Website Boundary Detection via Machine Learning

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy
by

Ayesh Alshukri

August 2012

Preface

This thesis is submitted to the University of Liverpool in support of my application for admission to the degree of Doctor of Philosophy. No part of it has been submitted in support of an application for another degree or qualification of this or any other institution of learning. This thesis is predominantly my own work and the sources from which material is drawn are identified within. Sections of some chapters of this thesis have appeared in the following publications as indicated:

Sections of Chapter 5 have appeared in:

[1] A. Alshukri, F. Coenen, and M. Zito. Web-Site Boundary Detection Using Incremental Random Walk Clustering. In *Proceedings of the 31st SGAI International Conference* (pp. 255–268). Cambridge, UK: Springer. 2011.

Sections of Chapter 7 have appeared in:

[2] A. Alshukri, F. Coenen, and M. Zito. Incremental Web-Site Boundary Detection Using Random Walks. In *Proceedings of the 7th International Conference on Machine Learning and Data Mining.* (pp. 414–427). New York, USA: Springer. 2011.

[3] A. Alshukri, F. Coenen, and M. Zito. Web-Site Boundary Detection. In *Proceedings of the 10th Industrial Conference on Data Mining.* (pp. 529–543). Berlin, Germany: Springer. 2010.

Abstract

This thesis describes research undertaken in the field of web data mining. More specifically this research is directed at investigating solutions to the Website Boundary Detection (WBD) problem. WBD is the problem of identifying the collection of all web pages that are part of a single website, which is an open problem. Potential solutions to WBD can be beneficial with respect to tasks such as archiving web content and the automated construction of web directories.

A pre-requisite to any WBD approach is that of a definition of a website. This thesis commences with a discussion of previous definitions of a website, and subsequently proposes a definition of a website which is used with respect to the WBD solution approaches presented later in this thesis.

The WBD problem may be addressed in either the static or the dynamic context. Both are considered in this thesis. Static approaches require all web page data to be available a priori in order to make a decision on what pages are within a website boundary. While dynamic approaches make decisions on portions of the web data, and incrementally build a representation of the pages within a website boundary. There are three main approaches to the WBD problem presented in this thesis; the first two are static approaches, and the final one is a dynamic approach.

The first static approach presented in this thesis concentrates on the types of features that can be used to represent web pages. This approach presents a practical solution to the WBD problem by applying clustering algorithms to various combinations of features. Further analysis investigates the "best" combination of features to be used in terms of WBD performance.

The second static approach investigates graph partitioning techniques based on the structural properties of the web graph in order to produce WBD solutions. Two variations of the approach are considered, a hierarchical graph partitioning technique, and a method based on minimum cuts of flow networks.

The final approach for the evaluation of WBD solutions presented in this research considers the dynamic context. The proposed dynamic approach uses both structural properties and various feature representations of web pages in order to incrementally build a website boundary as the pages of the web graph are traversed.

The evaluation of the approaches presented in this thesis was conducted using web graphs from four academic departments hosted by the University of Liverpool. Both the static and dynamic approaches produce appropriate WBD solutions, however. The reported evaluation suggests that the dynamic approach to resolving the WBD problem offers additional benefits over a static approach due to the lower resource cost of gathering and processing typically smaller amounts of web data.

Acknowledgement

First and Foremost, I would like to thank my primary supervisor Dr Frans Coenen for his extreme patience, encouragement and continued faith in me. I am very grateful for his constant guidance and support, I could not have started this journey, or seen it through without his help. I would also like to express my gratitude to my second supervisor Dr Michele Zito for his suggestions and valued comments. He has helped me and my research develop over the four years of my PhD study.

I am especially thankful to Hannah for standing by my side for the duration of my research, and for her extreme patience with my very, very long hours of study. I am eternally indebted to her for the support she continues to give me. Finally, I would like to thank my family and friends for the encouragement they have given me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The World Wide Web (WWW) is a prolific information source which provides a massive amount of information which is publicly accessible from almost anywhere, which makes it the largest data source in the world [49, 135, 128]. The past decade has seen an explosive growth of the WWW. This explosive growth has been fuelled by factors which include advances in hardware technologies, increases in the ease of use of software and the tremendous opportunities the web offers for business [50, 128]. The new generation of web applications, coined 'web 2.0', is also a contributing factor to this growth, with user authored content contributing significantly [139, 159]. User involvement in the use of the WWW is ever increasing [159]. It is integrated into everyday life; shopping, banking, conferencing and social networking can all be conducted online with many additional benefits [33].

Given that the WWW is the largest data source in the world there is a desire to both support the current benefits offered by the WWW (information sharing, networking and so on) and derive additional benefits using automated tools and techniques. Machine understanding and communication of data using sophisticated algorithms, not only makes possible, but can also enhance the services that are used by many users online today [33]. To achieve a high level of machine understanding of the WWW requires knowledge of the structure of the WWW. One way of modelling the WWW is to conceive it in the form of a *web graph*, where each node represents a web resource, and edges represent hyperlinks (each resource can be linked to many other resources via multiple edges). Using this interpretation of the WWW it is possible to design software tools (*web crawlers* [172]) that can automatically traverse the graph with the aim of collecting information.

The operation of many WWW tools (such as web archiving or indexing tools) can be enhanced if they can be provided with a clear notion of what a website is (see section 1.1), and what it's boundaries are. In some cases it is not unreasonable for this to be done by the user of the tool. In other cases it would be beneficial if this could be done automatically. The Website Boundary Detection (WBD) problem is an open problem (see section 2.2). The solution to the WBD problem is hindered by the lack of any clear understanding of what a WWW site is: the term means different things to different groups of users and different types of application.

This thesis seeks to provide an answer to the boundary detection problem by considering its solution in terms of a clustering (unsupervised learning) problem. A number of alternative possible solutions are presented, some operating in a static context and some in a dynamic context. The potential solutions include the adoption of established unsupervised techniques, such as the kmeans algorithm, and the use of more "up to the minute" techniques such as random walks. All the solutions are founded on a particular definition of what a WWW site is, the derivation of this definition also forms an essential element of the overall contribution of this thesis.

The rest of this introductory chapter is organised as follows. In Section 1.1 some further motivation, in addition to the above, is presented in terms of a number of specific applications. The research questions, and associated research issues, central to the work described in this thesis is then presented in Section 1.2. The research methodology adopted to address the research objective is then outlined in section 1.3. The contribution made by the work described is summarised in Section 1.4, followed by an overview of the remainder of this thesis in section 1.5.

1.1 Motivations

The work described in this thesis is motivated by a number of applications where WBD is central to the problem domain. These include digital preservation, web directory generation (indexing), web site map generation, web spam detection and various forms of web analysis.

Digital preservation is concerned with the archiving of digital information, such as web sites, for historical purposes. Examples of selective web archiving initiatives include LOCKSS[1] and OPF[2]. The general process of archiving digital content is a three step process: (i) collecting, (ii) storing and (iii) access. WBD is a central element with respect to the collection step. Currently data collection is usually conducted by manual traversal [46, 60]. Identifying the boundary of a website can automate the choice of pages to archive [168, 21].

Web directories are used by internet search engines to support web searching. Yahoo Directory[3] and Look Smart[4] are examples of online applications that operate using web directories. Currently web directories are often manually constructed by domain experts, comprising a small group of paid staff [117]. The Open directory project[5] is

[1]LOCKSS: Lots Of Copies Keep Stuff Safe. `http://www.lockss.org/lockss/Home`
[2]OPF: Open planets foundation. `http://www.openplanetsfoundation.org/about`
[3]`http://dir.yahoo.com/`
[4]`http://www.looksmart.com`
[5]`http://www.dmoz.org/about.html`

an effort to create the most comprehensive human edited web directory. This initiative is in response to the increasing rate of growth of the internet. The web directory is supported by a growing community of voluntary domain experts that have an interest in a certain topic. Such core web directories are typically used to power search engines. It would clearly be advantageous if the large scale generation of web directories could be automated. This in turn would require WBD.

A website map provides users with an overview of a website. Despite the obvious usefulness of such a resource, website maps are only available for a small number of websites; they can either be manually constructed or their generation can be automated [129, 64]. Constructing a site map for a website that has many different authors is difficult [130]. Automated solutions to site-map construction are often limited (by some degree) to domain name, however websites can clearly also be related regardless of domain name. For example Google site map generator[6] uses a technique that traverses a specific given URL domain to construct a site map. The resulting site map will thus not include content spread over different domains. The automated generation of web site maps would necessitate a solution of the WBD problem as envisaged in this thesis. This would offer a particular advantage with respect to websites that change rapidly.

Web spam (similar to email spam) is content on the web that is generally unwanted or fake, with possible malicious intentions. Detecting web spam can be very challenging and is a top priority for many search engines so as to stop this content from being ranked higher than legitimate content in user search results [35, 140, 52, 36]. Reported work on detecting web spam is typically directed at the detection of individual spam web pages as opposed to the identification of spam web sites [183, 182, 118, 181]. Solutions to the WBD problem would support the detection of entire spam websites.

Study of the WWW is conducted in many contexts, including authorship, accessibility, and structure analysis. Studying the world wide web at the website level rather than the web-page level can have a number of benefits [41]:

WWW Document Authorship Analysis Document authorship analysis is conducted with respect to a single authored document on the WWW. For example if we wished to analyse web document authorship, one publication can be represented by multiple pages on the web [63]. Thus it is not reasonable to study attributes like authorship at page level. Instead it might be better to study the web at the web-site level.

WWW Site Inter-relationship Analysis The inter-relationships between websites changes over time. A website entity may be reorganised at the site owners' control, causing pages and links to appear and disappear [128]. It is important to be able to identify website boundaries to accurately study inter website relationship.

[6]`http://code.google.com/p/googlesitemapgenerator/`

Connectivity Analysis It is sometimes important to analyse the connectivity between WWW sites, see for example [44]. An understanding of the boundaries of WWW sites is important to support such an analysis.

Statistical Analysis The statistical analysis of the WWW is a common application area. The study of the web founded on a statistical analysis of web-pages alone may be skewed due to the simplicity of rapid and dynamic generation. Studying the WWW at the web site level allows for better extraction of statistics such as measures of the growth of the WWW.

1.2 Research Questions

Given the above motivation the objective of the research described in this thesis can be summarised as the search for an answer to the question *is it possible to discover website boundaries using unsupervised learning techniques?*

This research question encompasses a number of component research questions:

1. **What is an appropriate definition of a website to be used within the context of this research?** From the literature there is substantial ambiguity concerning the definition of what is, and what is not, a website. The term website is widely used, but poorly defined; it means many things to many different observers. If we are to establish techniques for identifying website boundaries, a necessary precursor is to determine an appropriate definition for a website.

2. **What is an appropriate web page model to be used within the context of this research?** To identify website boundaries we need to model the domain in some way. More specifically we need to be able to model web pages, and by extension websites. A web page comprises many features: giving a definition of a website is also necessary to determine what features need to be taken into consideration if we are to build models of the web.

3. **What is an appropriate mechanism to deal with the magnitude of the web?** The size of the web is growing exponentially. There are a vast number of websites which means that if we wish to, in some sense, process the entire web this will entail an unacceptable computational overhead. An alternative mechanism is therefore required.

In the context of this research a practical definition of what is a website is proposed. This proposed definition allows for the WBD problem to be defined in terms of a clustering problem. The scope of the work described in this thesis is therefore limited to using unsupervised learning techniques. More specifically the work is directed at the identification of website boundaries using unsupervised learning techniques in both the static and dynamic contexts.

1.3 Research Methodology

This section presents the methodology adopted with respect to the research described in this thesis to address the proposed research question. A 5 phase programme of work was adopted. A necessary precursor to any investigation to address the WBD problem, as already noted, was to establish a definition of what a website is. This then comprised Phase one of the programme of work. Phase two was then directed at an investigation of the attributes most suited to defining a website, which in turn would influence the nature of any solution to the WBD problem. As already noted the solution to the WBD problem, with respect to this thesis, is conceptualised as a special type of clustering problem.

The remaining 3 phases of the programmes of work were directed at a potential solution to the WBD problem. Potential solutions were categorised as being either static or dynamic. Because static solutions were considered as a precursor to dynamic solutions these were considered first.

The first two proposed approaches to produce potential solutions to the WBD problem were directed at the static context, The first of the static approaches concentrated on investigating the best features to represent web pages. In combination with clustering algorithms the most appropriate features to represent web pages where found with respect to the WBD problem. The second approach, in the static context, was centred around the graph structure of the web, induced by the hyperlinks between pages. Graph partitioning algorithms were used to investigate the potential segmentation of the web graph in order to exploit the structural properties exhibited in the web with respect to the WBD problem. The final approach is considered in a dynamic context. Both the structural properties and features of the web were used in order to perform a "real time" traversal and incremental clustering of the graph in order to produce WBD solutions.

To evaluate the proposed techniques a number of data sets were used. Firstly synthetically generated binomial random graphs and artificial graphs, based on the preferential attachment model, were created. Secondly a set of real web data was obtained by collecting a portion of the WWW. The criteria used to measure the effectiveness of the proposed techniques was founded on the following:

- Performance of the website boundary detected with respect to measures.
- The total ratio of class and noise pages covered (requested and processed).

The performance of a website boundary was measured using supervised learning calculations that quantified whether the boundary solution included all the relevant (labelled) web pages relative to how many noise web pages were included. The ratio of class and noise items covered was designed to measure relative coverage of web pages associated with a WBD solution. The ratio was used to quantify the unnecessary processing of noise pages, and show the number of web pages that have been covered relative to noise.

Intuitively the reader may postulate that traditional unsupervised learning measures (cohesion, separation [95]) could be used to evaluate potential solutions. However, it is argued here that such measuring was not appropriate to measure the performance of a WBD solution. These measures are traditionally associated with problems that have no labelling associated with the data, therefore these measures are typically used to determine the quality of a solution relative to the data itself [176]. Also each of the data sets have an associated labelling of target and noise web pages. Given that the main objective of the research described in this thesis is to detect, with high performance, the boundaries of websites; this can be measured using traditional supervised learning measures using labelled data. A subsequent objective of the work is to deal with the large volume of data on the web, which can be interpreted as a desire to reduce the expense of unnecessarily processing noise items; this can also be measured using labelled data.

1.4 Contributions

This thesis makes a number of contributions. Firstly the work demonstrates that, given a definition of what a website is, it is possible to detect the boundaries of a website in a static context by applying unsupervised clustering techniques. However, in the static context, it is often impractical to pre-determine the collection of pages required to identify the desired boundary because: (1) there is uncertainty in "grabbing" data of an unknown size, implying great difficulty in pre-determining what to include in the collection of pages; and (2) it is resource intensive to process potentially large amounts of unwanted data that may be contained in the pre-determined collection. Thus the second claim made in the thesis is that website boundaries can also be identified using a dynamic process. In the thesis it is argued that by using a clustering algorithm and content information coupled with a random walk web graph traversal, in a dynamic incremental context, it is possible to produce a solution to the WBD problem with satisfactory accuracy while at the same time avoiding unnecessary processing of unwanted data. It is also argued that the proposed dynamic approach to WBD is more desirable because: (1) there is no need to have any knowledge concerning the web site to be identified prior to producing a solution to the WBD problem, and (2) the process requires a smaller computational resource than in the static context.

In summary the contributions of this thesis can be itemised as follows:

1. A proposed website definition which is used with respect to the rest of the research described in this thesis.

2. An investigation of the web page features that can be used to support the proposed definition, which can consequently be used to effectively model web pages for the purpose of WBD.

3. An investigation into mechanisms to identify website boundaries in the static context, using the kmeans clustering algorithms, combined with a particular set of web page features.

4. An investigation into mechanisms to identify website boundaries in the static context using the structural properties encoded in the hyperlink web graph.

5. An investigation of mechanisms to identify website boundaries in the dynamic context using a random walk graph traversal method, combined with the kmeans clustering algorithm and using textual features to represent web pages.

