Exploiting Geographical Location Information of Web Pages

Orkut Buyukkokten[*] Junghoo Cho[*] Hector Garcia-Molina[*]
`orkut@cs.stanford.edu` `cho@cs.stanford.edu` `hector@cs.stanford.edu`

Luis Gravano[†] Narayanan Shivakumar[*]
`gravano@cs.columbia.edu` `shiva@cs.stanford.edu`

Abstract

Many information resources on the web are relevant primarily to limited geographical communities. For instance, web sites containing information on restaurants, theaters, and apartment rentals are relevant primarily to web users in geographical proximity to these locations. In contrast, other information resources are relevant to a broader geographical community. For instance, an on-line newspaper may be relevant to users across the United States. Unfortunately, the geographical scope of web resources is largely ignored by web search engines. We make the case for identifying and exploiting the geographical location information of web sites so that web search engines can rank resources in a geographically sensitive fashion, in addition to using more traditional information-retrieval strategies. In this paper, we first consider how to compute the geographical location of web pages. Subsequently, we consider how to exploit such information in one specific "proof-of-concept" application we implemented in JAVA, and discuss other examples as well.

1 Introduction

The World-Wide Web provides uniform access to information available around the globe. Some web sites such as on-line stores [1, 2, 3] and banking institutions are of "global" interest to web users world-wide. However, many web sites contain information primarily of interest to web users in a geographical community, such as the Bay Area or Palo Alto. Unfortunately, current web search engines do not consider the *geographical scope* of the pages when computing query results. Consequently, search engine query results often include resources that are not geographically relevant to the user who issued the query. For instance, finding restaurants, theaters, and apartment rentals in or near specific regions is not a simple task with current web search engines.

Now consider the scenario in which we have a database with the geographical scope (e.g., a zip code, a state) of all "entities" (e.g., restaurants, newspapers) with a web presence. We can then exploit such information in a variety of ways:

To improve search engines: When a web user queries a search engine for "Italian restaurants," the search engine first identifies where the user is from. For example, the user may have registered this information at the search engine in a profile (e.g., as with the `my.yahoo.com` or `my.excite.com` services). The search engine then uses this information to rank Italian restaurants based on the distance of the restaurant from the user's location, rather than returning references to all Italian restaurants in the world. Note that this strategy is not equivalent to the user querying the search engine for "Italian restaurant" AND "Palo Alto," since such a query would miss references to restaurants in areas close to Palo Alto such as Mountain View and Menlo Park. Of course, if additional information such as the restaurant quality or pricing information is available, the search engine can factor these dimensions into the ranking as well.

---

[*] Department of Computer Science, Stanford University, Stanford, CA 94305.

[†] Department of Computer Science, Columbia University, 1214 Amsterdam Ave., New York, NY 10027.

To identify the "globality" of resources: A web user often wants to find "important" web pages, or web pages that are an "authority" on a given topic. For this reason, web search engines such as Google [4, 5] and HITS [6, 7] count the number of distinct hyperlinks that point to a given web page as an indication of how important the web page is. The rationale for such a heuristic is that the larger the number of web users who made a hyperlink to the web page, the higher must be the importance of the page. While this approach is often useful for certain applications, we believe such a count is often not a good indication of the "global" relevance of a site. For instance, many Stanford University students and Palo Alto residents have links to the Stanford Daily (the daily campus newspaper) since it covers news relevant to the local community. However, the number of hyperlinks to a "globally" important newspaper such as The New York Times may be smaller simply because the concentration of web sites in Stanford University and Palo Alto is one of the largest in the world. We believe that by exploiting geographical information about web sites, we can estimate how global a web entity is.

For data mining: Market analysts are often interested in exploiting geographical information to target product sales. The web provides a new source of such information. For instance, if a market analyst identifies from web data that outdoor activities such as hiking and biking are popular among the residents of the Bay Area and Seattle, the analyst may recommend opening additional sporting goods stores in these areas.

The following principal problems arise when we try to exploit geographical location information of entities (such as restaurants):

1. **How to compute geographical information?** Section 2 discusses some techniques for determining the location of web pages. Our techniques primarily differ in how *easy* it is to compute location information versus how *accurate* the corresponding information is.
2. **How to exploit this information?** Once we have a database of location information, the second question is how to use this information. We discussed earlier a few applications in which we can use such information. Section 3 concentrates on one such application and discusses an initial visualization prototype that we have developed for this specific task.

2 Computing Geographical Scopes

Ideally, we would like to identify the geographical scope of all web pages. An approach to computing these geographical locations would be to have a human "hand-classify" pages. Of course, this approach will be prohibitively expensive. Hence, we need to explore automatic ways of attaching geographical scopes to on-line resources.

