Design of a Social Networking Analysis
and Information Logger Tool

William Gauvin and Benyuan Liu
Department of Computer Science
University of Massachusetts Lowell
{wgauvin, bliu}@cs.uml.edu

Abstract. This paper describes a design and development methodology
used to facilitate the collection of information contained on Social Net-
working sites. It facilitates the ability to record data explicitly used by
individuals, using Web 2.0 technology, to host their profiles and allows
others to view, respond to, and share this information. The tool uses
unobtrusive methods, to elevate the problems of over burdening the So-
cial Network servers and to elude throttling and black-listing restrictions
commonly used by these services to prevent sources which do not act
as good network citizens in this environment. The information obtained
may be used to study the relationships of individual owners of a Web
site. It may also be used to predict the bandwidth requirements of a so-
cial networking site based on the amount of users and specific ‘activities’
heuristically modeled.

Key words: On-line Social Networks, Data Mining, Heuristics, Scan-
ing Techniques

1 SnailCrawler Overview

The overall objective is to focus on a specific social networking site, MyS-
pace.com, and provide a tool which will facilitate the classification blog ent-
tries and order the information in such a way that it can easily be analyzed to
understand the relationship between individual members of a group. It is used
to predict the level of interaction based on age, gender, publishing tendencies,
friends distribution and content used to express themself. Understanding the ‘de-
gree of membership’ of specific individuals, will determined trends which can be
categorized and used to predict future usage patterns, system and network pro-
visioning requirements and help to model communication patterns. Specifically,
the Social Network Analysis Information Logger and Crawler, snailCrawler, will
focus on analyzing social network relationships and allow these relationships to
be automatically ordered, monitored and categorized. Individuals will be iden-
tified using their personal MySpace ID, a unique identifier for their profile. All
individual blog entries will be analyzed to ascertain the amount of content shared
by all members of the social network. This information will include the number
of images, object references and content (the number of words used within the
SnailCrawler employs a web engine used to glean the electronic data off the social networking site, using Web crawling. For snailCrawler, the crawler is both a front-end and back-end engine. The front-end scans a range of MySpace user IDs to retrieve profile information. The specifics are identified below, the back-end scans the publishers once identified. For public sites that contain a friends list and blog entries, a JavaScript Robot (JBOT) is deployed to navigate each entry and collect information and statistics. Each blog within a public profile is scanned to retrieve the information of that specific blog. The ‘friends list’ is scanned, creating groups of users whom communicate with each other. Scanning is accomplished by further directing the friends site pages to a ‘JBOT’ which provides the ability to run the JavaScript used for MySpace navigation to scan the friends list. SnailCrawler, when given a MySpace target, will produce a data set to include a profile and the publisher social network that constructed the blogs for that owner. By obtaining the individual information of both the profile owner and publisher, an analytical representation of the activity occurring on that site is obtained.

SnailCrawler is a combination of multiple tool sets that are listed in figure 1. The framework for the tools is built around the Microsoft C# .NET WebBrowser component object class. This framework automatically provides the session management required for navigating through a users blogs, when required. The WebBrowser client is extended to provide a Profile Scan Engine, a Blog Scan Engine, Google Scan Engine, and a HTML Scrapper. In addition, snailCrawler provides a Report Generator used to analyze and generate GNU plots.

The basic objective of the tool is to provide the ability to scan any MySpace site to provide a dataset for statistical analysis, that is used to help derive specific information about generic publishing trends for both temporal and contextual
behaviors. The collection methodology takes into consideration the sequential ordering of MySpace profiles. Using this approach, a range of IDs can be randomly generated and then scanned to determine if a valid profile exists for that ID. There are three specific types of profiles that may be obtained during the collection of data, namely, Public, Private, and Custom. Public profiles allow general access to a MySpace members main page or wall for viewing. Private profiles do not allow general access to a MySpace members main page or wall. This type of account restricts the information provided about the owner to specific elements, such as name, age and location. You must befriend the owner to gain access to the profile content. As such, these sites cannot be scanned to obtain friends list and blog details. The above fore mentioned profiles use a standard template which allows the screen/HTML scrapping of the content to be easily obtained using specific keywords found within the rendered content. The members of MySpace have the option of applying non-standard, Custom templates, which make it difficult for screen/HTML scrapping to work efficiently. Therefore, special considerations for this type of profile have been made to identify those accounts which are not successfully scanned due to format inconsistencies, which generally leads to the inability to acquire all information that may be available.

An optimal profile is one that is marked as public, follows the standard profile template, and contains a number of blog entries. A Screen/HTML scraper is used to retrieve random profile samples from MySpace and extrapolate publishing information for each public site. The tool uses a three phased approach for the collection of data outlined in in Figure 2.

The main purpose of snailCrawler is the collection of public user profiles and their corresponding blogs, this phase is represented in figure 2, phase A. Both, profiles and blogs are scanned using the Session Parsing Engine. Once blogs are scanned, the corresponding publisher information is obtained using the Publisher Parsing Engine, figure 2, phase B. The final phase in the gathering of information is the use of the General Parsing Engine 2, phase C, which is a crawler friendly interface for gathering MySpace content. The general parser provides content that may not have been obtained from phase A or B parsing.