6. Evidence that the proposed dynamic mechanisms offer advantages, with respect to the practical challenges of WBD, over static mechanisms.

1.5 Organisation of Thesis

The organisation of the remainder of this thesis is as follows. Chapter 2 provides some background and related work to the WBD problem, and the approaches used in this research. Chapter 3 presents a discussion of the WBD problem with respect to the definition of a website, and gives some information concerning the static and dynamic contexts. Chapter 4 presents the synthetically generated data sets, and the real web data sets which are subsequently used to evaluate the proposed static and dynamic approaches. Chapter 5 presents the first approach to the WBD problem, in a static context, and concludes with the most appropriate features to represent a web page with respect to the WBD problem. Chapter 6 presents the second static approach, founded on the structural properties of the web graph, using two graph partitioning methods. Chapter 7 presents the third proposed approach, which is directed at the dynamic context. This approach uses both the structural properties and the features of web pages to traverse the web graph using probabilistic methods. In this approach the graph is clustered in real time using an incremental clustering algorithm to produce WBD solutions, while at the same time aiming to reduce the amount of noise pages visited. Chapter 8 in this thesis presents a conclusion and the main findings of the research, followed by some suggestions for future work.

Chapter 2

Background

This chapter presents the necessary background information that underpins the work described in this thesis. The chapter commences (section 2.1) with consideration of the initial inception of the WWW and how it has evolved and its continuing impact on society. Section 2.2 then considers previous and related approaches to solve the Website Boundary Detection (WBD) problem. The Knowledge Discovery from Databases (KDD) process model is a general model outlining the steps to extract interesting patterns and new knowledge from data. The steps of the KDD process are presented in section 2.3. Data mining is a specific step in the KDD process, and is concerned with the actual pattern discovery in data. Web mining is a related field to data mining and, given its relevance to this work described in this thesis, is reviewed in section 2.4. Section 2.5 presents some background information on web crawling, introducing a random method of crawling the web, which has significance with respect to one of the proposed approaches presented in this thesis. The solutions to the WBD problem considered in this thesis viewed the solution to the WBD problem in terms of a clustering problem. Section 2.6 therefore considered the clustering problem in relation to this research, and further explains the static and dynamic contexts in which it is used in this work. The final section of this chapter (section 2.7) gives a summary of the previous work presented in this chapter.

2.1 The World Wide Web (WWW)

The World Wide Web (WWW) was invented by Sir Tim Berners-Lee in 1989 when working at the Center for European Nuclear Research (CERN) [37]. The name 'World Wide Web' was the term he coined for the project. The WWW (or web) is the largest and best known repository of hypertext content in the world [128, 49]. The content is distributed and heterogeneous in its nature. The integration of the web is apparent in almost all aspects of society [39, 40, 94, 170, 171].

Publication of content onto the web follows a simple concept in which hypertext documents can be linked to other documents that are distributed across the web [49].

The publication of content on the web is not controlled by a central authority or administered in any way [50]. Due to the lack of editorial control it allows the publication of content on any topic in any format [128]. The easy and unrestricted publication of content provides a platform that can have advantages for many applications, and can provide a great future for open and democratic communication from any participant.

The organisation of information that is distributed on the web is determined by the author or publisher of the information. An example of which is the organisation of the content in collections of web pages that constitute a website. This content can be (hyper) linked to other related content across domains or other physical locations as directed by the publisher or authoring body. This organisational structure is generally understandable by other human users. However this open and unrestricted environment can also have adverse effects when looked at from an information retrieval point of view, as there is no strict and enforced guidelines as to how content is to be published, described and disseminated on the web. Also the size and growth of the web means that desired content cannot be easily accessed by users, and thus it is essential to use software to aid information retrieval from such a vast data source [82, 26].

2.2 Website Boundary Detection (WBD)

The Website Boundary Detection (WBD) problem is an open and difficult problem to solve [41, 28, 98, 104]. As already noted the WBD problem is commenced with the task of identifying the collection of all web resources that are contained as a single website (see section 3.3 for further discussion on the WBD problem). To the author's knowledge there is currently no automated machine learning technique to detect the boundaries of a website with the precision of a human user.

The field of "website mining" [75, 178, 54, 158] describes an area of research that falls under the main field of web mining (see section 2.4), but is directed at specific applications with respect to a website's structure (in contrast to web pages for example). The general problem of website mining is to discover hidden patterns or knowledge in relation to websites. Website mining techniques utilise web structure (hyperlinks) and web content (attributes of a page).

There are many website mining applications presented in the literature, which can include evaluating the quality of websites [101, 155] or detecting the structure of websites [142, 88] to name but a few. Although each of these problems are related to websites, the specific problem of defining the boundary of a website, which is often a necessary precursor, is often solved manually or overlooked completely.

There are several approaches and related research that attempt to provide solutions to the WBD problem. Section 2.2.1 describes some of the most relevant approaches with respect to the context of the research presented in this thesis. Section 2.2.2 then presents a review of previous research on the types of features that have been used to model web pages, which is directly related to the approaches presented later in this thesis.

2.2.1 Related Work

This sub-section first presents some selected related work on the WBD problem. The related work discusses: link block graphs, logical websites, compound documents, directory based websites and website hierarchies. Secondly some related work that focuses on related website mining issues is presented with respect to website classification.

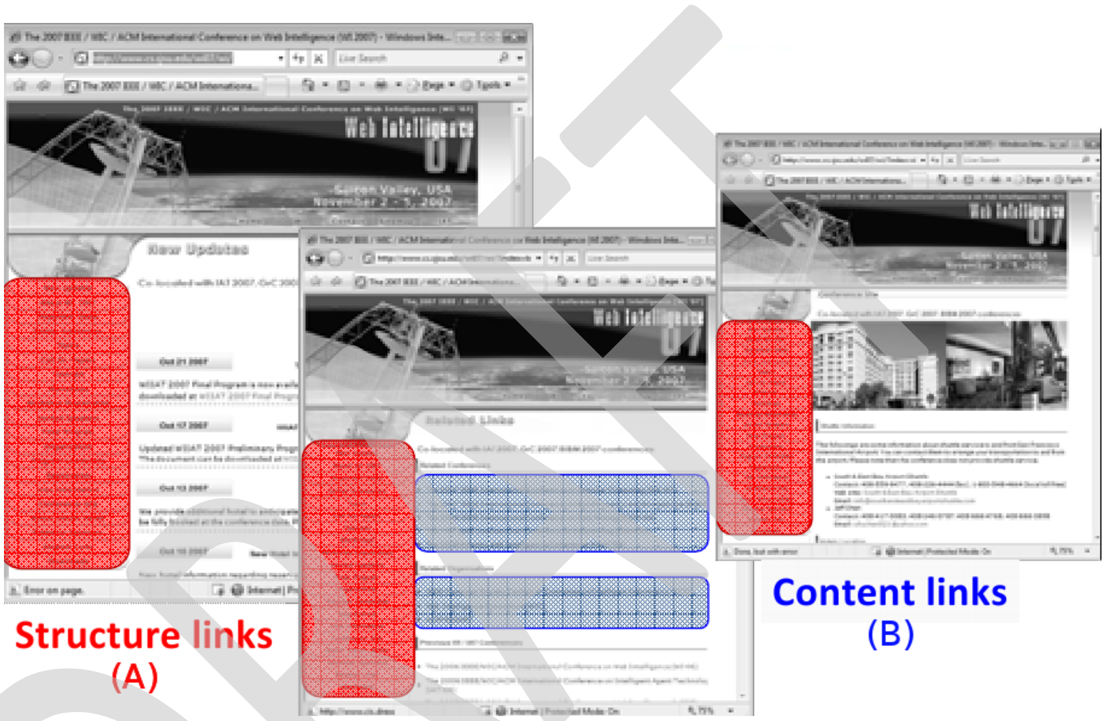


Figure 2.1: An illustration of individual web pages segmented into (A) structure link "blocks" (Red) and (B) content link "blocks" (Blue) [162]

Link Block Graph The WBD problem has been approached by using methods based on the link block graph representation. The link block graph is created from web pages by segmenting individual pages into "blocks" representing navigation menus or sets of coherent links (Figure 2.1). This is typically done by segmenting the HTML Document Object Model (DOM) of web pages. Once segmented a graph structure can be used to represent the link blocks (a link block graph). The individual blocks represent vertices, and the connections represent links between blocks.

In the work by Rodrigues [161] and related work by Keller [105] the main emphasis was concentrated on link blocks that represent structural menus (s-menus) of the web pages (an s-menu is restricted to only structure link "blocks" (A), as shown in Figure 2.1). A link between s-menu exists when a link from a s-menu can be used to navigate to another page that contains the same s-menu. The pages that contain this menu are said to represent the skeleton structure of some hidden content hierarchy defined by the particular s-menu. Web pages that contain common s-menus can define the boundaries of a website. While s-menus that are present on only a smaller subset of pages within a website represent sub-sites.

In [105] a menu miner algorithm is used to decode the architecture of websites, and produce a content hierarchy. The algorithm uses a bi-directional graph of the web, where vertices represent s-menus, and a link exists only if it is reciprocated (Figure 2.2). The content hierarchy is found by identifying pages containing common s-menus by discovering maximal cliques in the graph. The content hierarchy is then used to detect the boundaries of a website by using a disjunctive set of pages defined through the identification of common s-menus.

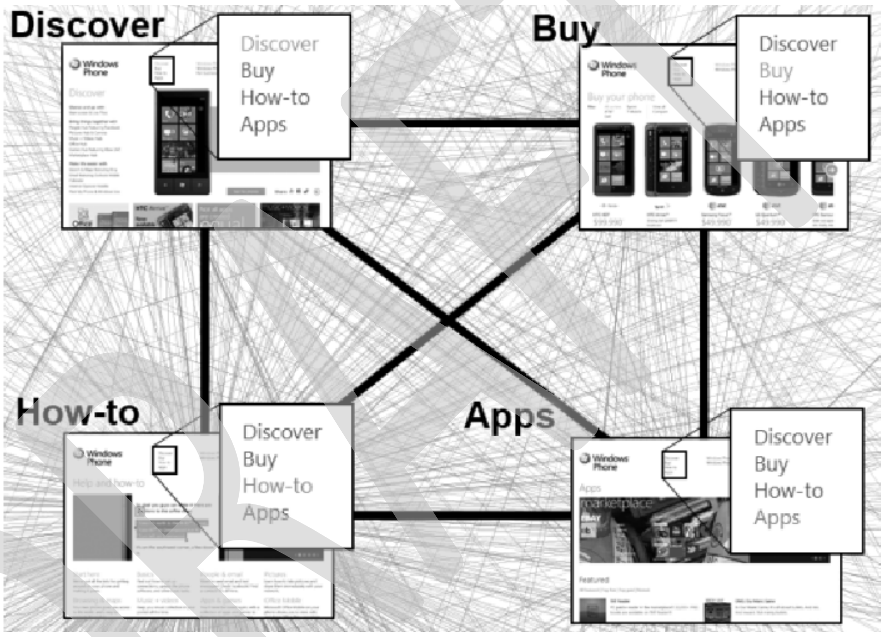


Figure 2.2: Illustration of a collection of four web pages (Discover, Buy, How-to and Apps) connected in a link block graph due to the presents of a common s-menu [105]

In [161, 160] identification of Strongly Connected Components (SCC) in the link block graph is performed using a linear time algorithm. A sub site is then detected by creating the union of all web pages that contain the particular s-menu in the identified SCC. Both of the proposed methods require only the preprocessing of a web page's HTML structure in order to segment them into coherent structures representing link blocks.

The static nature of the above described approaches is apparent. In the work by Rodrigues [161] and related work by Keller [105] both proposed approaches that require the static preprocessing of a collection web pages in order to segment them into coherent structures represented by their corresponding link blocks. Also if pages are from the same website, but do not contain a common s-menu, they will not be detected as being within the same website boundary.

Logical Websites In the work by Senellart [167, 168], the aim is to find web pages that are contained in "logical websites". A logical website is described in terms of the link structure and is defined as a collection of nodes (or web pages) that are significantly more connected than other nodes. A gathered portion of the web is modelled using flow networks. The assumption is made that if a flow is pushed around the network, bottlenecks will occur in the less connected noise pages, but flow more freely around the highly connected target pages of the website. To detect the boundaries of a website flow is pushed from a set of seed pages until a bottle neck is created around the boundaries of the website. The set of nodes that have flow pushed to them are then identified as part of the website.

Compound Documents Research by Eiron [73] proposes the notion of *compound documents*. This is a set of web pages that can be aggregated to a single coherent information entity. A simple example of a compound document is a news article, although it is created conveying ideas in text format, it could also have illustrative elements like images or video. This collection of web pages aggregates to form a compound document representative of a news article.

In the work by Dmitriev [63, 65] a method to discover compound documents is proposed. The method uses supervised learning techniques to train a model using manually labelled compound documents. The correct weighting of edges between related pages from the input labelled compound documents are assigned based on extracted features. The features used were web page content similarity, along with anchor text and title similarity. Features based on structural components were also considered, they include the number of common in and out links, links with common anchor text and the number of hyperlinks pointing to common directories [66].

The model is then applied to a new set of pages to calculate edge weights based on features which relate pages of compound documents. A clustering method is then applied to partition the model into compound documents using the weighted edges.

The features used in the research described above concentrates on the textual similarity that can be extracted from content contained on the web page. The features do not consider any styling or meta elements of web pages, for example image links or styling scripts. Such styling features are considered in the research presented later in this thesis where it is shown that using additional features can prove effective in the discovery of website boundaries.

Directory Based Website In [28] the difficulty of the WBD problem is enforced by a claim regarding the vague nature of the current definitions of a website. The proposed solution is to define a "directory based (web) site". This is described as a section of a web server over which an author has full control. The method proposed to identify the website is to use various filters based on characters in the URLs of web pages to categorise which child pages are under the control of a certain author. For example in URLs the tilde symbol ($\sim$) is sometimes used to represent a user. Subsequent pages in the following directories can be identified as being under the control of that user. This method is consequently limited by a restricted definition which does not allow flexibility for a website to span beyond multiple domains or directories.

Website Hierarchies Mining website hierarchies is an area of research that aims to extract hierarchical structures from the web pages contained in a website. A hierarchy is a tree structure that can be used to represent the organisation of a collection of web pages based on the link structure. A method of extracting a hierarchy from a collection of web pages of a website is proposed in [130, 131, 188]. The method uses a machine learning approach to weight edges based on the different predetermined categories of links. A shortest path algorithm is then used, rooted at the home page, to build the tree structure to represent the hierarchy of the website.

In [54] an approach to produce clusters of hierarchies for a website using a link structure is proposed using the ReHITS method. The ReHITS method comprises an iterative application of the HITS algorithm which is used to discover authorities and hub pages in the website. These authorities and hubs are then used to represent a cluster. The HITS algorithm is then reapplied after removing the previously identified authorities and hubs. This process is repeated to build a set of clusters. Sub trees are formed using the path structure of the urls of pages in a cluster. If the paths of two authorities overlap they are merged into one sub-tree. This method is used to identify logical sub-trees representing clusters within a website.

In [113] a method of segmenting the URL tree of a website into topical cohesive regions is proposed. This produces a hierarchy that reflects topical sub-sites of a website. The algorithm requires an existing tree structure of the web pages in a collection, this can be retrieved according to the URL of the pages in a website. The approach segments the given tree into collections representing topic related sub collections using methods based on recursively traversing the tree, selecting to add nodes based on a cost measure associated with adding nodes to a sub tree.

The methods presented above make simple assumptions about the web pages contained in a website prior to applying the hierarchy extraction. The collection of pages making up a website is assumed to be given prior to applying extraction methods. There is no mention of an attempt to apply these methods to extract website hierarchies where the collection of pages making up a website is hidden within a larger collection of noise web pages. Noise pages in this thesis are referred to as unwanted web pages that are irrelevant to the goal of WBD, i.e. noise pages are outside the boundary of a website.

