A Weblog Recommender using Machine Learning and Semantic Web Technologies

Marc Crean

Supervisor: Peter Flach
External Supervisor: Simon Price (ILRT)

16th May 2003

Abstract

The semantic web is an extension of the existing web that adds a layer of information semantically describing the data on the web. It is an emergent technology that has great potential. In this paper we give a discussion of the background to the semantic web followed by an introduction to one of the areas where it is starting to be introduced – weblogging. A Weblog or blog is a website that aggregates data from other, usually similar, websites. This paper goes on to propose a system that will make use of machine learning technologies to classify and recommend weblogs to users.

CONTENTS

1. INTRODUCTION 4

1.1 THE SEMANTIC WEB 4

1.1.1 Resource Description Framework 4

1.1.2 RDF Vocabularies 6

1.2 WEBLOGS AND NEWS FEEDS 7

1.3 INTRODUCTION TO THE PROJECT 9

2. BACKGROUND 9

2.1 SEARCHING THE WEB 9

2.2 CONTENT-BASED SEARCHING 10

2.3 LINK STRUCTURE ANALYSIS 10

2.1.1 Kleinberg's algorithm. 11

2.4 COMBING LINK-STRUCTURE ANALYSIS WITH CONTENT-BASED APPROACHES 14

2.5 RELATIONAL DATA MINING AND INDUCTIVE LOGIC PROGRAMMING 14

2.6 OTHER TECHNIQUES 15

3. IMPLICATIONS FOR THE PROJECT 17

4. AIMS AND OBJECTIVES 18

4.1 STATEMENT OF MAJOR AIMS 18

4.2 LIST OF OBJECTIVES 18

4.3 DESIGN DECISIONS 19

4.4 PROJECT IMPLEMENTATION PLAN 20

REFERENCES 21

1. Introduction

"*The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.*"
Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

1.1 The Semantic Web

As it is today the World Wide Web (WWW) provides a mechanism that allows humans to communicate information to one another. Essentially the only data that is processed by machine is that which is used for formatting and displaying data. The major aim of the semantic web is to represent data in such a way that semantic information may be used not only by humans but by computers as well. This section gives a brief overview of some of the technologies that are driving the semantic web. In section 1.1.1 we will describe the Resource Description Framework (RDF), which is perhaps the single-most important element of the area. In section 1.1.2 we will look at RDF vocabularies. Finally in section 1.2 we will look at an area where some semantic web technologies are being applied.

1.1.1 Resource Description Framework

The Resource Description Framework (RDF) is the language around which the semantic web is being built. It is being developed by the w3c [1]. RDF is specifically aimed at expressing metadata about web resources [2], but its power comes from its extensibility. Essentially the language allows the author to describe any resource that can be uniquely identified by means of a URI (uniform resource identifier).

Description of resources is achieved using triples to represent *subject, property,* and *object*. To illustrate this concept consider the sentence[1] "http://sis.bris.ac.uk/~mc9854/index.html was created by Marc Crean ". In RDF this could be represented by the graph in figure 1.1.

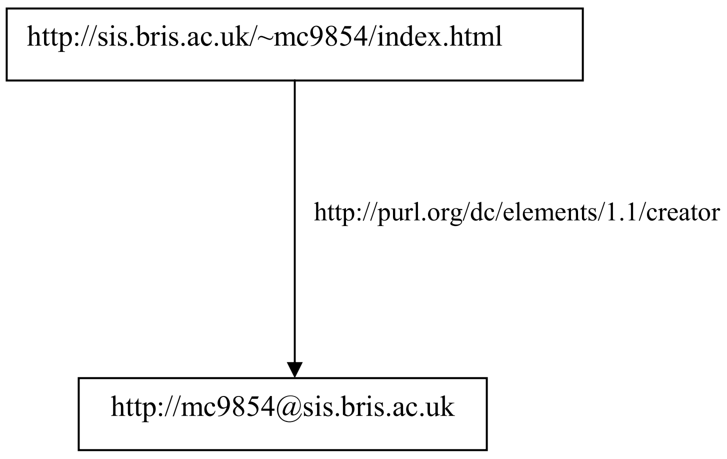

http://sis.bris.ac.uk/~mc9854/index.html
http://purl.org/dc/elements/1.1/creator
http://mc9854@sis.bris.ac.uk

Figure 1.1 A simple RDF graph

[1] This example is adapted from [2]

Essentially the graph in figure 1.1 is the following triple:

Subject: http://sis.bris.ac.uk/~mc9854/index.html

Property: http://purl.org/dc/elements/1.1/creator

Object: http://mc9854@bris.ac.uk

There are several things to notice here. First is that each element of the triple is identified by a URI. This is the main mechanism in RDF for identifying resources and is one of the most powerful aspects of RDF. A URI can be used to refer to anything; they are not just limited to things that can be retrieved via the Internet. People, real-world objects and concepts can all be identified by a URI and anyone has the freedom to create a new one[2]. In the above example the subject, property and the object are all identified by a URL, which is a type of URI which can be retrieved. It should be noted however that objects might also be represented as literals [3]. So in the above example the object might simply have been "Marc Crean". However this could lead to confusion as there would be no way of distinguishing between two objects of the same name. In actual fact RDF uses URI references to be more precise. A URI reference is simply a URI with a fragment identifier coupled onto the end. For example the URI reference http://example.com/#fragID is made up of the URI http://example.com/ and the fragment identifier "#fragID". In fact in RDF a resource is defined as anything that can be identified by a URI reference [3]. You will also notice in the example above that the property "creator" is not identified by a literal but by a URI - this has two advantages.

Firstly it means that we can distinguish between two different properties used by different people that would have had the same name. For example one person might want the property "address" to contain just a "street name" and "number", whilst another person might want it to include "town name". If they have different URIs then these two different definitions of address can be differentiated from one another.

The second major advantage is that because the property is identified by a URI reference it is a resource and because it is a resource we can now build more information around it. In fact it can now become the subject of a new relationship. For example[3] in the previous example the property "creator" could become the subject of a new RDF statement describing what is meant by "creator" [3]. Figure 1.2 shows how this might be achieved.

