

Extracting Cyber Communities Through Patterns

Tassos Argyros* Charis Ermopoulos* Vassiliki Pavlaki* Nidal Al-Said*

Abstract

This paper proposes a new approach to the problem of extracting Web communities. Due to the variety of topics found on the Internet, discovering online communities becomes a really difficult task. In our method, we exploit the observation that pages across the Web refer to other pages of a community using similar text templates in their links. Therefore, by creating patterns that describe these templates we can find a large subset of the pages of a specific community.

1 Introduction

The World Wide Web is a vast source of information scattered across millions of pages. The application of methods exploiting only the link structure of the Web, or just keywords of the site content, does not always lead to satisfactory results (see further §2).

The algorithm we propose for that purpose is based on a different assumption. We assume that authors of Web pages, belonging to a specific community, refer in their links to other pages of the same community in a similar way. This means that the text found near and on the links, that point to similar sites, usually follows some patterns. Our algorithm's goal is not to locate the whole community, but a sufficiently large number of sites belonging to a specific community. The following example will demonstrate in a representative, but somewhat simplifying way, how an algorithm of this type should work in general.

As an example, let us consider the problem of extracting an on-line chess playing community. We bring forth some links (with the anchor text underlined) that are fictional but representative of the reality.

Link 1: ...on-line chess tournament Kasparov's Chess Tournament, supported by FIDE ...

Link 2: ...visit the ultimate chess portal, chess replays, tournaments, openings ...

Link 3: ...watch on-line chess replay Kasparov vs Deep Blue the rematch ...

Link 4: ...world's chess leader Kasparov arguing against FIDE and IBM ...

The above links point certainly to Web pages belonging to the community we seek. Our goal is to use these links in order to find more pages of the given community. By studying those 4 links more carefully we notice that links 1 and 2 refer in a common way to an on-line chess community page, yet links 3 and 4 refer also, though in a rather specific way, to on-line chess community pages. Notice that all links contain some specific common words like "chess", "on-line", "Kasparov", "Tournament", "FIDE" etc. The most general type of links (links 1 and 2) can be pointed out by choosing those that contain the greatest number of common words with the others. Comparing links 1 and 2 with our whole database (potentially the whole Web) we can point out a new set of links that contain enough common words with one of them (link 1 or link 2). This new set of links will be expected to point at members of the given community. The common words of the new set of links will now be slightly different and may contain new common words like "PCA" (Professional Chess Association), "grandmaster" etc. such as the links below:

Link 5: ...on line chess discussion by FIDE, PCA and world's chess leader Kasparov ...

Link 6: ...grandmaster Kasparov beaten by Deep Blue on the early opening ...

By choosing, the same way as before, the most "general" links, we may create a new set of links that contain again enough common words with one of them (link 5 or 6) and thus create a new population of links hopefully pointing at sites of the given community. The Web pages that will appear in the new population will probably be somewhat different, since some of them might contain completely different common words from the links we used in the beginning. This way, pages from the same community will arise, which would be difficult to have been pointed out in some other way.

In other words, starting with a link containing the words "chess" and "FIDE" we may be lead to many new links containing the words "grandmaster" and "PCA", which are indeed very common in the

*National Technical University of Athens, ECE department. e-mail: targyros@cslab.ece.ntua.gr, ermop@softlab.ece.ntua.gr, vapulaki@softlab.ece.ntua.gr and nidal@softlab.ece.ntua.gr.

pages of the community we are looking for. The algorithm itself recognizes correctly that these new words “grandmaster” and “PCA” have to do with the community we are looking for (since they are common) and thus it uses the corresponding links as pattern links for the following step. The procedure described above can be continued iteratively for a number of steps until a sufficient number of Web pages have occurred.

2 Related Prior Work

The basic approach to extracting communities from the Web is the exploitation of the link structure of Web pages. According to Gibson, Kleinberg and Raghavan [1] communities can be viewed as containing a core of central, “authoritative” pages linked together by “hub pages” and they exhibit a natural type of hierarchical topic generalization that can be inferred directly from the pattern of linkage. Algorithms that discover “hubs” can therefore be used in extracting communities.