Website Classification The topic of website clustering/classification is an interesting topic related to the WBD problem. The aim is to label a website with a topic or subject, this is in contrast to web page classification which aims to do the same but for a single web page [156]. The vast majority of the literature on this subject aims to categorise the subject matter a website describes, or classify the website into predefined groups. Unfortunately much of the work simply makes an assumption concerning the boundaries of the website before any classification techniques are applied [152, 111, 75, 27, 115, 126, 179]. Therefore the problem of detecting the boundaries of the website in question are often overlooked, or an estimation is used as to how many pages should be collected from a certain domain to represent the website [127].

2.2.2 Web page features

This sub section presents some related work on the features that have been used to model web pages. This is an important aspect with respect to this thesis as the features used to model web pages will directly effect the measure of similarity between two pages. The notion of similarity used will inevitably have an impact on the pages identified to be within a website boundary.

A common approach to represent web pages is to use methods that are used in text mining. Traditional text data mining methods are applied where a web page is modelled as a text document. This document model is then used to determine the similarity between a set of web pages [50, 49, 135, 128]. Despite the application of text mining approaches, web pages can be distinguished from traditional text collections [156] for the following reasons:

- The web contains pages that are unstructured or semi structured with inconsistent formatting.
- Web pages usually contain markup code that is rendered visually for users.
- Pages on the web exist within a web graph of rich hyperlinks pointing to and from various pages.

The facts outlined above present the reasons which distinguish web pages from traditional text documents. In summary there are many more elements that make up a web page than are contained in a text document. Therefore a further investigation into additional features to represent web pages is justified [156]. The additional attributes that distinguish web pages are usually contained “on-page” and encoded in the web pages content in some way. A review of features that have been used in the literature for web page modelling is presented below.

Text The text of a web page is the most straight forward feature to use when modelling web pages. Hence the common application of traditional text mining techniques. Using textual content allows for the application of text mining methods, for example the bag of words model. The noise that is associated with data from the web can often hinder these techniques [156]. The technique of using n-grams has also been applied to web pages [141]; a technique which can be used to capture concepts and/or phrases. A draw back is that the n-grams method usually creates a higher dimensional feature space than in the case of the bag of words method. High dimensional data can introduce additional problems, particularly when using traditional similarity measures with respect to the application of clustering algorithms [110]. Therefore further techniques need to be applied to reduce the dimensions of the feature space, for example feature selection techniques [156].

URL Arguably the most prominent feature of a web page is its Uniform Resource Locator (URL). The feature is typically used for web navigation purposes [34]. This feature is not strictly encoded in the web page’s content, but has to be known in order to request web page content. The main elements making up a standard URL consist of: scheme, domain, sub domains, directory path and query. The URL `http://www.my.examples.com/example?query_string` is made up of: a `http` scheme, the domain and sub domain is `examples.com` and `my`, the path is `example` and the query is `query_string`.

There are techniques that have been successfully used to classify web pages using features constructed from the various elements making up a URL. Creating features based on the domain, sub domain and directory structure have been used [28]. There have also been methods that extract features from the URL by segmenting the various elements of the URL using delimiters.

Meta data In [90] the meta data that is sometimes embedded in web pages was used as a feature. Web pages were represented using: title, headings and meta data along side the main body text. Each of these features can be extracted from the content of a web page if available by focusing on what is contained within the HTML markup of a web page. The meta data features can include keywords or even a description of the content of a web page. It is concluded that a combination of the features should be used in order to gain the best results in terms of web page classification [90].

Styling tags In [116, 115] web pages are represented according to the presence of certain tags in the content. Some common features (title, headings, and meta data) were used in this work. However, additional features based on tags in the HTML content were also used. Web pages are segmented into features representing certain tags in the textual content. For example textual styling tags like bold, underline and strong may be extracted from a web page [116]. A weighting scheme is used to distinguish the significance of the tags used. If a tag is used very often, the assumption is that it is not used for emphasis, thus it is deemed unimportant, and the weighting is reduced [115].

Content and Structure The work described in [69, 68, 67] uses a hybrid approach to representing features from a web page which draws upon both the content and the internal structure of web pages. The content is represented by features constructed using keywords from the text. The structural properties are used to create features using page anchors. Page anchors are used as links "jumping" the user to various points in the same web page. This type of feature representation produces a small dimensional feature space, and does not involve the complexity associated with modelling the hyperlink structure [67].

User behaviour In [169] numerous sets of common features are used to model web pages, in particular they are used in combination with users navigation behaviour. The URL-based features are used as in previous work, but the features are constructed based on the depth of URL paths. The "type" of a web page is also used. A feature is constructed using the file extension, which can indicate the type of resource at a URL, for example .pdf, .jpg and so on. Features are constructed using the textual content in a page anchor (named entities, nouns and verbs). The web page is used to construct a DOM, which is then used to represent block based features. Links based features are used (incoming links). User behaviour features collected from browser toolbars are also used as features, examples include; the number of page visits over a certain time period, number of links clicked on each page and pages visited from bookmarks or browser history. Notice that the features used, and collected, are quite extensive. The research was conducted in part by Yahoo!, which is a major search engine based in the US. This work performs clustering on the features using a term-term co-occurrence model, which is created using a bag of words.

Other work There is other work that uses features based on the HTML structure of web pages [183]. These techniques concentrate on creating trees from the internal nested structure of web pages, and are less focussed on expressing similarity relationships between content in the pages. Theses techniques have been used successfully in identifying spam on the web [181, 182]. There is also an area of research that represents web pages based on visual properties [109], [29]. The techniques used are said to be expressed from user oriented point of view, as they derive features based on what is visually rendered. This point is in contrast to the majority of the features presented above, where the object is to encode content [156].

2.3 The KDD Process

The phrase Knowledge Discovery in Databases (KDD) was first used in [150]. KDD is described as “the process of finding useful information and patterns in data” [77]. KDD is a complete process involving a set of specific steps. There are various models that may be used to describe the KDD process. The most notable is the process-centered view model, shown in figure 2.3, which will be used to provide an overview of the KDD process with respect to this thesis. Other KDD models include: the Johns process model [102], the Human-centred process model [42] and the CRISP-DM process model [51].

Each of the steps of the KDD process shown in figure 2.3 is described in more detail below. The discussion is primarily founded on the review of KDD presented in [71, 95], but with additional information garnered from [78, 79, 151, 42, 124]. With reference to figure 2.3 the first thing to note is that KDD can be viewed as an iterative process, which means that when following each step the user can return to any of the previous steps if needed. The flow of information in the process-centered view model is indicated by the direction of the arrows in figure 2.3.

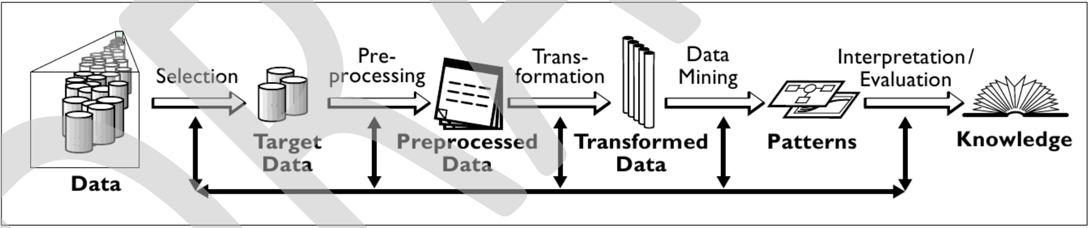


Figure 2.3: The Process-Centred view model of the KDD process [77]

The initial step of the KDD process, not shown in Figure 2.3, is to develop an understanding of the application domain [78]. Understanding the task that the user wants to accomplish is very important, this can be even more significant when the data is to be imported from multiple or complex data sources [151]. As the problem is investigated the goals of the data discovery task will emerge [42]. To understand and learn the application domain, the goals of the application and relevant prior knowledge need to be identified. For example to apply the KDD process to some data on the WWW, the goals of doing such an activity should be understood. This could be user traffic analysis from server logs or hyperlink structure to identify authority or hubs. The data from this domain must also be considered, this includes the format, and any particular techniques to process or gather the data. Once the application domain is understood the KDD process can commence. Thus:

1. **Selection** In this step we gather the "target data" needed for the knowledge mining [151]. This could involve the task of selecting the sources which are needed for the desired data discovery. The data may come from many different origins such as databases, flat files, the WWW and non-electronic sources [71]. The selection step may also include the identification of a target data set by focusing on a subset of the different sources of data [78, 79].

2. **Pre Processing** The data gathered in the previous step may be noisy. For example the data may include incorrect or missing values [95, 71]; or anomalies resulting from the use of different data sources, different data types and different metrics [95]. A strategy has to be adopted to deal with these problems [78, 79]. Erroneous data could be corrected or removed [95], missing data can be supplied or predicted. Care must be taken during this step as "occasionally what looks like an anomaly to be scrubbed away is in reality the most crucial indicator of an interesting domain phenomenon" [42].

3. **Transformation** The data that has been cleaned during the previous step typically needs to be converted to a common or more usable format [95]. Methods for "data reduction" may be used to reduce the dimensionality of data [95, 78, 79]. The transformation step can also involve finding useful features in the data to best represent it according to the intended goals of the KDD task [78, 79].

4. **Data Mining** The data mining step is where the actual knowledge discovery takes place. The typical aim is to extract useful patterns from the given data set using some particular data mining algorithm(s). This step thus commences with the identification of the most appropriate data mining algorithm given the desired KDD task, which is then applied to the pre-processed data generated by the foregoing steps.

5. **Interpretation/Evaluation** Testing and verification of the discovered knowledge is important so that the end user can gain confidence in the generated results. Different evaluation techniques are applicable to different types of data mining (KDD). How the patterns that are found in the data mining step are presented to the user is very important [71]. The usefulness of patterns, and the new knowledge that they represent, depends heavily on the interpretation placed on them by the end user. To assist interpretation various visualization techniques have been proposed founded on the use of graphs, charts and GUI's [71].

6. **Knowledge** This last step comprises the integration of the discovered knowledge with existing domain knowledge [151]. Methods for integration include simple

documentation of the knowledge discovered, or specific action taken as a direct cause.

As mentioned previously, the KDD process is iterative; this means that it can require interaction with the user between steps thus incorporating flexibility into the model [42]. There is no single KDD solution that fits all KDD tasks. The diversity of KDD goals, and their related complexity in terms of data types and knowledge interpretation, requires a tailored approach [53]. KDD solutions are typically constructed and developed for a particular type of problem or data set [53]. A discussion of how the KDD process may be applied to the WBD problem is presented in chapter 3.3.2.

In this section the KDD process has been described. In the following section the specific discipline of web mining will be considered.

2.4 Web Mining

Web mining is the task of discovering knowledge from web hyperlink structure, web content and web usage data [128]. Web mining can be thought of as the data mining step of the the KDD process model, when the knowledge discovery problem is concentrating on data from the WWW [50, 49, 135, 128]. The specific techniques used to discover non trivial patterns in web data are known as web mining techniques, which can be commonly based on traditional data mining techniques, but not exclusively so due to the specific challenges the WWW brings [128]. Web mining tasks can be categorised into three main topics:

Web structure mining techniques exploit the hyperlink (or link) structure between web pages to discover useful knowledge about the web.

Web content mining draws upon the information in the content of web pages.

Web usage mining typically discovers patterns from the user access data associated for a particular web service.

Each of these tasks has an associated set of techniques. It has been suggested that the three category view of web mining techniques could be represented by a merged list representing just two main categories; these are (1) web usage mining techniques, and (2) web content/structure mining techniques [58]. The two category representation is suggested because for many applications the techniques used draw upon more than one category to provide a solution.

The approaches presented in the research described in this thesis also draw on more than one category of web mining technique. The approaches to solve the WBD problem presented in this thesis first draws upon isolated techniques in the web content (chapter 5), and web structure (chapter 6) mining categories. An approach is then presented which utilises techniques drawn from both the web structure and content mining categories (chapter 7).

Two sub problems of web mining [105] that are related to the strucutres of the web can be broadly described as the problems of:

1. Generating new structures
2. Mining existing structures

With respect to the generation of new structures of the web, techniques use existing structural properties and content relationships. Examples include ranking algorithms like Page-rank [43], HITS [106], site rank and variations [186, 185]. These algorithms aim to supplement the existing structure with models that provide new knowledge that was previously unknown. Mining of existing structures is concerned with the problem of extracting models that are known to exist in the web, but are hidden by navigation, or lack of explicit description. The work presented in this thesis falls into this later category, a website boundary is known to exist, but the boundaries are not explicitly described in the web.

2.4.1 Web Mining Challenges

The WWW as a data source has many unique characteristics in comparison to traditional data that is mined for knowledge [128]. These differences contribute to making web mining both challenging and a rewarding process in terms of new knowledge discovered. This section discusses the issues and challenges associated with web mining applications in general. For a discussion of web mining with respect to the WBD problem see section 3.3.1.

Volume The volume of data and information on the web is huge and it is growing incredibly quickly [128, 95, 49, 135, 30]. This exponential growth poses many scaling issues that are becoming increasingly difficult to deal with [30]. This makes the task of discovering new knowledge using data from the web by applying traditional mining techniques a very difficult task. Work is ongoing to develop more scalable methods of data storage and analysis to address the increasing volume of data on the web.

Diversity The availability of low cost storage and networking has lead the WWW into a liberal information culture; content generation and dissemination is extremely diverse [49]. Personal corners of the web exist that are provided by social or corporate hosts that can hold a diverse range of information about a user. Weblogs can hold thoughts on topics of interest and social networking can hold information on a users day to day activities. Data of almost all conceivable types is available on the internet, the range is ever increasing. Images, videos and audio files in many formats are just a few common types of data that exist on the web today [128]. The diverse nature of the web can cause problems with the integration of information; multiple formats can cause information to be duplicated which can lead to inconsistencies [128].

The complexities that are associated with the web and its pages of diverse information can add additional data mining complexities that are far greater than when mining traditional data. Web pages do not have a clear unifying structure associated with their content, which can make diverse information seem unrelated. Web pages can contain many different authoring styles and the content can vary dramatically, covering any subject area [95]. This vast range of formats and structures can cause issues that need to be carefully considered with respect to mining activities.

Semi-Structured data As originally proposed the web was intended to improve management of general information at CERN [38]. The overall strucutre provided flexibility and convenience for the management of very large amounts of data [135]. The web today can be considered as a huge digital library [95]. However, the tremendous number of documents that are held are not arranged or sorted into a particular order as in the intuitive notion of a library [95]. This is also true for the majority of individual web pages on the WWW, as they can quite often be un-structured [30]. This makes the web an abundant collection of data that comprises both structured and unstructured, well formed and non-well formed, content [128]. This makes the web a difficult medium to gather information from [50]. Specific methods have to be employed to help extract information from the semi-structured data on the web, which contrasts dramatically to the processes required to extract from traditional data sources.

Authority The WWW spans many nations and features users from a wide range of cultures and backgrounds. There is currently no existing methods for verification of information or editorial guidelines, no approval from authority of the content on the web [49]. Data can be false, or poorly written. Data can be inconsistent and become quickly out of date. These are all facets of the publishing medium that the web has become [30]. Traditional publication mediums, like print or broadcast, do not have the same downfalls as the internet [49]. The problem of finding authority within the WWW can be difficult to solve completely, as some of the problems are intrinsic to human nature [30]. A method that can be used to gain some sense of authority is to look at the structure of the web. Pages that are pointed to by many other pages can have an increased sense of trust. These web pages can be deemed to be of high quality or have a heightened sense of authoritativeness because of the sense of trust invested by the hyperlinks from other pages [128].