By identifying thing as resources, we can continue to make statements about the different resources that we have. In this respect it is hoped that the semantic web will grow into a large network of machine understandable data.

To make RDF machine understandable a language known as XML/RDF has been developed the details of this language can be found in [1]. They have been omitted here so in order that we can focus on the concepts rather than the specifics of RDF.

---

[2] Provided that the URI does not already exist
[3] This example is adapted from [3]

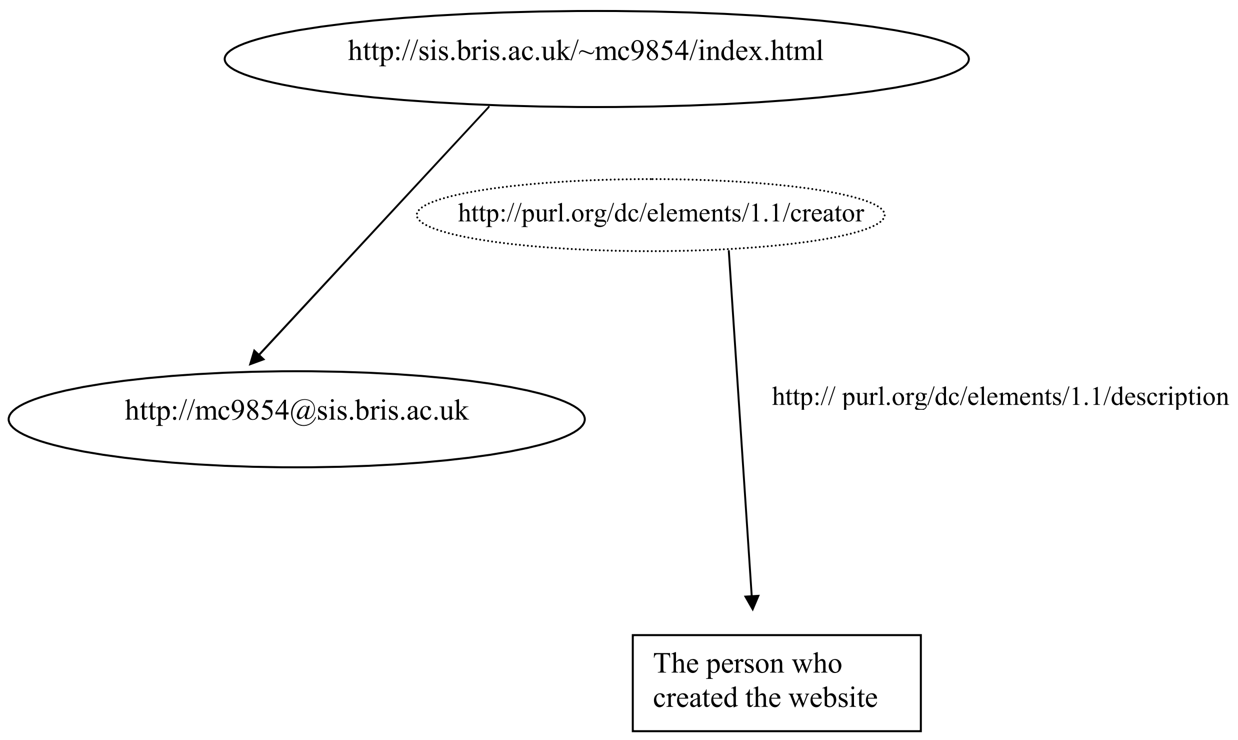

http://sis.bris.ac.uk/~mc9854/index.html
http://purl.org/dc/elements/1.1/creator
http://mc9854@sis.bris.ac.uk
http:// purl.org/dc/elements/1.1/description
The person who created the website

Figure 1.2 An RDF Graph with a property as a new subject

It should be noted that this is just a brief introduction to the RDF and the semantic web and many elements have been left out. Some of the more sophisticated elements will not be relevant to this project and have been omitted for this reason. Two such features are RDF typing and RDF classes. For more information on these features see [4] and for a general background see [1].

1.1.2 RDF Vocabularies

We have seen in the last section how RDF can be used to make statements about resources. How though can we provide the necessary tools to enable users to create their own statements about their own resources? One of the main aims of the semantic web is to keep it decentralised, much like the regular web. In this section we will briefly describe how RDF can be used to create vocabularies. Due to limitations on space we will omit the technical and syntactical details and instead concentrate on the principal.

RDF Schema [5] is a language that has be specified to allow users to create their own RDF vocabularies. It is a subset of RDF. Essentially, RDF Schema provides users with a language with which they can create their own RDF properties. To illustrate this we will now briefly describe an example of an existing vocabulary that exists in RDF.

The Dublin Core Metadata Initiative (DCMI) [6] is a project concerned with producing metadata standards. One of the major aims is to promote the development of vocabularies of metadata that can be used do describe different resources to help facilitate resource retrieval. Some of the metadata sets can be used within RDF. Figure 1.3 shows how some of the elements of the Dublin Core Metadata Element Set can be used, as RDF, to describe a resource.

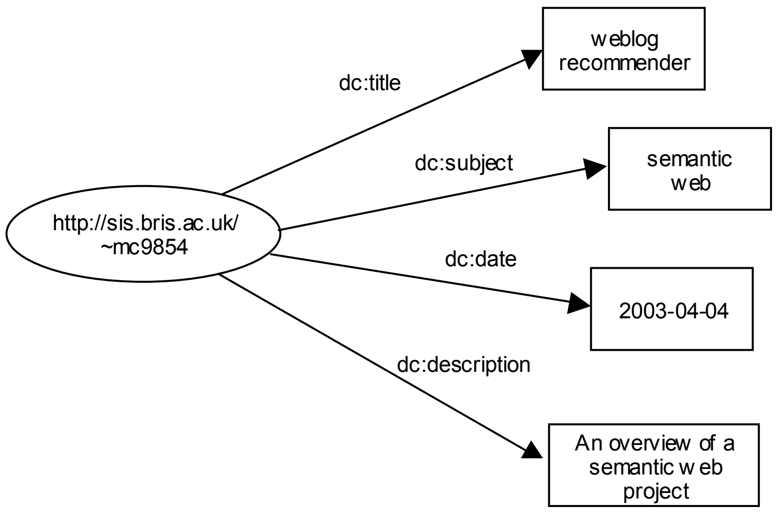


Figure 1.3 An RDF graph using some Dublin Core Elements.