Information Extraction: We can automatically analyze web pages to extract geographic entities (e.g., Palo Alto), area codes, or zip codes. When a user types in some search terms into a search engine, the engine can denote the geographical location in "close textual proximity" to the search terms to be the location of the search terms. For instance, consider a web page with the words "Florentine's Italian Restaurant, Menlo Park." If a web user in Palo Alto searches for Italian restaurants, we can associate Menlo Park to be the "closest" geographical location of the search terms "Italian restaurant" for that particular web page. The search engine can then rank the web page to be "close" to the user in Palo Alto. This approach is complementary to the approach that we present in this paper. However, our approach attempts to capture the geographical scope of the web resources from their *actual usage*, not just from the words and places mentioned on them.

Network IP Address Analysis: A related approach for determining the geographical scope of web resources is to focus on the location of their hosting web sites. Thus, we can define the geographical scope of a given web page to be "around" the location of the web site where it resides. For instance, consider the web page at `http://www-db.stanford.edu`. From the URL, we can query the database and assume the web page is relevant to the Palo Alto or Stanford regions. (As a side note, we have not found such a database despite repeated requests to INTERNIC and ICANN, the Internet domain registries. However, we can use UNIX tools such as `whois`, `nslookup`, or `traceroute` to identify `www-db.stanford.edu` to be located in Palo Alto, California.)

This approach suffers from three problems. Firstly, the above heuristic does not work in all cases. For instance, the geographical scope of a web page hosted by `www.aol.com` often has no correspondence to where the AOL web server is located. Secondly, while `http://www-db.stanford.edu` is located at Stanford, its content may be globally relevant. Finally, IP addresses are often allocated in an ad-hoc fashion and there is usually no inherent connection between an IP address and its physical location. In this context, tools like `whois`, `nslookup`, and `traceroute` do not always give us the location of a site [9]. Also these tools are slow and typically take between one and ten seconds to respond per web site. Such tools will not scale when we need to find the locations of hundreds of thousands of web sites.

In the next section, we show how we can find the location of a web page accurately and efficiently, especially for educational institutions with a `.edu` extension. We also describe how we can exploit this information to determine the geographical scope of web resources.

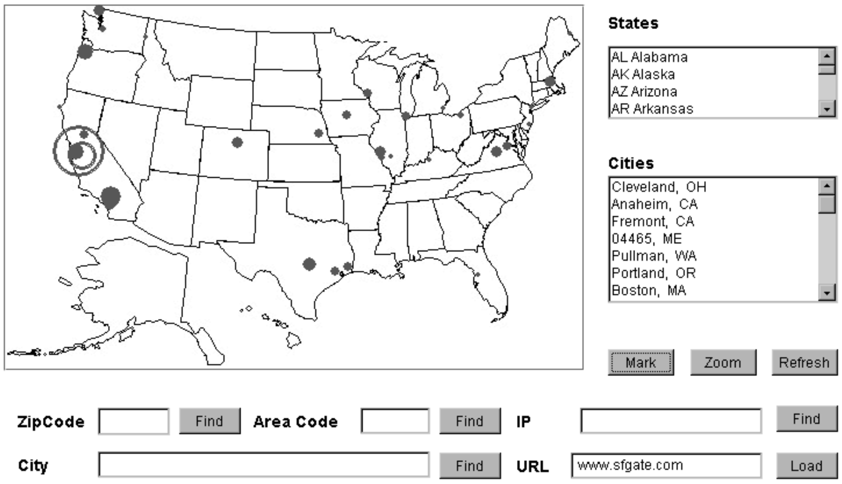
3 Prototype Application

We discussed in Section 1 a variety of applications that benefit from geographical location information. In this section, we first explore how we can map a web page to the geographical location where it originated. We then use these mappings to understand the geographical scope of web resources. For this, we developed a proof-of-concept prototype that shows how we can use geographical information to determine the "globality" of a web site.

To map web pages to their geographical location, our prototype uses a number of databases that we downloaded from the Internet. We needed these databases because tools like `whois` are too slow to compute the geographic locations of hundreds of thousands of web sites.

1. **Site Mapper:** We downloaded from `rs.internic.net` a database that has the phone numbers of network administrators of all Class A and B domains. From this database, we extracted the area code of the domain administrator and built a `Site-Mapper` table with area code information for IP addresses belonging to Class A and Class B addresses. For example, the original database contained a phone number with a (650) area code for the administrator of the 171.0.0.0 (Stanford) Class A domain addresses. Based on this information, we maintained in `Site-Mapper` the area code information for all the `171.*.*.*` IP addresses. We handled Class B addresses analogously.