Noise The WWW is full of irrelevant or noisy data that is not useful to many applications of web mining [128]. The noise in web data can be a feature of either: (i) Content or (ii) Quality. The content often includes information that is not needed or is irrelevant for a task. When mining a web page for a specific element of its content there is usually additional information on the page that is not needed. Examples include navigation links, advertisements or copy right notices. Only the content that is targeted is deemed as useful, everything else is considered as noise [128]. The quality can often be misleading, which is a cause of the issue of authority as mentioned above. There is no quality control regarding the information on a web page, therefore any information can be published. This can make a lot of information misleading or incorrect [128]. Spam content can also contribute immensely to the control of quality of web content. Finding the exact information that is needed from the web is essential to getting the most accurate results possible when applying mining techniques. "The clearer the data set the better the information and knowledge that can be extracted from it" [50].

Dynamic and Distributed The WWW is highly dynamic and distributed in its nature [128, 95]. The content of the web is dynamic as it is subject to constant change [128]. The changing of content can be very frequent which is true for such pages as current affairs, news and stock market information to name just a few examples [95]. Keeping up with changes is very important for many applications of the web [128], out of date or redundant data is not useful for such time critical data mining applications.

The locations of data can be spread across many locations of the WWW. However, the request and provisions of this data is often seamless to the user's experience. Redirections and load balancing of server resources across many geographical sites is very common, and can create a large distributed network of information for a single corporation.

This dynmaic and distributed nature is further increased due to the rapid growth of the web [95]. A large number of pages are added to the web each day [135]. Also a great number of computers can be connected to the network at any time, these can be of varying platforms, and in various geographical locations. New computers can be very easily added or removed, and can be intermittently connected or reconnected at varying times [30]. This adds to the volatility of the dynamic and distributed web. In addition broken links and relocation problems can occur due to moving computers, or when domains and files change names or disappear [30]. Multiple pages across the web may present the same information and could be inconsistent due to translations or mistakes, integration of such distributed data can pose a problem for mining tasks [128].

Virtual Society The WWW does not only contain information and services, it also hosts virtual societies and communities that are conceptualised on the web [128, 95]. Theses societies feature a multitude of interactions between people, organisations and automated systems [128]. They can be used to communicate instantly and express a user's personal view on many different subjects [128]. These interactions are important aspects of the characteristics of the web. for example they can encapsulate likes or dislikes for certain products or services. This information can be very valuable with respect to web mining applications.

Persistance Information that is available on the web has some similarities to published works, but they do not share the same immutable properties. The information does not disappear when it is first broadcast, nor does it remain in an imperishable state. The nature of the WWW provides a third model that coexists between the recorded and the unrecorded [107]. This model reflects the nature of the information on the web and the persistence it exhibits. Data can be created and maintained for ever, or can be destroyed in the same instance, and every time frame in-between this. This property means that new knowledge discovered may become out of date or irrelevant very quickly.

Deep web The WWW contains data that is not accessible using standard means of web navigation. The traditional method of navigation is conducted by moving from one web page to the next following available hyperlinks, this is known as the surface web. There is however information that cannot be accessed via hyperlink or can only be accessed using form submission; information that is dynamically generated as a result of web interactions. This is known as the "deep web". It has been estimated that the deep web is 500 times larger than the surface web [97].

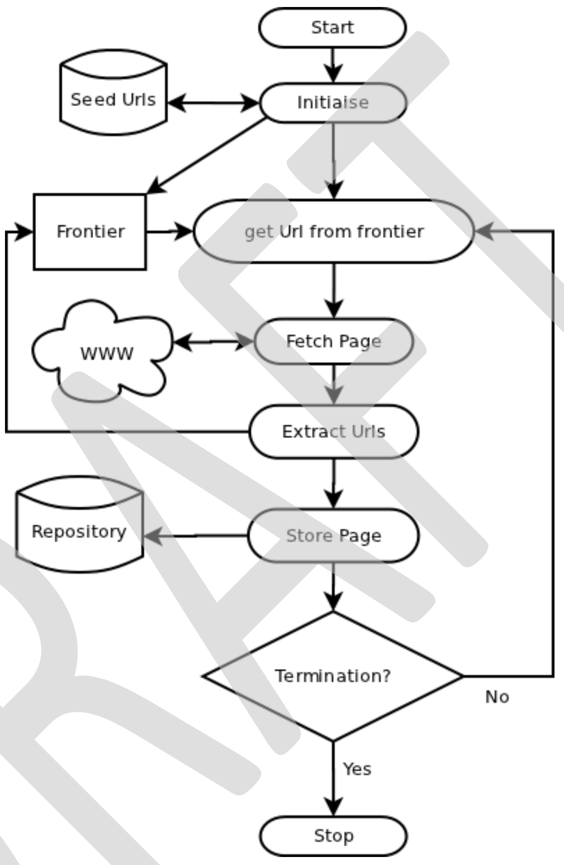
2.5 Web Crawling

Web crawling is the process of automatically traversing (crawling, spidering) the hyperlink structure of the web whilst downloading web pages, which can then be used with respect to various applications. Web crawling provides a method of collecting data from the web. There are three main categories of tasks that a web crawler might be used for [172, 128]: (i) broad, (ii) focused and (iii) continuous. A broad crawl aims to cover a wide area of the web. A focused crawl aims to cover a smaller deeper collection of pages, which has a distinct relation to a specific task. A continuous crawl is defined as covering a portion of the web at repeated time intervals. The approach used with respect to the work described in this thesis uses focused web crawling, as a specific target collection of pages is required for the task of WBD.

2.5.1 Basic web crawler

A basic web crawling process starts at some set of seed web pages. The links from within the pages are extracted, and used to fetch further pages. This process is repeated until some end criterion is reached.

Figure 2.4: The operation of a basic web crawler (based on [128]).



A basic web crawler is illustrated in Figure 2.4. The web crawler maintains a set of links (URLs) in the "frontier". These URLs are selected in some order according to which page should be visited next. Initially the frontier is populated with the set of given seed URLs. The main loop repeats the process of: (1) getting a URL, (2) fetching the resource that resides at the URL and (3) parsing the content of the resource for any further links to traverse, which are then added the the frontier. This process is terminated at some given criteria, for example, according to some time constraint or the capacity of the repository, or simply because there are no more URLs in the frontier.
Types of web crawler A web crawler is basically a graph search (traversal) algorithm, and so graph traversal techniques can be applied to crawl the web. The are various

types of web crawler that can be implemented using the basic process shown in Figure 2.4 There are two main categories of web crawler:

(1) Universal Crawlers are general web crawlers that aim to cover as much of the web, in the most efficient and fastest possible way. An example in which a universal crawler is used is to build a search engine index, which aims to be as comprehensive and relevant as possible. These types of crawlers are used for broad and continuous crawling of the web.

(2) Preferential or Focused Crawlers are web crawlers that select the URL from the frontier in some biased way. The biasing can be done using many different methods, and for many different goals. An example is to calculate a probability value/score for each URL in the frontier based on a notion of similarity. A (cosine) similarity can be assigned to pages residing at a URL with respect to a given topic of interest, using keyword relevancy for example [128, 147]. A probability can then be calculated based on URLs which are more likely to contain content on the desired topic. The URLs can then be preferentially selected from the frontier (and subsequently crawled) based on this probability. This process is known as focused web crawling.

Web Crawling Issues There are many practical issues related to crawling the web, some of these issues are presented in the list below:

- Coverage - visit as many pages (of importance) as possible in a timely and efficient manner.
- Importance - pages that are deemed important for a goal need to be identified to be crawled.
- Freshness - keeping the local index of web pages as relevant as possible.
- Fetching - HTTP requests, time outs, download sizes and so on.
- Parsing - from simple extraction of URLs to Document Object Modelling (DOM) of web pages.
- Stop word removal and Stemming - methods to apply to the content of a web page.
- Link extraction and Canonicalisation - reduce links to some consistent format.
- Spider traps - dynamically generated content can cause an infinitely traversable sets of links.
- Page repository - storing pages efficiently and in a format which allows for future processing if need be.
- Concurrency - network bottle necks , CPU and disk operations can each hinder a web crawlers performance.

2.5.2 Random Walk

As stated above, a web crawler is simply an implementation of an abstract graph search (or traversal) method but applied to the WWW. In this section a random walk method of web graph traversal is explained. Given an arbitrary graph G, the sequence of vertices visited by starting at a vertex and then repeatedly moving from one vertex to the next by selecting a neighbour of the current vertex at random is termed a *random walk* on the graph G (see for instance [133]). Random walks arise in many settings (e.g. the shuffling of a deck of cards, or the motion of a dust particle in the air) and there is a vast literature related to them (the interested reader is referred to the classical [81], or the very recent [24] and references there in). In particular random walks can be used [25] as a means for exploring a graph.

It is well-known [25] that any random walk on a finite connected graph eventually visits all vertices in it. Random walks, as defined above, are examples of so called Markov stochastic processes [81]. The evolution of a process of this type is fully determined by its current state. Assuming a graph is completely known to the walk, every time a vertex is visited there is an equal chance of moving to any of the connected links.

Random walks have been used to traverse the web on a large scale [99]. A random walk on the web graph is not as simple to perform as it may seem. In a true random walk pages would be selected uniformly at random from the graph, which does not happen when using the web. A random walk on the web graph suffers from: (1) starting state bias, and (2) can only randomly move to pages that are known so far. These aspects relate to the fact that until the whole web graph is known, including all the in and out hyperlinks of each page, a true random walk cannot be performed.

2.6 Clustering

Clustering is an unsupervised learning technique by which, given a set of objects, each object is assigned to a group (called clusters) so that the objects in the same cluster are more similar (in some measurable sense) to each other than to those in other clusters [71, 95, 176, 184]. The data objects to be clustered could be textual documents, where the similarity is measured based on common terms in the documents. Equally the data objects could represent vertices in a graph structure, where by similarity is measured based on (say) distance between vertices using links.

There are two main scenarios in which clustering can be performed, either in a (1) static context or (2) dynamic context. In a static context clustering is performed to all data items at once. This is also known as the "offline method" of clustering objects, where by knowledge of all data is known at the time of cluster assignment.

In a dynamic context clustering is performed to data objects in a step-by-step manner. This method is also known as "online clustering", data objects are assigned to clusters one at a time, using only previously encountered data. An online method of clustering should be able to provide an approximation of the clusters at any point during its operation, and should be able to incrementally incorporate new data objects into the existing clustering at any time. In an online clustering method the order in which data objects are seen can significantly effect the outcome of the clustering [164].

Iterative methods can exist with respect to both the static and dynamic contexts. The distinction is made between having access to all the data when making decisions on iterative cluster assignments (static), or only partial data (dynamic).

2.6.1 Clustering Algorithms

Clustering algorithms can be broadly categorised based on the model in which they use to group data objects. The main clustering algorithm categories considered in this research are:

- Hierarchical based
- Centroid based
- Density based

Hierarchical based clustering algorithms use an assumption that data objects, that are closer (using euclidean distance for example), are typically more likely to be related than objects far apart. Hierarchical clustering approaches either start with each data items being contained in a set of single clusters to which a merging process is applied (agglomerative, or bottom up, hierarchical clustering) or the data items can be all contained in a single cluster to which a splitting process is applied (divisive, or top down, hierarchical clustering). Regardless of the approach adopted a dendogram can be used to illustrate the merging or splitting process. An example of a hierarchical (graph) clustering method is the Newman approach [144], which has been used for detecting community structure in web networks [143].

A centroid based clustering algorithm uses a central vector as a prototype to represent each of the clusters. This centroid can be assigned using the value of an existing data object, or can be of an arbitrary value. If given a value of k for the number of clusters that exist in the data, an optimisation problem is defined, where by each data object is to be assigned to one of the k clusters according to some distance function. An example of a centroid based clustering algorithm is kmeans [134].

Density based clustering algorithm defines clusters to be areas of higher density within data object collections. Density based on methods are founded on the idea of connecting points that satisfy a certain density criterion. Clusters can be represented by arbitrary shapes, in contrast to, say, centroid based methods like kmeans. Example of density based clustering algorithms are DBScan [76] and KNN [71, 95].

```
Algorithm kmeans ($k$)
1: Assign initial values for centroids $c_1$, $c_2$, $\ldots c_k$;
2: repeat
3:     Assign each data object to cluster with closest centroid;
4:     Calculate new centroid for each cluster;
5: until Convergence criteria is met;
```

Table 2.1: Pseudo code for kmeans algorithm based on [71].

The algorithms that are considered in this thesis are presented in further detail below. It will be noted that significance will be given to the kmeans algorithm which is used for the majority of the analysis in the work presented later in this thesis.

2.6.2 Kmeans

The well known [71, 95, 176, 184] kmeans clustering algorithm [134] is a method of partitioning objects into a set of k clusters. Table 2.1 presents some pseudo code for the kmeans algorithm. The initial step is to assign values for the k cluster centroids, this can be done arbitrarily, or using data objects as a guide. The algorithm then proceeds to assign each data object to the closest cluster; cluster centroids are recalculated as the process proceeds. This process is repeated until some convergence criteria is met, this could be that no change in cluster centroids is observed after recalculation.

The result produced from the algorithm is that each data object is clustered with similar objects, while a low similarity remains between clusters of objects. The algorithm has been used successfully in many applications [71, 95, 176, 184]. Despite the algorithms success, it is sensitive to the starting conditions, such as initial clustering and ordering of input [31].

The kmeans algorithm is traditionally used on a collection of data apriori (in a static context), such that all objects are known in advance. The kmeans algorithm used with respect to the work described in this thesis follows [149, 165], and uses a variation of the kmeans algorithm in an incremental setting (dynamic context) as well as a traditional implementation (static context). In the dynamic setting the data is clustered as it is received (as a stream). As per the original algorithm, the number of k clusters is required initially. To initialise the k cluster centroids method the first k data objects received are used. The cluster centroid are assigned based on the order of objects from the input stream. Applying this to the web using a web crawler output as a data stream means that the initial seeds are not selected randomly from the entire data, but are selected from initially crawled web pages.

ICA The so called Incremental Clustering Algorithm (ICA) presented in this section is based on the work proposed in [132]. The algorithm was originally proposed for the

```
1: $C \leftarrow$ Empty Set {cluster set}
2: for each document $d$ do
3:   for each cluster $c$ do
4:     Simulate adding $d$ to $c$
5:     $CASA_c new = CASA_c$
6:   end for
7:   ADD $d$ to $c$ with lowest $CASA_c new$
8: end for
```

Table 2.2: Pseudo code for ICA in the context of this work, based on the method in [132].

incremental clustering of search results. It is based on the Clusters Average Similarity Area (CASA). For a cluster c containing documents such that $c = \{d_1, d_2 \ldots d_n\}$, the pair-wise similarity for two documents d_i and d_j is measured as s_{ij}. The CASA value is therefore calculated as:

$$CASA_c = \frac{2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} s_{ij}^2}{n(n-1)} \tag{2.1}$$

The algorithm assigns new data items to clusters based on the lowest calculated CASA value, by simulating adding the new item to each of the clusters. Alternatively if the CASA value is below a certain threshold, the item forms a new cluster. With respect to the work described in this thesis, the algorithm is used as a binary clustering method to segment a graph as it is dynamically traversed (see section 7.4.2). The pseudo code used in this work is shown in table 2.2.

Bisecting kmeans The *bisecting k-means* clustering algorithm is a partitional clustering algorithm that works by computing a user specified k number of clusters as a sequence of repeated bisections of the vector space. A k-way partitioning via repeated bisections is obtained by recursively computing 2-way clusterings. At each stage one cluster is selected and a bisection is made [189].

KNN The *k-nearest neighbour algorithm* is an iterative agglomerate clustering algorithm [71]. Items are iteratively merged into existing clusters that are "closest", within a user specified threshold value. If items exceed the threshold, a new cluster is created. The algorithm has the ability to find arbitrary shaped clusters in the vector space. The principle of the algorithm is to group a set of items based on a majority vote, items are assigned to a cluster most common within its neighbours, hence arbitrary shaped clusters can be found. This is in contrast to the kmeans algorithm, which typically determine spherical shaped clusters hindered by data items assignment to the nearest mean.