This diagram was adapted from [7]

RDF itself does not provide any vocabulary for describing resources. It provides the framework from which anyone can create his or her own vocabularies [8]. Although this provides a certain freedom to develop unique personal vocabularies for every occasion, it is hoped that people will reuse each other's vocabularies when possible. It is important to remember that RDF itself is completely unspecific in terms of the domains to which it can be applied and this is one of its major strengths [8]. People can create or extend a vocabulary to describe their particular resources but they do not need to create new languages to express these vocabularies nor do they need to develop new tools to deal with them.

As we have said, for brevity we will avoid specific implementation details of the RDF schema language (RDFS) and instead refer you to the literature [5].

1.2 Weblogs and News Feeds

Over the past 3 or four years the Internet has seen a rapid growth of site syndication technologies. Site syndication basically refers to the process of sharing information between websites. There are two main areas where this has become common; weblogging and news feeds.

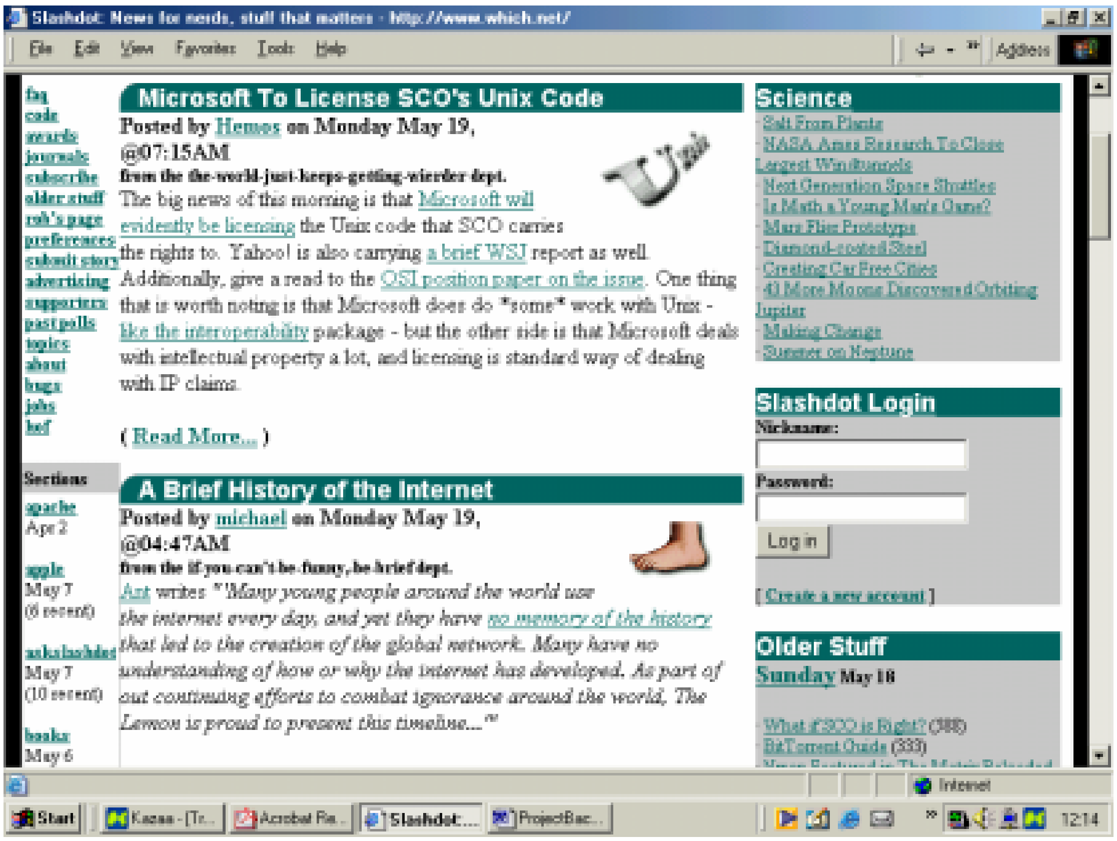
Both weblogs (or blogs) and news feeds work using the same basic principle. A website, or part of a website such as a news story, is summarised and described using a language known as RSS 1.0 (RDF Site Summary)[4]. This summary (or feed) can then be registered at an aggregation site and used by others who wish to include it in their website.

One of the most popular uses of this technology is in circulating news headlines and stories. A site that wishes to include all the main headlines from a variety of sources may simply reference the relevant feeds and display the relevant information from these feeds. When the owner updates the feed, all the people who receive that feed will not need to update their sites individually as their site will now be accessing the updated feed. This automation allows web authors to circulate all kinds of information about their sites with ease.

The other popular application of RSS is in weblogs. Weblogs are difficult to define exactly. This is due, in part, to their widespread application. One good explanation comes from [8] which describes a weblog as:

"...a weblog is a text publication that provides links and commentary around a specific topic".

Weblogs often link to other weblogs and in this way whole communities of interlinking blogs often emerge around a specific topic. Figure 1.4 shows a screenshot of a weblog.



Microsoft To License SCO's Unix Code

Posted by Hemos **on Monday May 19, @07:15AM**
from the the-world-just-keeps-getting-wierder dept.