Other work aiming at extracting on line communities is [5] by G. Flake, S. Lawrence, C. Giles and F. Coetzee, and [6] by the first three of these authors. In these papers, a community is defined as a set of pages that have more links (in either direction) to members of the community than to non-members. Another attempt to discover cyber communities using a graph-theoretic approach (utilizing link structure) has been made by Kumar, Raghavan, Rajagopalan and Tomkins in [2] (see also [3] and [4]).

Moreover, M. Toyoda and M. Kitsuregawa [7] propose a different approach to this problem by creating a Web community chart that connects related Web communities from thousands of seed pages. The chart is created by applying the algorithm to each seed, then using similarities of the results to classify seeds into clusters and to deduce their relationships. See also: [8, 9].

As far as patterns are concerned in the Web content mining field, to our knowledge, mainly S. Brin has attempted to experiment with patterns and relations. In his paper [10], Brin used patterns to extract a large database of books (title-author) from the Web, beginning with a small sample (five pairs of author-title).

The results of the above paper where both accurate and extend. However, the proposed method works only when the number of ways of reference is limited. For example, there are not many ways to refer to a book using the title and the author. On the contrary, consider a community of chess-related Web pages. There certainly is an extremely large number of ways to refer to a page of this community. Therefore, any attempt to create exact patterns of these references will fail.

3 Description of the Problem

Before we proceed to the presentation of the algorithm, we should first clearly define the problem it deals with and the hypothesis upon which it is based. The goal of the algorithm presented in this paper is to locate a sufficiently large number of Web pages belonging to a specific community. By the term *community* we define a set of Web pages dealing with a common specific subject.

The basic assumption upon which the algorithm is based can be formulated in the following way: there exist certain patterns, based upon authors of Web pages refer in their links to Web pages belonging to the same community. It is a fact that when a link appears, the text near the link, let alone the anchor text of the link, aim to describe the content of the page being linked. In that way, the anchor text and the surrounding text can provide a very precise and short description of the site being linked. The prior assumption is based on the observation that the descriptions of sites belonging to the same community are similar and so they can be considered as following a specific pattern.

Naturally, there exist some cases when the above does not hold, for example when pages of different languages belong to the same community. However, practice and experiments show that this assumption is accurate to an adequate extend.

4 Definitions and Method

Our algorithm is based on the concepts of tuples and patterns and their relation. Thus, before we proceed, we must define exactly the above.

4.1 Definitions We first define the “words” that we use.

DEFINITION 4.1. (WORD) *Whenever we refer to a “word” (in the context of a Web page) we define any continuous sequence of characters in the range $[a, z] \cup [A, Z] \cup [0, 9]$. Furthermore, we do not consider a “word” any stop words (like “when”, “and”, etc.) or any part of HTML or other kind of code (for example script languages) that might be part of a Web page.*

Below follows the definition of tuples.

DEFINITION 4.2. (TUPLE) *Given a link on a Web page, we define the tuple of the link as a vector containing five fields: (**page_url**, **prefix**, **anchor_text**, **suffix**, **link_url**) where: **page_url** is the URL of the page that the link rests, **prefix** consists of at most four words that appear in the Web page just before the link, **anchor_text** is the anchor text of the link, **suffix** consists of at most four words that appear in the Web*

page just after the link and finally, *link_url* is the URL of the page that the link points to.

Tuples are found using specific patterns, and patterns are extracted from sets of tuples. Here, we define what we mean when we refer to a “pattern”.

DEFINITION 4.3. (PATTERN) We define a pattern to be a vector of three fields: (*prefix*, *anchor_text*, *suffix*) where *prefix* and *suffix* are sets of (at most) four words, and *anchor_text* is a set of words.

We now define when a tuple “matches” a specific pattern along with the concept of extracting a pattern from a set of tuples.