2. **Area Mapper:** We downloaded from `www.zipinfo.com` a database that mapped cities and townships to a given area code. In some cases, entire states (e.g., Montana) correspond to one area code. In other cases, a big city often has multiple area codes (e.g., Los Angeles). We wrote scripts to convert the above data into a table with entries that maintained for each area code the corresponding set of cities/counties.



URL www.sfgate.com

Figure 1: Geographical distribution of hyperlinks to `www.sfgate.com`.

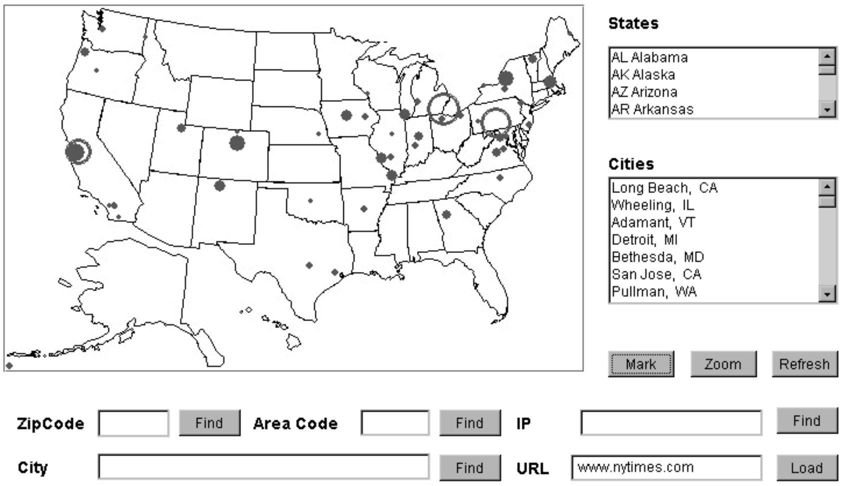
3. **Zip-Code Mapper:** We downloaded from `www.zipinfo.com` a database that mapped each zip code to a range of longitudes and latitudes. We also downloaded a database that mapped each city to its corresponding zip codes.

From the above databases, we computed for each IP address the "average" latitude and longitude of where the web site was located. For instance, when an area code covers an entire state, the longitude and latitude for an IP address in that state is the longitude and latitude averaged across the extent of the state. As we discussed earlier, the above process does not always yield the correct answer for arbitrary IP addresses. However, we found that this technique works very well in practice for educational institutions (with `.edu` extensions) since universities have all the IP addresses in a Class A or Class B domain allocated to web sites within the campus. Hence we restricted our database to educational institutes, and obtained accurate geographical information for web pages located in these institutes.

Given the above geographical database, we built a GUI in JAVA that performed the following tasks on a map of the United States.

- Given any latitude and longitude, the prototype places a point at the corresponding location on the map.
- Given a city name, the prototype places points on the corresponding locations on the map. (Many cities have the same name.)
- Given an IP address or the corresponding textual URL, the prototype places the site on the corresponding geographical location on the map.

With the above infrastructure in place, we built our prototype on top of the Google web search engine [4]. When a user types in a URL into our system, the prototype issues the "`link:`" query to Google through `http GET` requests, to retrieve the set of pages in educational sites that have a hyperlink to the given URL. For instance, when a user types in `www-db.stanford.edu`, the



URL www.nytimes.com

Figure 3: Geographical distribution of hyperlinks to `www.daily.umn.edu`.

Our prototype is a promising initial example of how we can build useful applications that exploit geographical information in a meaningful fashion. In the future, we will explore using this information for other applications, including the ranking of search-engine query results. We also plan to investigate how to identify the geographical location of web resources using the content-based information extraction approaches we mentioned briefly. Specifically, we plan to compare the information extraction approach against the network IP address based approach we adopted until now, and understand how the techniques differ. For this, the Stanford Database Group has built a web crawler that has archived over 30 million web pages. We plan to implement and run our geography extractors on this data soon.

References

[1] Amazon Bookseller. `http://www.amazon.com`.

[2] Barnes and Noble Bookseller. `http://www.barnesandnoble.com`.

[3] Jango Comparison Shopper. `http://jango.excite.com`.

[4] S. Brin, L. Page. Google Web Search Engine. `http://www.google.com`.

[5] S. Brin, L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International World Wide Web Conference (WWW7)*, Apr. 1998.

[6] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, Jan. 1998.

[7] S. Chakrabarti, B. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, and J. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. In *Proceedings of the Seventh International World Wide Web Conference (WWW7)*, Apr. 1998.

[8] G. Salton. Introduction to Modern Information Retrieval. *McGraw-Hill, New York*, 1983.

[9] Uri Raz. Finding a host's geographical location. `http://t2.technion.ac.il/~s2845543/IP2geo.html`.