The big news of this morning is that Microsoft will evidently be licensing the Unix code that SCO carries the rights to. Yahoo! is also carrying a brief WSJ report as well. Additionally, give a read to the OSI position paper on the issue. One thing that is worth noting is that Microsoft does do *some* work with Unix - like the interoperability package - but the other side is that Microsoft deals with intellectual property a lot, and licensing is standard way of dealing with IP claims.

(Read More...)

A Brief History of the Internet

Posted by michael **on Monday May 19, @04:47AM**
from the if-you-can't-be-funny,-be-brief dept.

Ant writes "*Many young people around the world use the internet every day, and yet they have no memory of the history that led to the creation of the global network. Many have no understanding of how or why the internet has developed. As part of out continuing efforts to combat ignorance around the world, The Lemon is proud to present this timeline...*"

Science
- Salt From Plants
- NASA Ames Research To Close Largest Windtunnels
- Next Generation Space Shuttles
- Is Math a Young Man's Game?
- Mars Flier Prototype
- Diamond-coated Steel
- Creating Car Free Cities
- 41 More Moons Discovered Orbiting Jupiter
- Making Change
- Summer on Neptune

Slashdot Login
Nickname:
Password:
Log in
[Create a new account]

Older Stuff
Sunday **May 18**
- What if SCO is Right? (388)
- BitTorrent Guide (333)

Figure 1.4 a screenshot of a weblog taken from slahdot.com

Recently, some weblogs have started to use RSS to summarise their site. The RSS feed can subsequently be used by other weblogs and in this way the content of each blog becomes more dynamic. As well as this added dynamism of course, is the added information that is

[4] In fact there are several different versions of RSS. We are concerned here only with the RDF based RSS 1.0.

provided by the RSS feed which could be used by machines filtering through various weblogs.

We have seen in this section a brief introduction to the semantic web as a whole. We have also seen how some of the emergent technologies are being employed for syndication purposes. In the next section we present the major aims and objectives of the project with regard to the context that has been introduced here.

1.3 Introduction to the project.

The central aim of this project is to produce a searching application for weblogs. The application should be capable of receiving the URL of an existing weblog and returning a list of weblogs that are similar. The idea is to produce a search engine that operates effectively by exploiting both the metadata and the structure provided by RDF.

Search engines for weblogs do already exist (for example see www.blogstreet.com) however the way in which they work is not documented. Also the existing search engines tend to exist on weblog archive sites and do not exist as stand alone systems. This project intends to produce a search engine that (once trained at least) will work independently of other weblogs. We have chosen to concentrate on weblogs rather than news feeds as weblogs currently make better use of RSS technology.

In order to achieve this aim the project will require some kind of searching and classification technique. There are many existing technologies, with differing origins, that could be adapted to this task. The main areas that have been considered are search engine technology and

machine learning techniques. One of the desirable features of the system is that it makes use of the structure and additional metadata provided by the RSS and RDF representation. For this reason some of the more old fashioned machine learning techniques were not considered. The following chapter provides a detailed account of the most compelling approaches.

2. Background

To recommend any document requires a number of things:

- An archive of documents to search through
- Some kind of strategy for searching
- Some criteria for comparing and selecting (and perhaps ranking) documents

In this chapter we will present some of the previous work in the area of information retrieval and Internet searching. We will look at some of the simplest techniques and go on to discuss some of the more advanced. First though we will give an overview of the general search process in the context of the Internet. At the end of the chapter we introduce Relational Data mining as an alternative to the search engine based approach.

2.1 Searching the Web

There are generally two types of Internet search engine [9] - those that use an automatic computer agent to crawl the web and those that use humans to direct the search. We will restrict our focus to the former.

A web crawler, or spider, is an agent that follows http links from site to site and accumulates an index of web pages. This index contains a copy of every web page that the crawler finds. When presented with a query the search engine will look through the index of sites and try to select and rank those sites that best match the query. There are many different strategies for this searching and ranking, some of which will be shown in the following sections.

2.2 Content-based searching

A great deal of existing search engines use keyword searches to rank pages. As most people will know this often leads to irrelevant and spurious results as well as lists of pages that can sometimes number in the millions.

In its most simple form, a content-based search engine will count the number of the query words (keywords) that occur in each of the pages that are contained in its index. The search engine will then rank the pages with the best being the page with the highest number of occurrences as a fraction of the size of the page.

More sophisticated approaches take into account the location of the keywords. For example keywords occurring in the title tags of the web page are more important than those in the body of the text and keywords occurring at the start of the body of the text are more important than those towards the end. Most modern search engines employ this kind of strategy.

Other types of search engine, like Yahoo™, use topic hierarchies to help to narrow the search and make search results more relevant. Theses topic hierarchies are human created. Because of this, they are costly (in terms of time) to produce and maintain and are subsequently not updated as often as the fully automated systems. Recently however there has been some work into the automatic construction of these topic maps [10].

2.3 Link Structure Analysis

Link structure analysis has emerged in recent years as an alternative to content-based search-engines. One of the major aims of these link-based approaches is to combat some of the problems that these content-based systems face. These problems are summarised succinctly in [11] and include:

- Synonymy – If the users input is "automobile" say, then ideally we would want documents also containing the term "car".
- Polysemy/Ambiguity – If the input is "Blair" do we mean Tony Blair or Lionel Blair.
- Authorship styles – Two web pages about the same topic often contain different vocabularies due to the different authors. This problem is essentially an extension of the synonymy problem.
- Spamming – As web authors become more aware of search engine strategies they are more equipped to design pages that will be retrieved more often, even when the query is for something unrelated to the site.

Link-based web searching was introduced by Kleinberg [12]. He suggested that if there is a link from website A to website B then website A recommends in some way the content of website B. Furthermore it is often the case that a link from A to B indicates some kind of common topic shared between the two sites, although this is not always the case. (It is fair to say that topical relationship between connected sites is at least as likely in a weblogging context as it is in the web at large.)