DEFINITION 4.4. (MATCH) A tuple matches a specific pattern if at least one of the fields of the tuple (*prefix*, *anchor_text*, *suffix*) contains at least two common words in arranged order with the respective field of the pattern.

DEFINITION 4.5. (EXTRACTION) A pattern is considered to be extracted from a set of tuples, if the *prefix*, *anchor_text* and *suffix* fields of the pattern are identical to the respective fields of a specific tuple of the set.

The following definition clarifies the relation of a tuple to a community.

DEFINITION 4.6. A tuple belongs to a specific community, or is a tuple of a specific community, if the *link_url* field of the tuple points to a page of the specific community.

4.2 Outline of the method A basic characteristic of our algorithm is that it consists of an iterative procedure. On each iteration, the following specific steps are executed:

1. Given a set of patterns (that we assume to have been extracted from tuples of our community), we create the set of tuples that match at least one of these patterns. The tuples examined, in order to determine if they match or not, are all tuples of our database (potentially the database contains the tuples of all links of the Web).
2. From the set of matching tuples we extract a set of patterns, according to a specific algorithm (explained in the next section).
3. Given the new set of patterns, we proceed to step (1).

Note that each iteration requires one sequential scan of the database. Therefore, the total procedure requires as many passes over the database as the number of iterations (in our experiments 3-10).

4.3 Description of the Algorithm We now proceed in presenting the algorithm that implements the pattern extraction procedure.

Given a set of tuples S_T , we aim at extracting a specific number of patterns. We assume here (and also in the experiments) that we want to extract three patterns for each field.

The outline of the algorithm is as follows:

1. We group the tuples of S_T according to how many common words they have in a specific field (*prefix*, *anchor_text* or *suffix*).
2. For each tuple we assign a number that we call *score*. The score equals to the sum of the maximum common words that the tuple has with other tuples in the three fields.
3. For each of the three fields, we select three tuples as patterns. The choice of the tuples is according to their score and group. The details of the above procedure can be seen in the pseudocode below.

Note that it is useful to think of a group as a collection of tuples that refer to a specific thing in a specific way.

The following algorithm implements the above and is given in three parts.

Classification of tuples (prefix field).

Input: A set of tuples, T .

Output: A classification of tuples in groups, where each group has the form $P_{n_i}^{W_i}$. W_i is a set of words: $\{w_1, w_2, \dots, w_{n_i}\}$ ¹. $P_{n_i}^{W_i}$ is the set of all tuples of our database that contain the n_i common words (in arranged order) W_i in the prefix field.

For each $t_1, t_2 \in T$

If t_1, t_2 have n common words in arranged order in their prefix field

Create a group P_n^W if this does not already exist where W is the set of their common words.

Add those two tuples t_1 and t_2 into that set if they are not already contained.

We further perform a classification of the given tuples (following a similar algorithm) in groups of the form $A_{n_i}^{W_i}$ and $S_{n_i}^{W_i}$ so that each set contains tuples with common words $W_i = \{w_1, w_2, \dots, w_{n_i}\}$ in their *anchor_text* and *suffix* respectively.

The following algorithm extracts the patterns that occur from the prefix field.

¹In our implementation, $n_i = 1, 2, 3$.

Pattern Extraction (prefix field).

Input: The set of groups \mathbf{P} that includes the classifications of tuples $\{P_{n_1}^{W_1}, P_{n_2}^{W_2}, \dots, P_{n_k}^{W_k}\}$ that we get as an output from the previous algorithm.

Output: Three sets of at most three patterns; that is, one set for each field: prefix, anchor_text and suffix.