DBSCAN The *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)* algorithm creates clusters that have a small size and density [76]. Density is defined as the number of points within a certain distance of one another. Note that the number of clusters, k, is not prescribed, but it is determined by the algorithm.

2.7 Summary

This chapter has presented the background and previous work with respect to the research presented in this thesis. The WWW was described in relation to the WBD problem. The KDD process model was presented which was then followed by background information and challenges related specifically to mining new knowledge from the web. The basic processes of a web crawler was explained in relation to random crawls of the web. The final section presented the clustering approaches used with respect to the work described later in this thesis.

Chapter 3

The Website Boundary Detection Problem

This chapter explains in more detail the WBD problem and the associated issues and challenges that will be approached in this thesis. This chapter first, in Section 3.1, gives a discussion on the ambiguous nature of current definitions of a website. Subsequently, in Section 3.2, a proposed definition of a website, in the context of the work described in this thesis, is derived. This definition is then used throughout the remainder of this thesis. The WBD problem is then described in Section 3.3 in terms of the issues and challenges encountered when attempting to derive a solution to this problem. The chapter is concluded with a brief summary in Section 3.5.

3.1 Website Definition

The term *w*ebsite is commonly used to describe a collection of web pages on the WWW. This term and its usage are understood by many users of the WWW, but despite its common usage, the term has no formal definition. The term website is therefore commonly defined according to each case in which it is used (either philosophically or practically). The combination of no formal definition, and the various applications in which it is used, makes the term website very ambiguous.

There is even some variation in the notation of the actual term "website" itself. Some variations include WWW-site, web-site and web site. Each term refers to the same concept never the less. As the usage and popularity of the Internet and corresponding terminology increased, the term 'website' (one word) was proposed in [85]. In the remainder of this thesis, the term *w*ebsite is therefore used.

As already noted, there is currently (at time of writing) no precise and consistent formal definition of what a website is, which means that a universally used notion of a website does not exist [22]. The World Wide Web Consortium (w3c), an international community whose objectives include the development of web standards and best practises, has proposed various definitions of what a website is since the communities inception. This work is presented in Sub-section 3.1.1 below. In the following two sub-sections (Sub-section 3.1.2 and 3.1.3) further discussion of the existing website definitions are given. These have been categorised according to whether they can be considered to be theoretical or practical definitions of a website (the theoretical definitions are divided further into dictionary and public definitions). This section is concluded, in Sub-section 3.1.4, with a discussion of the three level model of a website.

3.1.1 W3C Definitions

A number of formal definitions of a website have been developed by a variety of w3c working groups (`www.w3c.org/Consortium/activities`). Two definitions of noteworthy mention are those that were produced by the Characterisation Activity working group and the Device Independence working group.

The main aim of the Characterisation Activity working group was to establish, more precisely, the semantics for common web concepts. The group acknowledged the fact that preceding the groups' formation, the web had been in existence for some time without consistent definitions of web concepts that were part of the common vernacular such as 'website' or 'web page' [119]. The group first proposed a draft "terminology sheet" in 1998 [154], which then later lead to a public version of a Terminology and Definitions Sheet in 1999 [119]. This document conveyed the groups combined efforts at defining some key web concepts, in particular a website was described as

> "A collection of interlinked Web pages, including a host page, residing at the same network location" [119].

The documentation produced by the Characterisation Activity working group made a key point in that it acknowledged the author and or publisher of a website as being an important aspect of what a website is:

> "A person or corporate body that is the primary claimant to the rewards or benefits resulting from usage of the Web site, incurs at least part of the costs necessary to produce and distribute the site, and exercises editorial control over the finished form of the Web site and its content" [119].

The Web Characterisation Activity group came to a close shortly after the Terminology and Definitions Sheet was completed in Nov 1999 [153].

The Device Independence working group was set up in 2001, the aim of this group was to address the principle factors that effect the usage of the web by an increasing number of different devices (mobile phones, pda's and so on) [157]. The group produced a "working draft" of a glossary containing terms which were used in the groups main activities [122], this document was last updated in 2005 [123]. In the glossary the use of the term "unit" is significant with respect to the definitions proposed. The term unit was used to refer to a sense of creation of material on the web and also in the viewing of material by a user agent. An "Authored Unit" is described as being:

> "Some set of material created as a single entity by an author. Examples include a collection of markup, a style sheet, and a media resource, such as an image or audio clip" [123].

The creation of the content of a unit by an author or publisher can be considered as a consistent notion used in previous terminology when referring to the creation of related content for a website. The only difference in this description is that the content is consider as a related collection, and deemed as a single unit of information. In relation to the combination of such units for the purpose of presentation, an interesting term is that of a "Perceivable Unit", which is described as being

> "A set of material which, when rendered by a user agent, may be perceived by a user and with which interaction may be possible. User agents may choose to render some or all of the material they receive in a delivery unit as a single perceivable unit or as multiple perceivable units" [123].

The term can be described as a view over some set of individual units, that is customised for a certain individual's need, in relation to the relationship the units have. This implies that some set of web content has distinct relationships that are defined in their entirety by the author or publisher which can be directly interpreted.

3.1.2 Theoretical Definitions

The term website has been used in a theoretical sense to give an understandable description aimed at general users of the WWW. A selection of definitions from various sources aimed at a general audience are discussed in this section. The aim is to give an overview of what is thought to be a website, and highlight interesting characteristics from the selected sources reviewed.

The term *theoretical* is used here because the definitions are, in some sense, a discussion of ideas which are applied in the abstract. The theoretical ideas that are discussed need not be derived from practically applied settings, but rather from contemplation or theory. The practical application of these definitions are commonly a secondary concern (which is subsequently discussed in section 3.1.3), the first concern is to describe the characteristics of a website to the reader. In some sense to give a broad overview of what the term website means, rather than how it can be applied. The understanding of the concept is not to the level whereby the ambiguities of practical application are removed, but to educate about the existence of the concept. Tables 3.1 and 3.2 present a selection of definitions which are aimed at educating a general audience as to the meaning of the term website.

Table 3.1 lists definitions obtained from a selection of dictionaries, which can be thought of as being "established" definitions in the sense that they are supported by some sort of authority (the compilers of the dictionary), and thus these definitions can be said to be "trustworthy" definitions. This table will be referred to as the table containing "dictionary" definitions. An established source may have a more confined set of constraints when developing the definition. These constraints are in place for obvious reasons, and need to be backed up with evidence, not just opinion. The definition needs to be as comprehensive as possible, it also needs to be easily understood and applied by the target audience.

Table 3.2 includes definitions from various other online sources, these may contrast with the dictionary definitions in that they are not supported by some formal process as in the case of dictionary definitions. We refer to this second set of definitions as being public definitions in the sense that they are also publically available but are non-dictionary definitions. This table will be referred to as the table containing "public" definitions. A non-established source does not have to adhere to the same constraints as in the case of dictionary definitions, and thus can perhaps be seen as being more freely derived definitions. The ideas supporting the definitions can be subjective, and do not have to cover a wide variety of applications. Another difference can be that matters of opinion can be included, and therefore the derived definition can include, in some sense, notions of current sub-culture.

Considering the dictionary definitions (table 3.1) and the public definitions (table 3.2) in more detail it can be seen that they both convey some common ideas as follows:

Connected Web pages from the website are connected via hyperlinks [100, 32, 1, 59, 70]. This implies that some form of traversal can be used to reach all pages of the website.

Related Web pages from the same website are related in some way, hence they have some sense of relationship.

Home Page A single web page, the home page, acts as an entry point to the website, and can be used as the main point of access [100, 32, 1, 2, 70].

Single Entity The web pages from a website represent a single authoring body, or publisher [32, 1, 59, 120], where some specific information on a subject or topic is presented [100, 32, 2, 70]. This can be company/organisation/person/topic.

The notion of single entity is an important point to highlight in the definitions shown. It is a notion that is not consistent amongst all definitions [4, 61, 3, 136]. An entity is described as having a distinct and independent existence [7, 173, 55], it is a set containing sub sets that represent an abstract object [7]. A single entity is distinguishable from other entities [138], and is a uniquely identifiable element in a world [47].

Table 3.1: A list of theoretical dictionary definitions of the term “Website” (or equivalent, as discussed in section 3.1).

Ref	**Definition**
Dictionary of the Internet [100]	“This is the term used for a set of linked themed pages which are stored on a web server. Such a server can host one Web site or, in the case of an Internet service provider, can host a large number of sites. Access to a Web site is usually via a home page.”
The Canadian Oxford Dictionary [32]	“[A] hypertext document or a set of linked documents, usually associated with a particular person, organization, or topic, that is held on a computer system and can be accessed via the World Wide Web.”
The American Heritage Dictionary [1]	“A set of interconnected webpages, usually including a homepage, generally located on the same server, and prepared and maintained as a collection of information by a person, group, or organization.”
The Concise Oxford English Dictionary [174]	“[A] location connected to the Internet that maintains one or more web pages.”
A Dictionary of Computing [59]	“A collection of hyperlinked web pages owned by an individual, organization, or company.”
Collins English Dictionary [2]	“[A] group of connected pages on the World Wide Web containing information on a particular subject”
A Dictionary of Business and Management [120]	“Content accessible on the World Wide Web created by a particular organization or individual. The location and identity of a website is indicated by its web address (uniform resource locator). It may be stored on a single server in a single location, or on a cluster of servers.”
A Dictionary of Marketing [70]	“A location on the World Wide Web where specific information is grouped together in a collection of pages. A website has a main home page and the user clicks on links from this to other pages on the site.”
Princeton University WordNet 3.0 [4]	“[A] computer connected to the internet that maintains a series of web pages on the World Wide Web”
Free OnLine Dictionary Of Computing [61]	“Any computer on the Internet running a World-Wide Web server process. A particular website is usually identified by the hostname part of a URL. Multiple hostnames may actually map to the same computer in which case they are known as virtual servers.”
Dictionary.com [5]	“[A] connected group of pages on the World Wide Web regarded as a single entity, usually maintained by one person or organization and devoted to a single topic or several closely related topics.”

Table 3.2: A list of public definitions of the term “Website” (or equivalent, as discussed in section 3.1).

Ref	**Definition**
Wikipedia [3]	“A website (or web site) is a collection of related web pages, images, videos or other digital assets that are hosted on one web server, usually accessible via the Internet.”
Creative Retirement [136]	“A collection of World Wide Web documents managed by a single entity that provides information such as text, graphics and audio files to users, as well as connections called hyperlinks to other Web pages.”
Social by Social [16]	“Websites and webpages are the building block of the internet, constructed from HTML to present text, images and rich media within a browser”
Sound marketing concepts [17]	“Websites with any level of functionality. This can include e-commerce, Intranet or Content Management.”
Red Hat [12]	“A collection of webpages, files, and other resources published or aggregated by one source.”
Cardinal Business Group [9]	“A visual interpretation of information by a web browser in a user-friendly manner.”
WebSiteSlug [19]	“[A] place on the internet where someone has posted some kind of information. A website can be made up of one or more pages.”
WolfWebDesign [20]	“A collection of web pages or WWW files starting with an index page or home page. The larger website are often hosted on multiple servers in various geographic locations.”
California performance [10]	“A website is a collection of network services, primarily HTML documents, that are linked together and that exist on the Web at a particular server. Exploring a website usually begins with the home page, which may lead you to more information about that site. A single server may support multiple websites.”
University of Minnesota [11]	“A website is a related group of web pages published on the World Wide Web.”
National Archives of Australia [15]	“A collection of electronic files, usually under common administrative control, linked together and made accessible via the internet.”
JIRA Studio [6]	“A website is a collection of related web resources, usually as grouped by some common addressing as when all resources on a single host, or group of related hosts, are considered a website.”
Delft University [14]	“[A] location on the (World Wide) Web usually containing a collection of hyperlinked documents and files.”
IT Connections [13]	“Collections of web pages linked together as a whole.”
Unified Process for EDUcation [18]	“A Web system that is all on one server.”
about the web [8]	“A group of similar web pages linked by hyperlinks and managed by a single company, organization, or individual”

Certain analogies between a website and an entity can be made. A website exists independently of other websites, it can be uniquely identified using a set of uniform resource identifier's (URI's) and is distinguishable in the WWW. A website is a set (or collection) of interlinking pages, these interlinking pages can be further grouped into sub sets, and as an entire collection represent an abstract object. This abstract object is that of a website. The reason a website is considered an abstract object is because it is not explicitly described in the web, it does not necessarily have a concrete existence. Thus a website can be argued to be a notion, or concept, that is thought of, but is in essence a notion that is abstract from the physical description of the web. Thus the notion of a website does not have to have an explicit physical encoding in the web itself. Despite this, the theoretical notion of a website cannot be disputed to exist.

A website as an entity has a further distinction that can be used to define and highlight its existence in the WWW, that is its purpose of creation. The website entity is created and focused towards a goal. This could be to serve an individual, organisation, company or even provide targeted information on a certain topic. This is supported in the definitions presented [136, 12, 15, 8, 32, 1, 59, 120, 5].

The final point that is common to all definitions presented in tables 3.1 and 3.2 is the fact that they are vague. The ideas and concepts that are used in the definitions can be ambiguous when applied to such a complex structure as the web. For example both sets of definitions imply that the set of pages that form a website can be located on a single web server [100, 1, 174, 120, 3, 10, 6, 18, 32, 1], or multiple web servers [100, 120, 20, 6]. Also, it could be the case that a single web server can host a multitude of websites [100, 120, 10, 61]. The point of how the information is hosted, and served to users is an ambiguous point which is common across the definitions.

The definitions presented in the tables 3.1 and 3.2 provide several theoretical definition of what a website is. They serve to express the notion of the existence of a website, and some fundamental aspects that contribute to a collection of pages that could possibly be deemed to be a website (for example being connected, having a home page, being related in some way, and so on). It is argued that the definitions shown in the tables provide an indication of what is representative of the common concepts and notions, in the theoretical sense, of the term website. As a result of several theoretical definitions, it can be seen how the nature of defining a website is highly subjective, and can vary greatly between sources while at the same time being aimed at the same target audience. This subjective nature can lead to an open interpretation of what a website is, if applied in a practical context.

Inspection of the definitions indicates that vagueness exists, which subsequently means that we can conclude that there is no universally accepted term. Each definition of the term website can include varying concepts, but can each still be argued as an equally viable definition of a website. As a result these ambiguities make it difficult to choose and apply a definition in terms of a technique to define the boundaries of a website.

3.1.3 Practical Definitions

A number of proposals have been put forward over the years, in the academic literature, to characterise the idea of a collection of strongly related web-pages for use in a practical application. Of note are the definitions proposed by Senelleart [167, 168], Neilsen [146], Asano [28], Dmitriev [65] and Henzinger [98].

In the work by Senellart [167, 168], the aim was to find web pages that are contained in logically related groups using the link structure that might connect a set of web pages. Senellart emphasised the fact that there is no clear definition of what a website is, and defines a "logical" website as a collection of nodes that are significantly more connected than other "nodes". This definition abstracts from the physical notions described in the traditional definition (single server, single site) and makes a more subjective claim that concentrates on the similarity between pages.

In the work by Neilsen [146] the notion of a web *subsite* is defined. This is a collection of pages on the web that have shared navigation mechanisms, a consistent styling and a distinct entry point. Rodrigues [160] applied this subsite definition, although the notion of styling was disregarded. This was due to the applied technique using only the hyperlink structure of the web. The technique of detecting subsites uses the notions outlined by Nelisen; a common entry point and style of navigation between pages, both regarded as strong characteristics of a subsite [161].