Kleinberg went on to identify two types of web sites, hubs and authorities. As Kleinberg puts it in [13]:

"Some pages, the most prominent sources of primary content, are the *authorities* on the topic; other pages, equally intrinsic to the structure, assemble high-quality guides and resource lists that act as focused *hubs,* directing users to recommended authorities"

What is meant by prominence here is that the site is linked to frequently. Given that a link from A to B indicates some kind of recommendation of B from A, then a site that is densely linked to is recommended by many and can hence be considered an authority. This concept is illustrated in figure 2.1.

Node A would be considered an authority and the other nodes considered hubs. As you can see in the diagram many hubs link to an authority but have few incoming links themselves. It is this notion that is central to Kleinberg's approach.

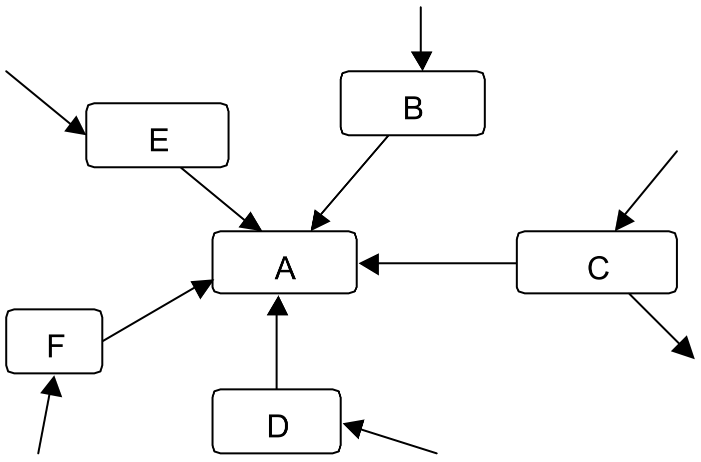


Figure 2.1 A graph illustrating the hub and authority concepts. Here the nodes in the graph represent websites and the arcs represent links between the websites

2.1.1 Kleinberg's algorithm.

Originally the algorithm developed by Kleinberg took a broad topic as a query string. The term "broad topic" is not exactly defined but it can be considered a topic where there is an abundance of documents that contain the query string and the problem is deciding which documents actually relate to the topic The algorithm developed by Kleinberg can be broken down into two stages:

1. Create a subgraph of the web to use as the search space

2. Compute the hub and authorities in the subgraph

We will now examine these two stages in more detail

Creating the Original Subgraph

The overall aim of the algorithm is to find authoritative pages on a given (broad-topic) query. It is obviously not feasible to try and attempt to apply this algorithm to the web as a whole. One possibility is to apply the algorithm to a subgraph of the web where pages in the graph contain the query string. However, as Kleinberg points out this could result in a subgraph of a million or more nodes. Instead the original subgraph, S_σ, is created by algorithm[5] 2.1

```
Subgraph(σ,ε,t,d)

σ : a query string.
ε: a text-based search engine.
t, d: natural numbers.
Let Rσ denote the top t results of σ on ε.
Set Sσ := Rσ
For each page p Є Rσ
Let Γ+ (p) denote the set of all pages p points to.
Let Γ- (p) denote the set of all pages pointing to p.
Add all pages in Γ+(p) to Sσ.
If |Γ- (p)| <= d then
  Add all pages in Γ- (p) to Sσ.
Else
  Add an arbitrary set of d pages from Γ- (p) to
Sσ.
End
Return Sσ
```

Algorithm 1 - Kleinbergs Subgraph Algorithm

The first part of the algorithm takes the query string and inputs it to some text-based search engine. The first t results are then considered for expansion. The reason why this set alone cannot be used is that it will not contain enough authority sites to be useful. Kleinberg observes that although an authority is unlikely to be in this set it is likely to be pointed to by at least one member of the set (R_σ). So the set is grown such that each page in R_σ brings with it any page that points to a page in R_σ and any page that is pointed to by a page in R_σ up to a maximum of d pages.

Compute the Hub and Authorities in the Subgraph

Now that there is a small enough subgraph to work on the hubs and authorities can be located. One simple way of doing this would be to consider the best authorities to be those pages in S_σ with the largest number of incoming links. However this is not sufficient. Some pages that remain in S_σ will have a large number of incoming links but will not be related to the query topic. Popular sites (such as Yahoo) will always have a very high number of incoming links. What separates these sites from good authority sites is illustrated in figure 2.2[6].

[5] This algorithm is taken from [12] pp 6-7
[6] This figure has been slightly adapted from [12] pp 8.

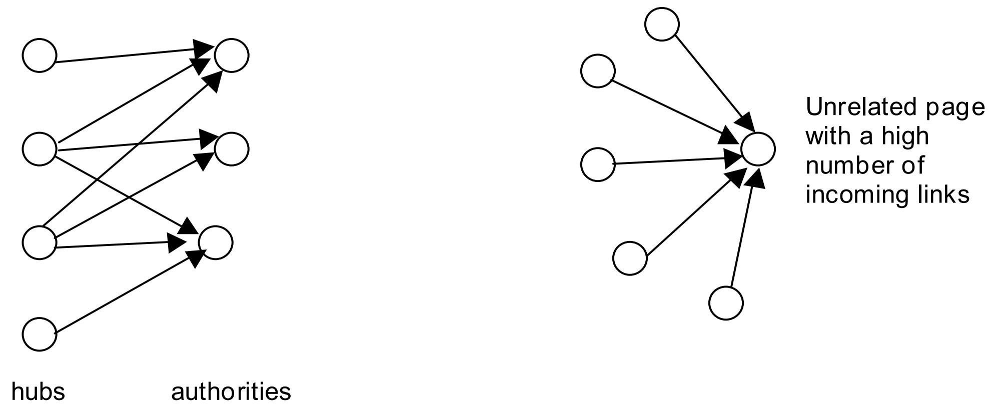


Figure 2.2 The difference between good authorities and popular unrelated pages

The idea is that good hub pages will point to many good authorities and good authority pages will be pointed to by many good hubs. The algorithm makes use of this relationship by assigning to each page, p, an authority score x(p) and a hub score y(p). The sum of the square of all authority scores is normalized to 1. The sum of the square of the hub scores is normalized to 1. Larger scores mean better authorities or better hubs (for x(p) and y(p) respectively).Algorithm 2.2[7] below shows how the algorithm iteratively updates the x(p) and y(p) scores of each page in the subgraph.

```
Let S be a set of n pages
Let a be a vector containing an authority score for each page p in S
Let h be a vector containing a hub score for each page p in $S_\sigma$.
$a_i$ = 1 for i = 1...n
$h_i$ = 1 for i = 1 … n;

do
   for i = 1 to n
      $a_i = \Sigma_{\{x \mid x \text{ points to } Si\}} h_x$
    for I = 1 to n
      $h_i = \Sigma_{\{x \mid Si \text{ points to } x\}} a_x$
   normalize the scores in a and h
until convergence
```