The tool provides the means to generate the random IDs that are used to identify the profiles scanned within the range of 1 to 1,500,000,000. The max-
num range identified by the scanner for MySpace is 1,310,258,402. During the scan, if the profile is public, the number of friends and/or blogs in the list are obtained. Non-zero friend and blog list, are collected using the navigational aspects of the session engine to index the pages for both the friends list and blog pages. Random IDs that result in a profile being identified as a private or a custom site are stored accordingly. The friends and blog list for a custom site may or may not be accessible dependent on the format used and the profile account type (Public or Private).

2 Analysis Methodology

The result of the the collection process is a dataset which may be for Gender, Age, Profile, Publisher, Demographic analysis and Network traffic predictions. An indexer is used to order and break up the profile collection into small, sequential files, to provide an efficient means to locate profiles based on their IDs. The name of the blog file corresponds to the actual ID of the profile and forms a cross reference to the profile owner of the same ID. Once the process outlined in ‘Collection Methodology’ is complete, an analysis is performed, the steps of which are defined in figure 3. Phase A of the analysis process is to merge the blog information with the publisher information obtained. The operation creates a merged object, which contains the content and publisher information. This consolidation allows the analysis engine to have all information required in one dataset for temporal, content, publisher gender and age analysis. Previous attempts, resulted in long processing times when consolidation was not used.

Once the merge process in phase A is complete, phase B provides the analysis of the data gathered and consolidated. In this phase, various arrays and matrix are used to ‘count’ the occurrence of interaction between owners and publishers or time related events, such as month, day or times of publishing. This information is ‘rolled-up’ using collection containers. Once this phase of the analysis is complete, phase C is used to generate the graphs required to provide a visual representation of the data by translating the collection objects into GNU plot data points.

Fig. 3. snailCrawler Analysis Process
3 Problems Encountered and Solutions

The main problem of building a Social Network Analyzer is managing the scale of the site, that contains million of users. These numbers are much too large to allow for a total scan of the site to obtain the information required for analysis. The approach taken was to identify a range of possible profile IDs, scan random entries within that range and build a friends list, which will also be scanned. Another goal was to create a scanner which would not in-adversely affect the hosting site (in this case MySpace). To accomplish this, each page scanned was rendered. This provided a running characteristic which looks much like an individual user of the site. Multiple instances of the scanner can be used; but it was limited to not more than 5.

Any page that did not follow a strict MySpace HTML format is treated as a custom page, tagged as such and any information available, obtained. To ensure enough information is available for analysis, General parsing is required; this introduces additional processing for both MySpace and the collector. Exceptions occur when the MySpace server fails to respond to a request, causing timeouts. These exceptions occur when an image being requested was no longer available or some references within the blogs did not follow a normalized format. A watchdog timer provides the solution to this problem, and allows the scan to skip to the next profile ID. A major problem with scraping technologies is the requirement to 'keep up' with the tags being used to identify the parsing points within HTML. Generally the change is minor, such as incrementing a control tag from ctl00 to ctl01. This leads to the stoppage of the scan, with the requirement to determine were the scan should be re-started after the problem is corrected. Adding the ability to track the last known good scan allows the scan to continue at the failure point once the problem is resolved. In addition, the tag parsing was generalized such that minor changes would allow the scanner to continue to run. The HTML format and navigation process for the US MySpace site is different than the Chinese site. This lead to the development of locale detection to determine which site was hosting the desired profile and use the correct format when parsing and navigating.

Session timeouts were experienced for two types of problems, one normal MySpace session duration timeouts, the other, dialogbox response timeouts. The ability to restart the scan at the point of the session timeout was added to make the restarts seemless after completion of the login page. The second problem was more problematic, while scanning user profiles, just accessing the page would sometimes result in either a user programmed dialog box to be displayed or a plug-in request to install, or set the application as the default. For application level dialog box request, the ability to ignore/disable the query was used. For user programmed dialog boxes, the only alternative was to issue the user response to continue. The generally lead to another session timeout problem with the same symptoms as listed above. An unfortunate aspect of the scanner is the requirement to fetch images twice. The double fetching of images causes a performance problem for the scanner and the site the image is located on. The double fetch is a result of the requirement to obtain the size of images within blogs. The
problem is compounded when the image no longer exists and a timeout occurs. A solution would be to use a caching proxy which would sit between the scanner and the server, this was desired, but not implemented.

4 Conclusions

The ability to analyze the individual blogs of a Social Network can be used for both network traffic predictions as well as to obtain social indicators for behavior analysis. Determining who and how users use the social networking sites may help applications better meet the requirements for those users and provide a better 'ease of use' for features desired. Because activity is generally event driven, the popularity of a social networking tool may be increased by providing a richer feature set which entices its users to participate in the use of the service provided.