In [28] the difficulty of the notion that web server equals website is acknowledged, and is further enforced by a claim of the vague nature of the current definitions of a website. The proposed solution given in [28] is to define a "directory based site". This is described as a section of a web server that an author has full control over. This is in contrast to a "one site server" that hosts only a single website. The aim of the work described in [28] is to detect the boundaries of a directory based website; good results were reported with respect to some data sets, however the proposed approach does not scale well because of the uniform resource locator URL filtering technique adopted.

Research by Eiron [73] proposed the notion of *compound documents*. This is a set of web pages (web-documents) that can be aggregated to a single coherent information entity. An example of a compound document is a news article published to the web. The article may be displayed over several pages within a news website, each page with a unique URI (Uniform Resource Identifier). Thus each page can be seen to remain a separate entity in its own right. The intention of the author of the article however is for the reader to absorb the article as a single piece of information [65]. The aggregation of the web documents making up the article is known as a compound document. A compound document is described as having an entry point (which can be non trivial to identify) which is similar to the definition of a subsite presented above. Note that this definition challenges the synonymous notion of web node equals web page.

The work by Henzinger [98] makes a simple assumption of the definition of a website according to the URL associated with a web page; namely that the host domain included in the URL should be used as the primary indicator of a website. Thus all pages under the domain www.liv.ac.uk (including variants like www.csc.liv.ac.uk and www.liv.ac.uk/chemisty) would be deemed as a single website.

3.1.4 The Three Notion Model of a Website

This section summaries the investigation into the previous existing theoretical and practical definitions of what a website is. From the foregoing it can be observed that the term website is ambiguous in its nature, this fact does not help the desire for a universal formal definition. In [137] it is argued that the term website has three distinct levels of derivation:

1. **Intuitive notion**: Describing a website in terms of a set of related resources, sharing a common creating entity.
2. **Network notion**: Consider a website in terms of a host or web server; the machine serving the content of the site.
3. **Topological notion**: Definition of a website in terms of some naming conventions (e.g domain or sub domain).

The intuitive notion describes an entity as being responsible for the creation of the resources of a website. The idea of an entity could describe an individual person, an organisation, a department within an organisation, or even a project with its own site. The issue is not that an entity is not flexible enough to encompass each of the possible descriptions, but the fact of consistency when applied to the web can be a problem. The content published on the web can be described by a range of applications as defined by the entity responsible. Examples include:

1. Single website representing an organisation.
2. A collection of department websites within an organisation.
3. A collection of department projects within a department of an organisation.
4. A collection of individual staff's web pages within an organisation that work in a department on various projects.

The network level states that a host or web server that provides the content defines the website. Although this is true in some cases, it is not universally applicable to all websites. A single machine can host several websites, also a single site can be hosted on several machines. In general:

- A website can be comprised of many hosts. For example www.intel.com and support.intel.com can be considered to collectively constitute the Intel website [41] [168].
- A single host can comprise many websites. For example members.aol.com hosts multiple websites published by many individuals [41, 168].
- The notion of a web server or host can become unclear when referring to distributed hosts, mirrored servers or virtual hosting [168].

The topological level defines a website in terms of the components of the domain name. Defining a website in accordance to this topological fashion can be applied correctly for certain websites, but is not universal. The problem in this case is that there is no strong naming convention that can be adopted on a universal scale. A website can be located using three main components of a domain name:

1. domain level, example.org
2. sub-domain level, zone.example.org
3. directory level, example.org/blog

Despite the distinctions between the three levels two common problems arise (also highlighted by the definitions presented in the previous section). The first is that each of the individual levels can cause confusion when applied to the web environment, and the second is that the definitions can become blurred.

The possible advantage of providing flexibility in application of the levels to the web environment can lead to further confusion concerning the three notions which would otherwise produce a formal definition of a website. The majority of the definitions of a website blur the distinction between each of these terms, and provide an idea of a website that includes constraints from each of the levels of derivations.

3.2 Proposed Definition

The existing body of website definitions are ambiguous in the sense that they are open to multiple interpretations. It is suggested, in the context of boundary detection, that an appropriate definition must encompass several of the above concepts while at the same time reducing the vague nature of a definition which can hinder its practical application with respect to the website boundary detection problem.

The proposed definition is founded on the idea that a website must fulfil the following three features (labelled WS1, WS2 and WS3):

WS1 Have a common entry point, referred to as the website *home-page*, such that every page in the collection is reachable from this home-page through a sequence of directed hyperlinks;

WS2 Have distinct navigation or styling features; and

WS3 Have a focused content.

Thus a website is a collection of web pages that fulfils these three central features (WS1, WS2 and WS3). The first two of the above are syntactic in nature. They refer to the clearly recognisable aspects of a given collection of web-pages. The third one is intended to capture the purposes of the creators of the given collection.

Considering the above definition in further detail it should be noted that the definition is couched in terms of the expected structure of a website, and that some of the elements of the definition build upon existing ideas found in the literature. WS1 is probably the most obvious one, and its importance has been recognised previously (see for instance [113, 125, 146, 187]). It is also natural to add WS2; similar styling is a clear sign of authorship. Collections of web pages that have the same styling tend to have been created by the same people. Minor differences may arise between pages in the same collection, however common themes will often be shared by all pages that are part of a single conceptual unit. WS2 also encompasses the possibility that many pages in the same site may have similar link patterns. The styling may be completely different, but the navigation of the pages may share some common links (for instance a back link to the website home-page). Considering WS3, the idea of focused content which is implicitly present in other proposals (see for example [28, 48]). The idea reflects the situation where an author has control over a collection of pages so that the pages can thus be said to be related by the author's intentions with respect the dissemination of content on a certain topic(s).

The above definition (in the context of boundary detection) offers a number of advantages:

Generality: It is more general than previous proposals. Constraint **WS1** clearly relates to the notion of *seed pages* that has been used in the past as a means of clustering content-related web pages. **WS2** encompasses the approaches based on the study of the URLs and the link structure of a given set of pages.

Precise: The definition adds an element of preciseness to the otherwise vague definitions. It is argued that any sensible definition must contain a semantic element referring to the characteristics present that constitutes a web pages belonging to a website. Adding such element to the definition (constraint **WS3**) makes it suitable to describing a wide range of boundary detection scenarios.

Effectiveness: The proposed definition is effective because, as will be demonstrated, it can be used to identify web site boundaries using data mining techniques.

The degree of membership of a web page to a website is an important notion with respect to the proposed definition and its application in the remainder of this thesis. A single web page must belong to a website. This could simply be a website hosting information on a topic or service offered, which is published on the web as a single page. A web page can only belong to a single website. There is no fuzzy relationship with respect to a page belonging to several websites. As presented in previous work, there are many levels of structural relationship that can be defined in the web. For example, the pages from a website could in fact be grouped to reveal subsite relationships with a main website. In this work however, no such hierarchical segmentation of pages in a website is defined.

3.3 WBD problem

The nature of the web is such that information of almost any type can exist, this information can be connected in a multitude of ways, thus serving to demonstrate that the web is a complex interconnected web structure of great diversity. However, there are intuitive notions that are used to add meaning to the complex interconnected structure of the web. The notion of a website is one of these. This common term is used to describe a certain type of relationship that information can have on the web. Unfortunately due to the nature of the web the information that is contained within the boundary of a website is not explicitly described. This fact can hinder the definition of this commonly used term, as there is no description of what should be included or excluded from a particular website. Hence, the website boundary detection problem.

In this work, the WBD problem is defined as follows:

The problem of identifying all web pages/resources/media that are part of a single website

It is argued that in a broader context the WBD problem can be approached in two particular ways, either; (i) Philosophically or (ii) Practically. In the philosophical approach to the WBD problem the focus is on the meaning of the term website, its relationships and collaborative interactions. This has already been discussed extensively earlier in this chapter. In contrast to the philosophical debate, in this thesis a practical approach to the WBD problem is taken. Despite appearing in the information retrieval and machine learning literature [41, 98, 104] there has been no clear outline of the problems associated with the implementation of practical WBD solutions. Any practical WBD solution must consider:

1. How to define a website.

2. How to practically detect the boundaries of such a website.

The first problem has already been addressed earlier in this chapter (see section 3.2). The second problem is addressed in the following two sub-sections 3.3.1 and 3.3.2. Sub-section 3.3.1 outlines issues related to the practical implementation of a WBD solution. Sub-section 3.3.2 outlines an approach to the WBD problem with respect to the KDD process.

3.3.1 WBD Mining Issues

In this section the issues related to general web mining, as discussed in chapter 2.4.1, will be considered in terms of the WBD problem.

Volume: The amount of data on the web is large and increasing every second. The use of subsets or small parts of the whole web structure is an essential factor in the success of reducing the potential complexity of the WBD problem. In each case any assumptions or limitations that have been used to scope the WBD problem, in the context of the work described in this thesis, are highlighted.

Diversity: The web contains data in many formats, this makes it extremely diverse. To avoid costly resource hungry techniques and processes to deal with this diverse data, this research has been limited to WWW pages expressed using only HTML/XHTML/XML. These formats were selected because they are the most commonly used formats. This work will not attempt to integrate other proprietary formats, or deal with parsing or feature extraction associated with video or multi-media type data.

Semi-Structured data: The retrieval of information from a semi-structured data source can prove to be extremely challenging. The research described in this thesis calls upon some standard HTML parsing libraries to help extract relevant attributes from the acquired web data. The strategies and steps involved in the features extraction are given in more detail in chapter 5.

Authority: This issue does not directly concern the work in this thesis. The only assumptions made are that: (1) the data gathered is provided by the host as required, and is not adversely affected directly by spam content, and (2) that the content to be analysed is not malicious in any way.

Noise: An inevitable aspect of KDD is that some amount of noise data will be acquired, that must be processed accordingly. This is especially true when dealing with data from the web. This aspect is directly addressed in each of the techniques that are described in this thesis. A successful WBD solution will distinguish noise data from the target data sufficiently well, which in turn will help to achieving a highly accurate WBD.

Dynamic and Distributed: The dynamic and distributed nature of the web can make it a problematic data source for many time critical applications. To avoid the issues raised by the ever changing location and content of the data on the web this research uses snapshots of web content. It is common to limit a web crawl by domain or sub domain to some degree; however such a method would stifle the gathering of content that is distributed over various domains, but is actually part of the same website. This is the main reason the crawls of the web are unrestricted when creating RDG data sets (section 4.3.2). To overcome the distributed nature of the web, when data is gathered the web crawl is not limited by URL or domain or sub domain.

Virtual Society: The interactions of users of the web are considered as very important in the research presented in this thesis with respect to the proposed website definition. The key aspect of the research in this thesis is to exploit the user's intention which is encoded in to related web documents of the same website. This aspect is reflected in the proposed definition that was given in section 3.2.

Persistence: The ever changing nature of the web is addressed as described above under "Dynamic and Distributed". A snapshot of the web is used to achieve the desired website boundary. The assumption is made that this data remains fixed for the duration of the evaluation of the approaches presented in this thesis. To use "live" web data would cause issues if content changed during the WBD resolution process. These issues are well documented and is an active area of research [121, 108, 180, 91, 177].

Deep web: In this research only the surface web is considered when creating the data snapshots. The deep web is used to describe content that is not directly accessible using a normal web browsing pattern. Access to this content may require forms to be complete or other interactions that are far beyond the scope of a standard web crawler, and consequently this research.

3.3.2 WBD solutions derived using The KDD process model

The WBD problem shares many similarities with the general Knowledge Discovery in Databases (KDD) problem, which is important with respect to the research discussed in this thesis. In chapter 2.3 the KDD problem was presented in terms of a 6 step process. The WBD problem can also be described using this 6 step process as shown in in Figure 3.1. This sub-section discusses the WBD problem with respect to the process presented in Figure 3.1.

It is important to note that as in the general KDD process model, the proposed WBD process model can also be an iterative process. This means the steps can be repeated. This is an important point with respect to the static and dynamic approaches

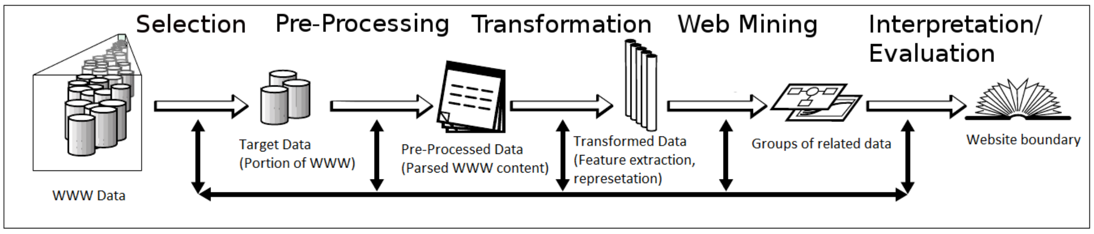

Selection
Pre-Processing
Transformation
Web Mining
Interpretation/ Evaluation
WWW Data
Target Data (Portion of WWW)
Pre-Processed Data (Parsed WWW content)
Transformed Data (Feature extraction, represetation)
Groups of related data
Website boundary

Figure 3.1: The WBD problem as modelled using the KDD process based on the Process-Centred view model by [77]

to WBD considered in the later chapters of this thesis (static in chapter 5 and dynamic in chapter 7).

The task of identifying the complete boundary of a target website commences with some seed page(s) from the website of interest. Typically this is the home page or entry page. This step is related to understanding the selection criteria for a WBD problem given such a seed page. The WBD solution generation process then continues as follows:

1. Selection: A proposed WBD approach commences by obtaining a collection ("snapshot") containing web pages by crawling a portion of the web from the given seed page. An ideal WBD approach would gather a snapshot containing only pages from the target website, thus producing a website boundary solution. However, in practice, a "snapshot" is created comprising of both "target" and unavoidable "noise" pages.

In this thesis, two main approaches are used, static and dynamic. In the static context, a snapshot needs to be gathered before any website boundaries can be detected, while in the dynamic context, this is done as the crawl proceeds. The web crawl aims to do two things: (1) gather all target content from the website, and (2) gather as little noise content as possible. The crawl must be wide and deep enough so as to cover at least the target website pages. There are two main techniques that are considered in this thesis:

1. **Semi-Automated:** This is a web crawling process guided by some initial input from a user. The process is conducted using some limit to control the web crawling. This process is used in chapters 5 and 6.

2. **Automated:** A fully automated crawl produces a snapshot of the web based on some heuristic approach. The crawling process is revised as the crawl progresses. This process is used in chapter 7.

The web crawling adopted in this work uses both the automated and semi-automated approaches. The static work in chapter 5 uses a semi-automated web crawl, which requires a user depth input before the data acquisition begins. The dynamic work in chapter 7 uses an automated random web crawling technique to acquire the data from the web.

2. Pre Processing: During the pre-processing stage the aim is to make sure the web page content is valid; in other words that the content comprises a well formed document structure suitable for parsing and information extraction. A well formed document has no missing tags, and includes elements that support feature extraction. If pages are not well formed (which is often the case with data from the web) the web pages are "cleaned" and output in a suitable format for feature extraction. This is done using a library that can handle ill-formed web content, and output corresponding valid content. Examples of such libraries include HTML parser[1] and HTML Tidy[2].

3. Transformation: Using the pre processed web content which has been validated in the previous step, information extraction can take place. The features that are considered in this work are considered in section 5.2, and are modelled using the vector space model (see section 3.4).

4. Data Mining: The data mining step is the core element of the entire proposed process. The clustering algorithms that are considered in this work were detailed in the chapter 2.6.1. The clustering algorithms are used to produce clusters of related pages. The relationship of the pages in the same cluster as the seed page, is deemed to represent the target website. The remaining clusters are then identified as the noise cluster. The target cluster reflects the relationship that is encoded in the content of the website, which in turn is a reflection of the definition of what a website is as derived in this thesis.