Algorithm 2.2

In fact the algorithm does not need to be applied until convergence and Kleinberg provides a method for choosing a suitable number of iterations. The algorithm has been successfully applied and some of the results are particularly impressive (see [12]). Furthermore the paper points out that this technique can be adapted to a similarity measure. That is, instead of receiving a query string as input, the URL of an existing website is provided and the algorithm will find similar pages. In the context of this project that is an invaluable attribute. The adaptation is so trivial that both a query based system and a similarity based system would be possible using this algorithm or variations thereof.

Other systems have adapted and expanded on Kleinberg's system, most notably [11] which uses a stochastic approach and [14] which we will see in the next section.

[7] This algorithm has been adapted from [11]. The full and original version can be found in [12] pp9

2.4 Combing Link-Structure Analysis with Content-based Approaches

In the previous two sections we have seen two approaches to document retrieval. In this section we will present an overview of an approach [14] that combines the two approaches. It is this approach that will form the basis for a large part of this project.

The method builds upon the algorithm of the previous section by incorporating a certain amount of textual information into the process. What motivates this approach is the observation that the text immediately surrounding a link to a page usually bears some kind of reference to the context of the page that the link points to.

So instead of the scores being dependent solely on the link structure surrounding a page, they are now weighted according to certain key-word matching procedures.

For each link from page A to page B we want to increase the weight, w(A,B), proportional to the amount of topic-related text surrounding the link in A. One of the problems of this is deciding how big an area should be considered. This problem was overcome by using certain experimental techniques to determine the optimal size of area to use (for details see [14]). In weblogs however this is not a problem.

Due to the structure of RSS we do not need to consider text embedded in HTML. We need only consider the metadata contained in the RSS file that describes the site. Links to other weblogs will be surrounded by meta-descriptions of those sites. In this way the weighting scheme should become even more accurate.

2.5 Relational Data Mining and Inductive Logic Programming

The field of relational data mining has seen a steady growth in recent years. In normal data mining applications the agent(s) and human are looking for rules and patterns in a single table database. In relational mining, however, the agents must search for data in multi-table relational databases [15].

The task in relational mining is: given a relational database D, search for patterns and rules that are valid for D. This is much the same as in regular data mining. What is different is the language used to describe the data, which, in relational data mining, is typically first-order logic [15].

Inductive logic programming (ILP) can be seen as being concerned with "… the development of techniques and tools for relational data mining"[8]. The goal of ILP is to induce a set of rules, or a hypothesis, about instances contained in a relational database.

Relational databases capture more semantically rich information than their single table counterparts and hence ILP stands to produce more semantically rich hypotheses. An area where this has recently been applied is the field of web-page classification [16 ,17].

Due to limitations on space we cannot present a detailed account of these studies but will instead give a brief description of the central concepts

The idea is to represent the web as a relational database. This could be achieved using a representation something like that shown in figure 2.3

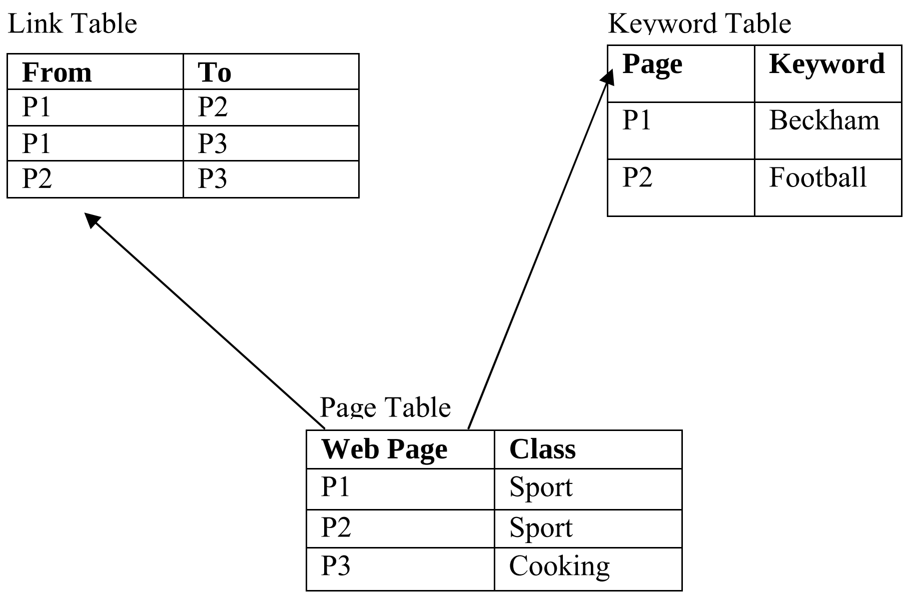
---

[8] From [15] pp 48.

We then provide the learning ILP algorithm with this data and some background knowledge describing the relationships[9]:

has_link(P1, P2) : true if there is a link from P1 to P2
has_word(P1, Word): true if page P1 contains Word.

The idea is to induce rules in terms of the background knowledge, which best classify instance of web pages in the database. We can then use these rules to classify future instances. Of course this approach does depend on having a training set containing classified web pages, something that is not always possible.