$n \leftarrow 3$

While ($n \geq 1$) AND (number of assumed patterns < 3)

 While (it does not exist W such that $P_n^W \in \mathbf{P}$ for some W) $n \leftarrow n - 1$

 Among all $P_n^{W_1}, P_n^{W_2}, \dots, P_n^{W_k} \in \mathbf{P}$ find W_m for which $|P_n^{W_m}|$ maximum

 For each tuple t in $P_n^{W_m}$

 Find the largest n_1, n_2, n_3 such that

$t \in P_{n_1}^{W_1}$ and $t \in P_{n_2}^{W_2}$ and $t \in P_{n_3}^{W_3}$

 Set the score of the tuple $t_s = n_1 + n_2 + n_3$

 Create a pattern p whose fields are copied from the tuple $t \in P_n^{W_m}$ with the highest score t_s

 Mark $P_n^{W_m}$ in \mathbf{P} so as not to consider it again

The algorithms that do the pattern extraction for the other two fields, anchor_text and suffix, are similar to the above.

Pattern Match.

Input: The previously extracted patterns P and the set of tuples of our database T .

Output: A set of tuples T_M that match the given patterns.

For each $t \in T$

 For each $p \in P$

 If t matches (according to Definition 4.4) p

 Insert t in T_M

The above algorithms implement fully an iteration that includes: classification of tuples, pattern extraction and pattern matching. The tuples T_M are considered tuples belonging to the community we seek, and are used as input in the classification algorithm of the next iteration.

We consider tuples of our community the union of all T_M that are generated in each iteration. The algorithm ends after a number of iterations which is determined by the total number of community's tuples we aim to find.

4.4 Initial Patterns In order for the iterative procedure to begin, we need an initial set of patterns that correspond to the target community. One could think of many ways of achieving this. For example, if we have a Web page of a specific community we could get the set of

tuples corresponding to the backlinks of this page from our database. The application of the pattern extraction algorithm on this set of tuples outputs the initial set of patterns. In this way, the method can be used to find similar pages of the same community of a given Web page.

5 Experiments

For our experiments we needed a sample of the Web. In order to build a sample database, we developed a Web crawler. Our database consisted of approximately forty million of tuples. It is worth noticing, that for each Web page we do not need to store its entire content but only the links along with the text near them.

We tested the algorithm searching for many different communities. When the community we were looking for was very extended the results were very satisfactory. Given a sample as described above, we were able to find a lot of pages belonging to the same community. It is very interesting that many of these pages had not profound keywords similarities - that is, it would be very hard to be found by any typical keyword-based search algorithm.

For instance, in a specific example, while trying to find pages that dealt with movies of Steven Spielberg, the results that came through were very satisfactory. More specifically, using as initial tuples twenty backlinks of a Web page dedicated to "Jurassic Park" and following four iterations of the described algorithm, we resulted in about 120 Web pages. About 80% of them contained information on Spielberg's movies ("E.T.", "Indiana Jones" etc.) or contained various information on Spielberg. The remaining tuples did not belong to the specific community we were seeking. However, the quality of the results reduced when applied to communities that were either very small or were overlapped by other, larger communities.

We now present an experiment which demonstrates the above very clearly. In this specific example, we used tuples referring to *Michael Jordan* and *Chicago Bulls* Web pages trying to find pages that dealt with *NBA*. We performed six iterations, starting from a set of 3 patterns. In the sixth iteration, we found that approximately half of the extracted patterns were referring to the desired community, in a totally different way from the patterns of the first iteration. To our surprise, the other half of the patterns was referring to a different (but rather specific) community. Actually, the other community had to do with applied mathematics. Furthermore, we discovered that the reason for this was the existence of a researcher named Michael Jordan who had worked on this area.