5. Interpretation/Evaluation: The website boundary patterns that are discovered in the previous step are interpreted as a solution to the WBD problem. This may be evaluated against labels representing the ground truth of the specific WBD problem. Details on how solutions maybe evaluated, in an experimental setting, are given in section 3.4.3.

6. Knowledge: The discovered website boundaries can be used as new knowledge about a particular WBD problem. This new knowledge can be used as a basis for the applications presented in section 1.1.

[1]HTML parser: `http://htmlparser.sourceforge.net/`
[2]HTML Tidy: `http://tidy.sourceforge.net/`

At this point it is important to highlight the fact that the entire process can be executed non-sequentially. This is referred to as an alternative selection method. Recall in the explanation of the KDD process that it was acknowledged that the process could be iterative, and did not have to be sequentially executed. Steps; 2. Selection, 3. Pre processing, 4. transformation and 5. Data mining can all be performed either sequentially (as in the case of the proposed static solution to the WBD problem discussed in chapter 5) or iteratively (as in the case of the proposed dynamic solution to the WBD problem discussed in chapter 7).

3.4 Formal Description

The general WBD problem can be described using a formal description as follows. A web page was defined as a WWW resource written in HTML format (as opposed to a web resource written in, for example, pdf or postscript) and is denoted by w. Therefore a collection of web pages W, comprising n individual pages such that $W = \{w_1, w_2, \cdots, w_n\}$. There are various techniques for representing web page's according to their content and structure, the techniques used in this research are explained below.

Structure: The WBD techniques in this research will work on a portion of the WWW. This can be modelled using a graph $G = (W, E)$, where W is a collection of web pages (as noted above), and E is the set of directed (hyper) links between pairs of elements in W. Without loss of generality we assume that G is connected.

In the remainder of this thesis the terms page, web page, node, and vertex are used interchangeably, but in all cases we are referring to an element of W.

Content: Each page (w) has a numerical feature vector associated with it. Technically this can be determined once the page has been downloaded from the internet. There exists a function $f : W \to \mathbb{R}^k$, for some fixed integer $t \geq 1$, such that for any $w \in W$, $f(w)$ is an ordered sequence of t real numbers characterising the page w.

A common method to represent a set of numerical features, and the one used in this thesis, is the vector-space model [163]. Using a vector space model each web-page, w_i, is described in terms of a m dimensional feature vector, $V = \{v_1, v_2, \ldots, v_m\}$, which resides within some m dimensional feature (vector) space. Each dimension of the feature space v_i is described by some feature; more details on the features used is given in chapter 5.2. The sequence (set) of possible values for a feature (v_i) is then described by the set of values $\{\varphi_{i_1}, \varphi_{i_2}, \cdots, \varphi_{i_k}\}$. The value of k will depend on the nature of the feature; given a binary valued feature $k = 2$ the value set will be $\{0, 1\}$. Given (say) a numeric feature the value of k may be substantial. An example is shown in Figure 3.2.

A measure of similarity can be expressed between two web pages using the distance (d) between the associated feature vectors of each page (using content). The (Euclidean) distance is a measure of how related two web pages are: $d = 0$ indicates exactly the same page, while $d > 0$ indicates a relative value of similarity.

WBD Using the description of a web page, the general WBD problem can be formally defined as follows. Given a collection of web pages W comprising n individual pages such that $W = \{w_1, w_2, \cdots, w_n\}$ we wish to find the sub-set of W that describes a bounded set of web pages (ω), i.e. a web site (as defined in section 3.2), centred on some given seed page w_s (typically a known home page, or entry page). The set ω is said to be a group of similar web pages in relation to the seed(s) web pages. This effectively means distance d between pages in the set ω is minimal (close to $d = 0$).

$$VSM = \left[\begin{array}{c|cccccc} & v_1 & v_2 & v_3 & v_4 & \dots & v_m \\ \hline w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\ w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\ w_4 & 1 & 0 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ w_n & 1 & 0 & 0 & 1 & \dots & 1 \end{array}\right]$$

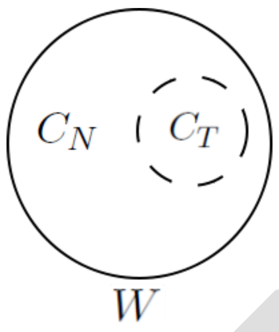
Figure 3.2: The vector space model for the WBD problem using some formal notation. In this example $k = 2$ for all features and thus a binary value of $\{0, 1\}$ is allocated to each feature.

3.4.1 WBD as a clustering problem

Using the formal notation outline above, the WBD problem can be shown to be a special type of clustering problem. Given a collection of pages W with a seed page w_s, there exists a bounded sub set of WWW pages ω. The set ω represents the collection of web pages making up the website of interest, which is effectively defined by $w_s \in \omega$. Note that W is a pre selected (data) set of WWW pages in this case (see chapter 4 for more details). Using a preselected set W means that each collection of web pages can have some associated labelling, which corresponds to exactly which pages are part of the website, and which are not. C_T is the number of pages labelled as contained within the website, thus essentially the cardinality of set $|\omega|$, these pages are known as *target* web pages. Similarly, C_N is the number of *noise* pages. See Figure 3.3 for an illustration. Intuitively the set $W = C_T + C_N$.

The web pages in W can be modelled using a vector space where each page is described as a feature vector (as explained above and illustrated in Figure 3.2). Using this vector space model any one of a number of clustering algorithms can be applied (see

Figure 3.3: An exmaple data set W, showing a composition of subsets C_N (noise) and C_T (target).



section 2.6.1). This clustering process essentially groups similar web pages together, based on the values in the associated feature space. When a clustering algorithm is applied the output is a set of clusters $K = \{K_1, K_2, \ldots, K_z\}$ where z is the total number of clusters produced. Thus a cluster configuration K is essentially the set of pages W divided into groups of related or similar pages as deemed by a particular clustering algorithm. For example, if the set K contains $z = 3$ clusters K_1, K_2 and K_3, each of these clusters would contain a disjoint sub set of pages from W. This is shown in Figure 3.4.

One of the clusters in the set K will contain w_s and is therefore identified as the target cluster K_T ($K_T \in K$). Cluster K_T will therefore contain pages most similar to w_s. In the three cluster example (Figure 3.4), K_1 might be defined as K_T as it contains w_s ($w_1 = w_s$).

The clustering produced can actually be further reduced to a binary clustering problem. The target cluster K_T is as defined above, while similarly the remaining clusters $K - 1$ (the set of clusters K minus the target cluster K_T) can be aggregated into a set K_N, containing what is deemed to be noise items. Again, using the three cluster example from above, if $K_T = \{K_1\}$ then $K_N = \{K_2, K_3\}$. Therefore the original clusters $K = \{K_1, K_2, K_3\}$ can be mapped into the set $K = \{K_T, K_N\}$ using w_s. Using the illustration in Figure 3.4, the transformation can be seen in Figure 3.5. Why we might wish to reduce the WBD clustering problem to a binary cluster representation. The reason is that given a set of web pages W there may be many clusters in the overall cluster configuration (see for example Figure 5.2 presented in chapter 5).

In the context of the WBD problem, the only concern are the pages inside a website, forming the website boundary. Thus, to eliminate the various other possible configurations during website boundary detection, a binary cluster transposition, using w_s, allows the detection of "web pages in the website" (target pages K_T) and "everything else" (noise pages K_N).

$$\left[\begin{array}{cc|cccccc}
 & & v_1 & v_2 & v_3 & v_4 & \dots & v_m \\
\hline
\multirow{2}{*}{K_1} & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\
 & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\
K_2 & w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\
\multirow{3}{*}{K_3} & w_4 & 1 & 0 & 1 & 0 & \dots & 0 \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
 & w_n & 1 & 0 & 0 & 1 & \dots & 1
\end{array}\right]$$

Figure 3.4: An example cluster configuration showing a grouping of web pages into $K = \{K_1, K_2, K_3\}$ clusters.

$$\left[\begin{array}{cc|cccccc}
 & & v_1 & v_2 & v_3 & v_4 & \dots & v_m \\
\hline
\multirow{2}{*}{K_T} & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\
 & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\
\multirow{4}{*}{K_N} & w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\
 & w_4 & 1 & 0 & 1 & 0 & \dots & 0 \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
 & w_n & 1 & 0 & 0 & 1 & \dots & 1
\end{array}\right]$$

Figure 3.5: An example cluster configuration showing a grouping of web pages into $K = \{K_T, K_N\}$ clusters.

To summarise, given a WBD problem in the form of a set of pre-labelled pages W and seed page w_s, by defining the problem as a special type of clustering problem, the pages in W can be labelled as belonging to the target set K_T or the noise set K_N. This set of clusters is a potential solution to the given WBD problem. This target cluster is then deemed to be the collection of pages that represents the website, which in turn also identifies the website's boundary. However, the quality of the website boundary discovered depends entirely on the composition of set K_T, as this is what makes up the boundary of the website. In subsequent chapters the evaluation measures to quantify the performance of a WBD solution will be discussed, see section 3.4.3.

3.4.2 Static vs Dynamic

In this sub-section the application of the WBD clustering paradigm described above will be discussed in terms of the static and dynamic contexts.

Static In the static context, the entire vector space is given at the start of the WBD solution generation process. Thus the collection of web pages $W = \{w_1, w_2, \cdots, w_n\}$ and the set of features for the web pages $V = \{v_1, v_2, \dots, v_m\}$ is known a priori. Recall that the applied clustering algorithm produces a set of clusters K, from which the sets K_T and K_N are extracted using all available data.

Dynamic In a dynamic context, the entire vector space is not known in advance. The web pages in the vector space are in fact populated based on some type of traversal of the web graph (see section 7.3). This essentially means that the vector space is updated at each time 'step'. More details on the nature of the traversals of the "web graph" are given in section 7.3.

Recall that the seed page w_s is given for a particular WBD problem. This seed page is used as the starting point for the traversal of W. Prior to the start $V = \{\}$, $K = \{\}$ and of course $VSM = []$. An example of a traversal of W in the dynamic context, is given in Figure 3.6a - 3.6f. At each step the VSM is updated with information from an additional page. Also notice the sets K_T and K_N on the left. Note that dimensionality m of the feature space is likely to increase with each step as new features and feature values are encountered.

As illustrated, in a dynamic context, during the early stages the vector space might contain none or only partial information on some sub set of the web pages of interest. This characteristic has implications on the feature space as defined by the formal notion explained above. Only a partial number of web pages are known; set $W = \{w_1, w_2, \cdots, w_n\}$ does not contain all items, and thus n, the total amount of web pages is unknown. Also, as shown in Figure 3.6, this has implications on the vector representation. The clustering algorithm has to make decisions based on what groups the web pages w_1 to w_n belong to using only information about the known pages available at the current step. This means the clustering method has to effectively react to additions and updates to the collection of web pages represented in the vector space. The complete set of WWW page feature values available in the static context, are not known in the dynamic context.

In chapter 5 the nature of web page attributes (features) are explored so as to discover the "best" performing feature set F with which to model web pages in the context of WBD. The focus of chapter 5 is to highlight the difference in the performance of various features. In chapter 7 the traversal of web pages is explored; the focus here is not on the features, but on the order of web page traversal to maximise WBD solution accuracy while minimising the number of noise pages visited.

3.4.3 Evaluation Metrics

The evaluation measures presented in this section are used later in this thesis to evaluate the effectiveness of proposed WBD solution mechanisms. A WBD solution is given as a partition of the set of web pages W into K groups (or clusters). From these K clusters a target cluster K_T is identified. The target cluster K_T is said to be the cluster that contains the given seed page. This seed page is used as it is the only known web page that is correctly labelled as being part of the WBD solution.

Each of the evaluation measures described below can be used to gauge the perfor-

Figure 3.6: The dynamic construction of a vector space created from n web pages (w_n). At each step (a-f) a new web page is added.

$$[\,]$$

(a) Step 0

$$\left[\begin{array}{ll|cccccc} & & v_1 & v_2 & v_3 & v_4 & \dots & v_{1m} \\ \hline K_T\{ & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \end{array}\right]$$

(b) Step 1

$$\left[\begin{array}{ll|cccccc} & & v_1 & v_2 & v_3 & v_4 & \dots & v_{2m} \\ \hline K_T\{ & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ K_N\{ & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \end{array}\right]$$

(c) Step 2

$$\left[\begin{array}{ll|cccccc} & & v_1 & v_2 & v_3 & v_4 & \dots & v_{3m} \\ \hline K_T\{ & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ K_N\Big\{ & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\ & w_3 & 0 & 1 & 1 & 1 & \dots & 1 \end{array}\right]$$

(d) Step 3

$$\left[\begin{array}{ll|cccccc} & & v_1 & v_2 & v_3 & v_4 & \dots & v_{4m} \\ \hline K_T\Big\{ & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\ K_N\Big\{ & w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\ & w_4 & 1 & 0 & 1 & 0 & \dots & 0 \end{array}\right]$$

(e) Step 4

$$\left[\begin{array}{ll|cccccc} & & v_1 & v_2 & v_3 & v_4 & \dots & v_{5m} \\ \hline K_T\Big\{ & w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ & w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\ K_N\Bigg\{ & w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\ & w_4 & 1 & 0 & 1 & 0 & \dots & 0 \\ & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ & w_n & 1 & 0 & 0 & 1 & \dots & 1 \end{array}\right]$$

(f) Step 5

mance of a WBD solution. Which is essentially measuring the degree to which the complete collection of pages that belongs inside a website boundary are contained in K_T, and the extent to which the complete set of noise pages are contained in K_N.

Using the above notation any WBD solution can be interpreted as a binary clustering problem. Using traditional terminology, the class (target) cluster is represented by K_T and the noise cluster is represented by K_N.

Thus:

m_{TT} The number of target pages in the target cluster.
m_{TN} The number of noise pages in the target cluster.

Similarly for K_N:

m_{NT} The number of target pages in the noise clusters.
m_{NN} The number of noise pages in the target cluster.

The evaluation metric above are also shown in the confusion matrix in Figure 3.7.

Figure 3.7: Evaluation metrics expressed as a confusion matrix.

		Pages	
		Target	Noise
Cluster	Target	m_{TT}	m_{TN}
	Noise	m_{NT}	m_{NN}

Accuracy. The accuracy is the measurement of how close the true value is to the value being measured [176]. The general formula to calculate the accuracy a_{ij} of cluster i with respect to class j is defined as:

$$a_{ij} = \frac{m_{ij} + \left(\sum_{i=1}^{K} m_i - m_{ij}\right)}{m} \tag{3.1}$$

Where m_{ij} is the number of objects of class j in cluster i, m_i is the number of objects in cluster i and m is the total number of objects in all clusters K.

Given the generalised formula of accuracy, we now consider the accuracy of cluster K_T. The accuracy is calculated as the sum of the correctly classified target class C_T web pages within K_T plus the sum of the number of "noise" web pages C_N correctly allocated to K_N divided by the total number of web pages, thus a binary class evaluation. More formally:

$$a_{K_T C_T} = \frac{m_{TT} + (|K_N| - m_{NT})}{|W|} \tag{3.2}$$

As previously noted, K_T and K_N are the target and noise clusters respectively. $|W|$ is the cardinality of the input set of all pages.

Entropy. Entropy is used to measure the degree to which each cluster or a cluster configuration consists of objects of a single class [176]. The general formula to calculate entropy of a cluster is $e_i = -\sum_{j=1}^{L} P_{ij} \log_2 P_{ij}$; where L is the number of classes, j represents the class and i the cluster being measured respectively (as above). Finally P_{ij} is the probability that a member of cluster i belongs to class j, this can be calculated as $P_{ij} = \frac{m_{ij}}{m_i}$ where m_i is the number of objects in cluster i and m_{ij} is the number of objects of class j in cluster i.