Link Table

From	To
P1	P2
P1	P3
P2	P3

Keyword Table

Page	Keyword
P1	Beckham
P2	Football

Page Table

Web Page	Class
P1	Sport
P2	Sport
P3	Cooking

Figure 2.3 A relational database representing web pages

In [17] a method is presented that combines relational techniques with more conventional statistical techniques for text classification with some success. Given that in a weblog the text contained in the RSS file is far more concentrated and relevant than it would be in a normal web page this technique may be ideally suited to this project.

2.6 Other Techniques

So far we have presented some of the literature in areas related to the project as a whole. It should be noted that the literature included in the previous sections was by no means the only material considered. It serves as a background to the approach that will be taken to the major parts of the project. Other ideas considered included XML document clustering and graph clustering which we will briefly mention here.

Yoon, Raghavan and Chakilam [18] describe a system for clustering XML document using a bitmap indexing technique. The technique exploits XML's hierarchical structure as well as making some use of the text contained between tags. They present results that show their technique to be a useful means of grouping XML documents. Due to the fact that RDF and RSS are subsets of XML this technique was considered. However, although the XML

[9] This representation is simplified and is provided for illustrative purposes only.

structure within an RSS file may be in some way hierarchical, the semantic information upon which we wish to base our classification is not encapsulated in a hierarchical way. As we have seen RDF has a graph structure and it this structure, if any, that we wish to exploit.

Graph comparison and clustering are quite well established techniques that share their origins between Mathematics and Computer science. They were considered here because of RDF's graph structure. However, due to the size of graphs that will be necessary and the size of the vocabulary (which translates into possible node values in terms of graph theory) we will be working with, this approach will not form any part of the project.

3. Implications for the Project

Many of the techniques of the previous chapter will be used as a basis for the project. In particular the hub/authority approach will be adapted for use with the semantic web. The assumptions that originally motivated the technique are intuitively valid for weblogs. Weblogs generally aggregate information from other sights that are similar to them. It stands to reason that, just as for normal websites, links between pages confer some kind of recommendation of the sights being linked to.

The hybrid system of [14] is particularly suited to the weblog environment. The reason for this is that in the original system it was assumed, quite reasonably, that the text surrounding hyperlinks in a web page is related to the page being pointed to. In the weblog domain it is not necessary to include the text surrounding the hyperlink on the HTML page, instead we can consider the metadata that surrounds a link in an RSS file. It is at least as strong an assumption that the value of metadata that describes a link/page will be related to the content or topic of the page.

A second approach that will be investigated further is relational data mining. With regards to the weblog domain this will entail representing a community of weblogs as a relational database. This could be much like the one shown in figure 2.3 with the exception that now we have the RSS metadata to use as well. So a table or tables representing the metadata in the RSS file would now replace the keyword table in figure 2.3.

Most ILP algorithms would require some kind of categorization of weblogs and a training set of classified examples and this could present a problem as not all weblogs have a category. One possible solution is to use an archive of categorized weblogs (such as blogstreet .com) as a training set and then to perform testing on both a subset of this archive and on an uncategorized set of weblogs. In the second case the success of classification would have to be evaluated subjectively as there would be no hard definition of what category an unseen weblog should fall into.

In ILP the choice of background knowledge and the way it is represented greatly biases the search. If you consider the simple background predicates provided in the example from section 2.5 it is interesting to note that a search for rules describing weblogs / web pages can be equated with the link structure analysis approaches of section 2.3. Instead of providing a rigid algorithm for learning the hubs and authorities, in the ILP setting we are trying to induce them from examples. If we change the representation so that the background predicates do not include descriptions of links then the similarity between the methods disappears. An interesting area might be in comparing the relative successes of the ILP approach with different representations of the data.

4. Aims and Objectives

4.1 Statement of major aims

The first aim of this project is to design and implement a weblog recommender by adapting existing technologies to the semantic web. The application will use text analysis combined with link structure analysis to rank and recommend weblogs.

The second aim is to investigate and implement a second recommender system using techniques from relational data mining and ILP. This second system will build on some of the infrastructure put in place by the completion of the first aim. That is, common components will be reused. On completion of this aim an evaluation and comparison of the two approaches will be conducted.

Both the aims will involve producing a graphical interface for input and the delivery of results.

4.2 List of Objectives

The following is list of high-level objectives that will need to be achieved in order to complete the task.

Construct a Web Crawler
In order to compile training and test data, it will be necessary to implement an agent that will crawl the internet by following http links from weblog to weblog. The agent will build up an index of weblogs that can then be used to retrieve RSS and structural (ie link) data.

Compile a Dataset
Having retrieved a catalogue of weblogs, the relevant information from each weblog will be required. A program will be written that will visit the sites catalogued by the crawler and store the RSS and link structure information in a database. It should be noted that this objective may be combined with the previous objective. The reason they are separate here is that the tasks will require separate algorithms and are sufficiently large to be considered individually.

Write a Content Based Recommender
The preliminary recommendation system will be based solely on content. A simple keyword-matching algorithm will be used to rank candidate weblogs.

Write a Link-Structure Based Recommender
This part of the system will be based on the work of Kleinberg [12]. The system will take the URL of an existing weblog as a query and use the web crawler to compile the initial set of candidate weblogs. The graph of this set will then be analysed according to Kleinberg's algorithm and the recommendation will be based on the top N authority sites.

Combine a Link-Structure with Content
The work of [14] will be adapted for the semantic web. Specifically the RDF retrieved from RSS files corresponding to weblogs will be used instead of the text contained on the web page itself. The notion of vicinity used in [14] will also have to be revised. Instead of considering text within a certain distance of a hyperlink we will consider the value of the most important elements of the RSS file

Investigate the feasibility of Relational Data Mining.
The initial research conducted into this area will need to be extended. From the preliminary findings it seems that this approach is certainly suitable. Further investigation is required though. Having completed this feasibility study an implementation of the approach will be realised.

Create an Interface
In order to be re-useable by others the software will be wrapped in some kind of UI. This may take the form of a desktop application or it may be a web service.