Indicatively, at the fourth iteration we had six

tuple 1	page_url prefix anchor_text suffix link_url	members.aol.com/DDPMJR/LadyasBullsPage.index.html Chicago Bulls NBA com 1998 NBA Mikes Chicago Bulls Homepage www.nba.com/Finals/triviaques.asp
tuple 2	page_url prefix anchor_text suffix link_url	sportznutz.com/nba Atlanta Hawks Charlotte Hornets Chicago Bulls Cleveland Cavaliers sportznutz.com/nba/chi
tuple 3	page_url prefix anchor_text suffix link_url	www.showmetickets.com/nba Charlotte Hornets Chicago Bulls Detroit Indiana Pacers Milwaukee Bucks www.showmetickets.com/nba/cavaliers_tickets.htm
tuple 4	page_url prefix anchor_text suffix link_url	www.geocities.com/Colosseum/Field/2302/links.html Page Tribute Michael Jordan Nikki Michael Jordan Homepage Rare Air Page Ryan www.geocities.com/SiliconValley/5834/jordan.html
tuple 5	page_url prefix anchor_text suffix link_url	www.flagline.com/nba.htm Boston Celtics Charlotte Hornets Chicago Bulls Cleveland Cavaliers Dalas Mavericks www.flagline.com/nba-all.htm
tuple 6	page_url prefix anchor_text suffix link_url	www.ai.mit.edu/projects/cbcl/publications/theory-learning.html Michael Jordan Mean Field Theory sigmoid Belief Networks CBL Paper 135AI Memo publications.ai.mit.edu/ai-publications/1500-1999/AIM-1570.ps

Table 1: The respective tuples of the resulted patterns

patterns. Table 1 presents the six respective tuples from which the patterns were extracted (in the way that the extracted patterns had the same content on the fields {prefix, anchor_text, suffix} as the presented tuples). Note that by definition, the tuples can not contain any stop words.

6 Conclusions

The issue of extracting cyber communities from the Web is very complex. In this paper, we presented a new approach to this significant problem. We expect that this new direction will be followed upon and even better results will come up with future research.

Using the algorithm presented in this paper, some results that would not be attained by other techniques (such as keyword search or link structure analysis) arose. Moreover, given the efficiency of the algorithm and the relatively small size of the database needed, we believe that this method can be implemented in realistic applications.

Acknowledgments

We would like to thank Foto Afrati for insightful discussions on this subject. Also, Dunja Mladenic for many useful comments and the anonymous reviewer of a pre-views version of this paper for helpful remarks.

References

- [1] David Gibson, Jon M. Kleinberg, and Prabhakar Raghavan. “*Inferring Web communities from link topology*”, In 9th ACM Conference on Hypertext and Hypermedia, 1998.
- [2] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. “*Trawling the Web for emerging cyber communities*”, WWW8 / Computer Networks, 31(1116):1481–1493, 1999.
- [3] Jon M. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew S. Tomkins. “*The Web as a graph: Measurements, models, and methods*”. In T. Asano, H. Imai, D. T. Lee, S. Nakano, and T. Tokuyama, editors, Proc. 5th Annual Int. Conf. Computing and Combinatorics, [COCOON], number 1627. Springer Verlag, 1999.
- [4] S. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew Tomkins, and Eli Upfa, “*The Web as a graph*”, in Symposium on Principles of Database Systems, pages 1–10, 2000.
- [5] Gary William Flake, Steve Lawrence, C. Lee Giles, and Frans Coetzee, “*Selforganization and identification of Web communities*”, in IEEE Computer, 35(3):66–71, 2002.
- [6] Gary Flake, Steve Lawrence, and C. Lee Giles, “*Efficient identification of Web communities*”, in Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 150–160, Boston, MA, August 20–23 2000.
- [7] Masashi Toyoda and Masaru Kitsuregawa, “*Creating a Web community chart for navigating related communities*”, in Conference Proceedings of Hypertext 2001, pages 103–112, 2001.
- [8] Masashi Toyoda and Masaru Kitsuregawa, “*A Web community chart for navigating related communities*”, in Poster Proceedings of 10th International WWW Conference, pages 62–63, 2000.
- [9] Masashi Toyoda and Masaru Kitsuregawa, “*Finding related communities in the Web*”, in Poster Proceedings of 9th International WWW Conference, pages 70–71, 1999.
- [10] Sergey Brin, “*Extracting patterns and relations from the world wide Web*”, in PWebDB Workshop at 6th International Conference on Extending Database Technology, EDBT’98, 1998.