reuse the answer set; otherwise, the client needs to issue a query with the new location to the server.

There are several challenges in computing the safe regions. The first challenge is how to represent the space region. Although some existing works utilize Voronoi Diagrams to represent the safe region. However, they did not consider the textual information. Wu et al [7] proposed to generate safe regions by considering the textual and spatial information. However, it only supported ad hoc functions while we propose new techniques to support widely used functions. Our method is much more efficient than their method. To address this challenge, we propose to use hyperbola to represent safe regions. We use polar coordinates to describe hyperbola which can facilitate computing the overlaps of different hyperbolas. The second challenge is to efficiently compute the safe region. To address this challenge, we utilize existing indexing structures to facilitate the pruning. We also propose incremental search algorithm to cache relevant information, avoiding repeated computations. We propose an approximate method which computes the local safe region. Although the local safe region is smaller than the global safe region, it can improve the performance of computing the safe region and boost the search performance.

3. DEMO SCENARIOS

We demonstrate our TsingNUS system in five scenarios. We have implemented a desktop system[1] and a mobile system on Apple iOS platform[2].

Scenario 1 - Location-aware Instant Search[3].

Suppose a smart phone user is searching for a restaurant with keywords "`Pizza Papa`". However she does not know the exact name of the restaurant. Therefore, she tries several times before she finally obtains the intended results. For

[1] http://tsingnus.cs.tsinghua.edu.cn
[2] http://itunes.apple.com/us/app/sword-map/id518593301?ls=1&mt=8
[3] http://tsingnus.cs.tsinghua.edu.cn/usa/

example, she may type "`Pizza Mama`" or "`Pizza Dad`". She has to wait every time after entering the complete keywords and keeps issuing queries if the results returning to her are not satisfying. However, in our system, she does not need to wait and click the search button. The results appear immediately when she types "`P`" and update to new results when she types more, helping users to save typing efforts.

What is more, for a mobile user, if she wants to find the nearest "`Costa Coffee`" in front of her but does not know the exact spelling of the keywords. Using the traditional search systems, she has to try several queries until she types in an appropriate one. It is also very common for mobile users to type wrong since the keyboard on a smart phone is small and it is easy to touch the adjacent letters. For these limitations, users may find many irrelevant results and take long time to get the expected answers. In contrast, users in our system can get the relevant results as they type the queries letter by letter. Thus users can browse and check the answers while typing queries, and this search-as-you-type paradigm can significantly save users typing efforts.

Scenario 2 - Location-aware Similarity Search[4].

Suppose a user wants to find a "`Burger King`" near her office. When she inputs the keywords, the traditional search may return the most relevant results but they may be quite far away from her office. Therefore, the results cannot satisfy users' requirements. She needs to modify the keywords and then searches again. On the contrary, our similarity search can overcome this problem. Our search engine can tolerate errors which first generates candidates based on the similarity scores in terms of both distance and textual descriptions, and then verifies the final results. She may input "`Burget Kimg`", while still obtaining relevant results.

When changing the dataset into social networks dataset, our system can be transformed into a friend recommendation system. Spatio-textual similarity search helps mobile users find potential friends with common interests (e.g., playing basketball) and overlap regions (e.g., Brooklyn), and thus facilitates users to form various kinds of circles with the same interests, such as sport games and fans' activities.

Scenario 3 - Direction-aware Search[5].

Recall the example of finding a "`gas station`" on the highway. Traditional spatial keyword search systems do not consider the direction constraints; therefore, the nearest gas station returned to the user may be behind her driving direction. In this way, she has to repeatedly issue different queries until she finally finds an appropriate gas station. In contrast, using TsingNUS, she can directly get the results that are in her direction constraints. If a user and her friends want to find a "`restaurant`" using the traditional system, they can only find the top-k restaurants. Those restaurants may be far away from each other, which makes it inconvenient for them to move to another if the current restaurant does not fit their tastes. Using our system, after the search, they can change directions first to find an appropriate direction that has more relevant restaurants near them and see the detailed information. Therefore, they can easily find the best direction to go. In this way, our system can give users a quick view and save much time.

In addition, suppose that a user wants to find an "`ATM`" and she is quite near the results. If the streets are complicated and there is no obvious sign for the ATM, she may spend long time, e.g., half an hour, finding it. For the traditional system, users have no choice but checking all the possible routes. For our system, on the other hand, users can put the phone in the direction of a certain street and check if the results are in this street. This special feature enables users to walk less and find the results faster.

Scenario 4 - Continuous Search[6].

Suppose that a mobile user visits a city for the first time on a business trip, searching for a "`hotel`". She wants to find several hotels and compares them to decide which one is better. It is possible that she is in a taxi or some other kinds of transportation. While moving to a new location, she has to issue the query again using the traditional search method. However, for our system, she can search for once and the results can be automatically updated with her movements. The search speed is fast since our framework uses a safe region to reduce the connection cost. The client needs to issue new query only when it goes out of the safe region. What is more, with the abundant data, she can easily compare different hotels and choose the favorite one.

As another example, a housewife is driving to a supermarket and may want to find "`car parking places`" near the supermarket. Since she is driving, her query location is continuously changing. She can use our method to find results. After issuing the query, she can follow the results and keep driving until finding the best parking lot.

Scenario 5 - Rich Structured Data[7].

Suppose that a database researcher wants to attend SIGMOD 2013. She wants to known about the conference hotel and nearby restaurants. She needs to use the search engines to search relevant information multiple times. However, in our system, we search the POIs with structured data and she can get all information through our system. Although Google map search can provide related reviews, it cannot provide high-quality structured data of a POI.

Acknowledgement: This work was partly supported by the National Natural Science Foundation of China under Grant No. 61272090, National Grand Fundamental Research 973 Program of China under Grant No. 2011CB302206, and project of Tsinghua University under Grant No. 20111081073, and the "NExT Research Center" funded by MDA, Singapore, under Grant No. WBS:R-252-300-001-490.

4. REFERENCES

[1] G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1):337–348, 2009.
[2] J. Fan, G. Li, L. Zhou, S. Chen, and J. Hu. Seal: Spatio-textual similarity search. *VLDB*, 2012.
[3] W. Huang, G. Li, K.-L. Tan, and J. Feng. Efficient safe-region construction for moving top-k spatial keyword queries. In *CIKM*, 2012.
[4] G. Li, J. Feng, and J. Xu. Desks: Direction-aware spatial keyword search. In *ICDE*, pages 474–485, 2012.
[5] S. Liu, G. Li, and J. Feng. Star-join: Spatio-textual similarity join. In *CIKM*, 2012.
[6] S. B. Roy and K. Chakrabarti. Location-aware type ahead search on spatial databases: semantics and efficiency. In *SIGMOD Conference*, pages 361–372, 2011.
[7] D. Wu, M. L. Yiu, C. S. Jensen, and G. Cong. Efficient continuously moving top-k spatial keyword query processing. In *ICDE*, pages 541–552, 2011.
[8] R. Zhong, J. Fan, G. Li, K.-L. Tan, and L. Zhou. Location-aware instant search. In *CIKM*, 2012.

[4] http://tsingnus.cs.tsinghua.edu.cn/
[5] http://tsingnus.cs.tsinghua.edu.cn:8080/directionsearch/
[6] http://tsingnus.cs.tsinghua.edu.cn:8080/movingsearch/
[7] http://tsingnus.cs.tsinghua.edu.cn/