4.3 Design Decisions

This section reviews some of the design decisions that have already been taken. Figure 3.2 below shows the overview of the pipeline from a user query to a list of results.

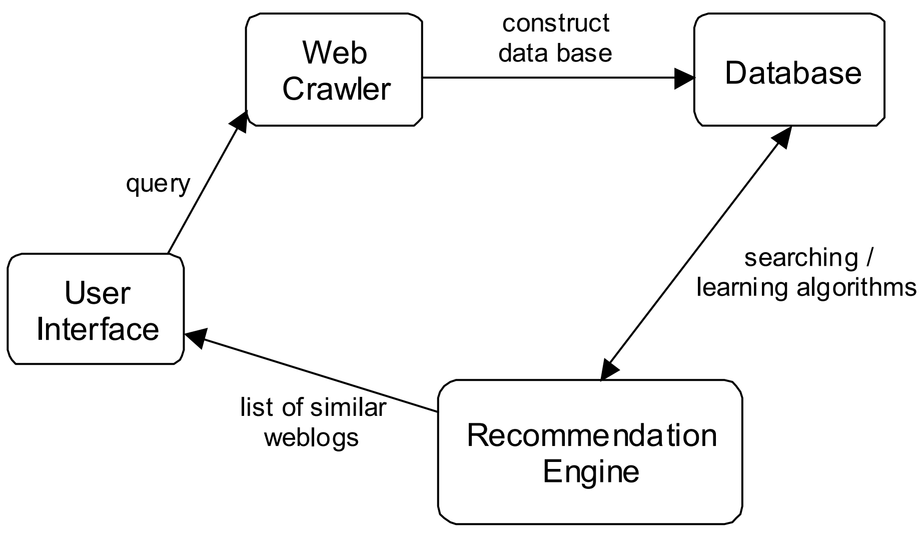


Figure 4.1 The components of the system and pipeline from query to recommendation

The Web Crawler
This will be written in Java using in built java classes to crawl the web.

The Recommendation Engine
The link-structure and content-based algorithms will be written in java. The reason for this is that there will be a considerable amount of interaction with the SQL database and the JDBC classes provided by java are well suited to this.

For the ILP algorithms there is the possibility of using existing packages (for example Tertius) or implementing an existing algorithm. In the latter case this would be achieved using Prolog given that it is well suited to the ILP domain.

The Database
The database will be an Oracle SQL database. The reason for this is that it is familiar to both the author and to the department.

The User Interface
Initially a simple java GUI will be implemented using the swing classes. This GUI will have minimal, but sufficient, functionality. Eventually it is hoped that the system will be turned into a full web service using SOAP, although this is not one of the primary objectives of the project and will be implemented only if time permits.

4.4 Project Implementation Plan

Task	Date
Web Crawling Software	June 10 – June 24
Database Construction Software	June 27 – July 6
Content Based Recommendation Software	July 7 – July 10
Link Structure Analysis Software	July 11 – July 25
Combining Link Structure with Content	July 26 – July 29
User Interface 1	July 29 – August 03
Relational Data mining Investigation / Implementation	August 04 – August 19
Evaluate the System	August 20 – August 24
User Interface 2	August 24 – August 28

It should be noted that the second User Interface task is not an essential component of the project. If it is deemed that extra time is required in one of the other areas then this part will be dropped. This is not to say that it is a meaningless part as turning the whole project into a finished useable application or web service would add completeness.

References

[1] Semantic Web Group at w3c:
www.w3c.org/2001/sw

[2] Resource Description Framework (RDF) Model and Syntax Specification
http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

[3] RDF Primer
http://www.w3.org/TR/rdf-primer/

[4] Resource Description Framework (RDF): Concepts and Abstract Syntax
http://www.w3.org/TR/rdf-concepts/

[5] RDF Vocabulary Description Language 1.0: RDF Schema
http://www.w3.org/TR/rdf-schema/

[6] Dublin Core
http://dublincore.org/

[7] Expressing Qualified Dublin Core in RDF / XML
http://dublincore.org/documents/2002/04/14/dcq-rdf-xml/

[8] JOHAN HJELM (2001), *Creating the Semantic Web with RDF,* John Wiley.

[9] DANNY SULLIVAN(2002), *How Search Engines Work*.
http://www.searchenginewatch.com/webmasters/article.php/2168031

[10] G. Attardi, A. Gull'i, & F. Sebastiani, "Automatic web page categorization by link and context analysis," in *THAI-99, 1st European Symposium on Telematics, Hypermedia and Artificial Intelligence*, C. Hutchison and G. Lanzarone, Eds.,1999, pp. p. 105–119.

[11] R. Lempel & S. Moran, *The Stochastic Approach for Link-Structure Analysis (SALSA) and the TKC Effect. Abridged Version.* Department of Computer Science,The Technion, Haifa 32000, Israel.

[12] Jon M. Kleinberg.(1997) *Authoritative Sources in a Hyperlinked Environment*. Journal of the ACM

[13] JON M. KLEINBERG (1999), *Hubs, Communities and Authorities*. Cornell University.

[14] S. Chakrabarti, B. Dom, D. Gibson, J. Keinberg, P. Raghavan, & S. Rajagopalan (1998) *Automatic Resource list Compilation by Analyzing Hyperlink Structure and Associated Text*. Proceedings of the 7th International World Wide Web Conference.

[15] SASO DZEROSKI , NADA LAVRAC (Eds), *Relational Data Mining* . Springer (1998).

[16] S. Slattery & M. Craven (1998) *Combining Statistical and Relational Methods for Learning in Hypertext Domains* . Proceedings of ILP-98. 8th International Conference on Inductive Logic Programming.

[17] M. Craven (1998) *First Order Learning for Web Mining*. European Conference on Machine Learning.

[18] J. Yoon, V. Raghavan, V. Chakilam (2001). *Bitmap Indexing-based Clustering and Retrieval of XML Documents***.** Centre for Advanced Computer Studies, University of Louisiana.