Given the general formula above, we now consider the entropy calculation of a single cluster K_T with two possible classes, a target class and a noise class. In such a case, the number of classes L equals 2, j refers to the target class, and i refers to the target cluster K_T. Therefore the entropy e_T is defined as:

$$e_{K_T} = 1 - \left[\left(\frac{m_{TT}}{|K_T|} \log \frac{m_{TT}}{|K_T|}\right) + \left(\frac{m_{TN}}{|K_N|} \log \frac{m_{TN}}{|K_N|}\right)\right] \tag{3.3}$$

Notice that the calculation expands and removes the summation for each class and expands the probability P_{ij} to show how it is calculated in this case. m_{TT} denotes the number of pages from the given website (according to the known classification, see chapter 4) in K_T, similarly m_{TN} denotes the number of pages not from the given website in set K_T. (Clearly, the size of cluster K_T satisfies $|K_T| = m_{TT} + m_{TN}$; the cardinality of cluster K_T equals the sum of all target and noise items in the cluster).

The entropy calculation for a single cluster is given above, for completeness, the formula to calculate the entropy for a set of clusters is defined as $e = -\sum_{i=1}^{K} \frac{m_i}{m} e_i$ where K is the number of clusters, m is the total number of data points, m_i is the number of objects in cluster i and e_i is the entropy for cluster i.

Given the general entropy calculation, the total entropy e_K for a set of two clusters K_T and K_N can be defined as follows:

$$e_K = \frac{|K_T|e_T + |K_N|e_N}{|W|} \tag{3.4}$$

Where e_N and e_T is the entropy of cluster K_N and K_T respectively. Again notice for simplification purposes that the summation has been expanded.

Purity. Purity is a measure of the extent to which a cluster or cluster configuration contains objects of a single class [176]. The general formula to calculate the purity of a cluster is defined as $p_i = max_j P_{ij}$. Where max_j is the total number of possible objects within class j and P_{ij} is as defined above. The purity of a single cluster K_T is defined as:

$$p_{K_T} = max_{C_T} \frac{m_{TT}}{|K_T|} \tag{3.5}$$

Where max_{C_T} is the maximum number of target objects in the target class, previously defined as C_T and similarly for the noise class C_N. The values m_{TT} and $|K_T|$ are as denoted above. The general calculation for the purity of a set of clusters is defined as $p_K = -\sum_{i=1}^{K} \frac{m_i}{m} p_i$. Where p_i is the purity of cluster i. Given the general formula to measure the purity of a set of clusters, the purity of a set of two clusters K_T and K_N is defined as:

$$p_K = max_{C_T} \frac{m_{TT}}{|K_T|} + max_{C_N} \frac{m_{TN}}{|K_N|} \tag{3.6}$$

Precision. The precision is the fraction of a cluster that consists of objects of a specified class [176]. The general formula used to calculate precision of a single cluster is simply defined as $pr_{ij} = P_{ij}$. The precision is measured relative to a given class. Therefore there are no derivations of this value that can scale to a set of clusters. Therefore the precision can either be measured relative to the target class C_T or the noise class C_N of a specific cluster configuration K. The precision associated with target class C_T is defined as:

$$pr_{K_T C_T} = \frac{m_{TT}}{|K_T|} \tag{3.7}$$

Where, as noted above, m_{TT} is defined as the number of objects of class C_T in cluster K_T.

Recall. The recall is the extent to which a cluster contains objects of a specified class [176]. The general formula used to calculate the recall associated with a single cluster is defined as $r_{ij} = \frac{m_{ij}}{m_j}$, where m_j is the number of objects in class j. Given the generalisation of the formula, the calculations for measuring the recall of cluster K_T relative to class C_T is defined as:

$$r_{K_T C_T} = \frac{m_{TT}}{|C_T|} \tag{3.8}$$

As can be noted from the calculation, the recall measurement is similar to the precision measure above. The recall is measured relative to a single cluster, as is precision. The main difference is that in the case of recall the denominator is the cardinality of the class in question (in this case C_T), where as in the precision calculation the denominator is the cardinality of the cluster (in this case K_T).

Fmeasure. The fmeasure measures the extent to which a cluster contains objects of a particular class with respect to all objects of that class. The fmeasure uses the precision and recall values. The formula below calculates the fmeasure for the cluster K_T relative to the class C_T:

$$f_{K_T C_T} = \frac{(2 \times pr_{K_T C_T} \times r_{K_T C_T})}{(pr_{K_T C_T} + r_{K_T C_T})} \tag{3.9}$$

Score. This is a measure defined by the author. The idea is to provide a value which incorporates all the above previous measures so as to give a single value whereby a WBD solution can be evaluated. The score value for a particular WBD solution is simply the average of the measured accuracy, entropy, purity, precision, recall and fmeasure. Use of the score value allows for the comparison and ranking of WBD solutions.

Coverage. Finally it is important to introduce the concept of coverage. This is particularly important for the dynamic approaches that will be presented in chapter 7. The calculation is simply a measurement of the fraction of target pages C_T and noise pages C_N that are retrieved at any point in a dynamic traversal of a graph. For a particular clustering K, the target page coverage is calculated as:

$$ct_K = \frac{M_{TT} + M_{NT}}{C_T} \tag{3.10}$$

Similarity for noise pages in K is calculated:

$$cn_K = \frac{M_{TN} + M_{NN}}{C_N} \tag{3.11}$$

Also, intuitively the total coverage of a dynamic walk of a graph containing only the target and noise nodes can be calculated as:

$$c_K = \frac{M_{TT} + M_{NT} + M_{TN} + M_{NN}}{C_N + C_T} \tag{3.12}$$

3.5 Summary

To summarise, this chapter presented the website boundary detection problem with respect to the work described in this thesis. The ambiguities in the existing definitions of a website were highlighted and it was argued that they were unsuitable for WBD. Subsequently a definition of a website was proposed. The formal description of the WBD problem and the two contexts whereby it may be addressed, namely the static dynamic context, was also presented. In the following chapter an explanation of the data sets used to evaluate the proposed WBD solutions is presented. This is followed by consideration of both the static and dynamic proposed solutions to the WBD problem.

Chapter 4

Data sets

This chapter presents a review of the data sets that are used to evaluate the approaches in this thesis. The data sets used in this work can be divided into two types; (1) synthetic data sets generated by the author and (2) real data sets, so called as they were created by crawling the web.

Related research reported on the WBD problem used data sets that were specified to each individual study. This makes comparisons with this related work very difficult. The most common method of producing a data set used in previous research is to use hand annotated collections of data extracted from the WWW [90, 160, 105]. This method has also been adopted in this study to produce the Real Data Graphs (RDGs, see section 4.3).

In this chapter the data sets are presented in three sections; the data sets gradually increase in sophistication and complexity as the chapter progresses. The first two are synthetically generated by the author, the last was created using WWW data. The first section, section 4.1 presents Binomial Random Graph (BRG) data sets, which were synthetically generated and designed to model the WBD problem in a simplistic manner. Section 4.2 presents Artificial Data Graph (ADG) data sets which reflect more "web like" data, generated based on the preferential attachment model. The final section (section 4.3) presents the Real Data Graph (RDG) data sets. These data sets were created using portions of the WWW collected from the University of Liverpool host domain.

The synthetically generated data sets BRG and ADG presented in section 4.1 and 4.2 respectively, are used to evaluate the dynamic approaches in chapter 7 only, and are not used to evaluate static approaches in chapters 5 and 6. This is because the dynamic approaches focus more on the structural connections for the purpose of providing an incremental clustering output with respect to WBD. The BRG and ADG are used to evaluate the traversal of the dynamic approaches in a controlled environment.

4.1 Binomial Random Graphs (BRGs)

With respect to the work described in this thesis, the WBD problem was first simplistically modelled using Binomial Random Graphs (BRG) generation approach. The problem was reduced to a scenario defined using two distinct collections of web pages, target pages and noise pages (as previously described in section 3.4). The first represents the collection of target pages in a website boundary (C_T), and the second represents noise pages (C_N) not in the website boundary. In terms of graph structures, these distinct collections where each modelled using a single cluster. The clusters were similar (high intra connections between nodes) while remaining dissimilar between clusters (low inter connections between nodes).

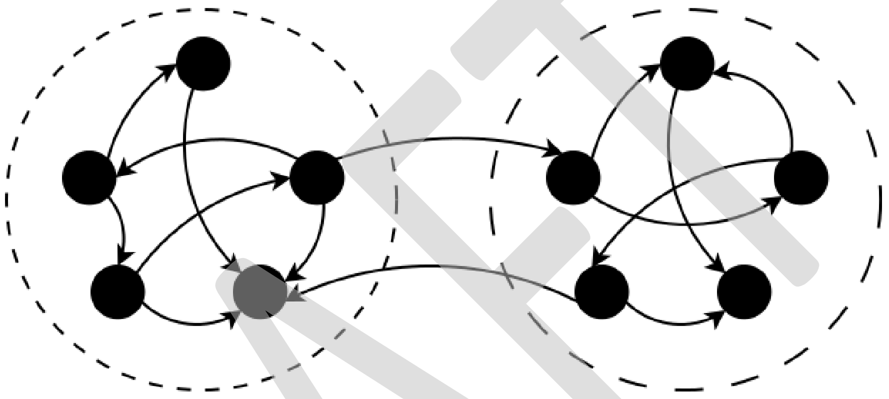


Figure 4.1: Example of a Binomial Random Graph

A random graph model $G(n, p)$ is constructed by randomly including edges between nodes (intra) with probability p [74]. This model is popular due to its simplicity. A binomial graph is referred to as a graph that has two distinct structures (Figure 4.1). The binomial random graphs used for evaluation were generated by connecting two independently generated random graph clusters. Connections were then added between the two graphs (inter edges) with probability $p\prime$ to produce a single binomial graph.

The following two sub-sections describe two specific types of data set generated using the binomial random graph method described above: (1) simple binomial random graph data sets (sub-section 4.1.1) and (2) feature binomial graph data sets (sub-section 4.1.2).

4.1.1 Simple Binomial Random Graphs

The simple Binomial Random Graph (BRG) data sets were synthetically generated to simulate the simplified WBD problem. Two clusters of n nodes where generated (without features). For every pair of nodes (u, v) an edge was added according to a probability p. The two generated clusters of nodes represented target C_T and noise C_N web pages. A number of edges were then added between the two generated clusters.

The number of edges added essentially controls the difficulty of defining which nodes belong to which clusters. The number of nodes and connections that were used for evaluation purposes was extensive, therefore not all the results are reported in this thesis due to space limitations and similar performance. In the data sets created for evaluation presented in this thesis the number of connections were added based on an increasing scale relative to the number of nodes in the clusters. The cluster sizes where fixed at $n = 50$, subsequently the connections across clusters were:

- 5 connections (10%)
- 50 connections (100%)
- 500 connections (1000%)

In each case two groups of data sets were created, three data sets in each group: complete data sets and partial data sets. The details of their generation and associated graph properties are given below.

Complete For the complete BRG data sets the probability of a connection existing between nodes of each cluster was $p = 1$, hence all nodes in each cluster were connected to each other. The clusters therefore exhibited a strong structural cohesion between nodes. The graph properties are shown in Table 4.1.

Name	**Cluster C_T**			**Cluster C_N**		
	Size	**Intra edges**	**Inter edges**	**Size**	**Intra edges**	**Inter edges**
BRG C5	50	2450	2	50	2450	3
BRG C50	50	2450	26	50	2450	24
BRG C500	50	2450	259	50	2450	241

Table 4.1: The graph-theoretic statistics for the Complete BRG data sets. *Intra edges* are links connecting two pages in the same cluster. *Inter edges* are links connecting pages in different clusters.

Partial The Partial BRG data sets were generated using a probability of a connection existing between nodes of each cluster was $p = 0.5$ (in contrast to $p = 1$ for the complete data sets). This created clusters that were less structurally cohesive. The graph properties are shown in Table 4.2.

4.1.2 Feature Binomial Random Graphs

The Feature Binomial Random Graph data set were manifested versions of simple BRG data sets in that a set of features was associated with each node. The features for a

Name	**Cluster C_T**			**Cluster C_N**		
	Size	**Intra edges**	**Inter edges**	**Size**	**Intra edges**	**Inter edges**
BRG P5	50	1275	3	50	1252	2
BRG P50	50	1229	27	50	1260	23
BRG P500	50	1211	254	50	1256	256

Table 4.2: The graph-theoretic statistics for the Partial BRG data sets. *Intra edges* are links connecting two pages in the same cluster. *Inter edges* are links connecting pages in different clusters.

single cluster of n nodes was a distribution in $d = 2 * n$ dimensions. The means of each distribution could be varied to create difficulty in defining the similarity between clusters. In the data sets generated the dimensions was set to $d = 100$, the feature distributions for each cluster C_T and C_N were plus or minus a random Gaussian distribution.

The method of generating the clusters was identical to that of the Partial BRGs above, where $n = 50$ and $p = 0.5$. This created clusters that were less structurally cohesive than complete BRGs, while the addition of the feature distribution created extra cohesion as if to model the WWW. The graph properties are shown in Table 4.3.

Name	**Cluster C_T**			**Cluster C_N**		
	Size	**Intra edges**	**Inter edges**	**Size**	**Intra edges**	**Inter edges**
BRG F5	50	1261	3	50	1255	2
BRG F50	50	1235	28	50	1243	22
BRG F500	50	1252	254	50	1237	256

Table 4.3: The graph-theoretic statistics for the Feature BRG data sets. *Intra edges* are links connecting two pages in the same cluster. *Inter edges* are links connecting pages in different clusters.

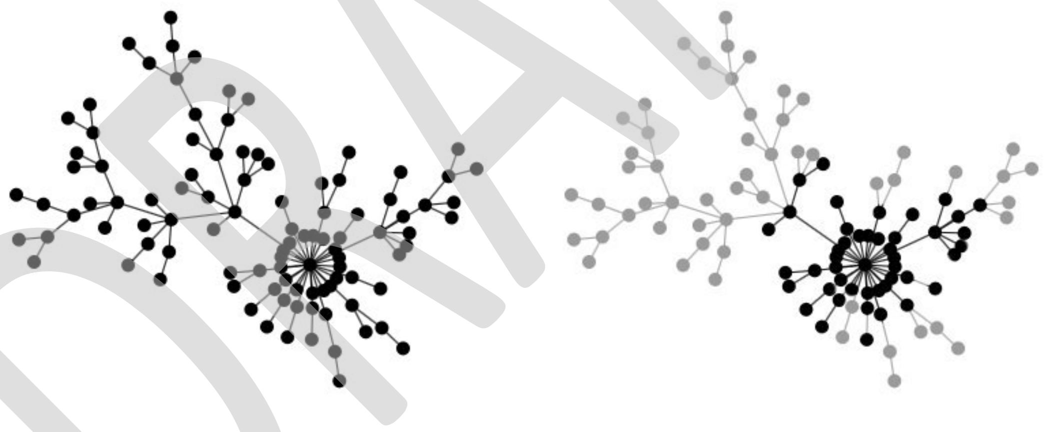
4.2 Artificial Data Graphs (ADGs)

This section presents the second method used to generate the Artificial Data Graphs (ADGs) used for evaluation purposes with respect to the work described in this thesis. The ADGs data sets generated in this case were more sophisticated than the BRGs presented in the previous section. They where generated using the preferential attachment model proposed by Kumar *et al.* [112]. This model produces a graph structure that is a better reflection of the WWW with respect to the WBD problem than the previous BRG method. Sub-section 4.2.1 that follows presents the method that was used to generate the artificial data sets. Section 4.2 then presents detail on the composition of the data sets generated.

4.2.1 Artificial Data set generation

The first step to generating an artificial data set was to create a host graph (Figure 4.2a). The graph $G = (V, E)$ containing a particular set of pages similar to some chosen element of V were put together by creating an artificial host graph using the web-graph model proposed by Kumar *et al.* [112]. Clusters depicting target (C_T) and noise (C_N) were then created using the host graph (Figure **??**). The clusters were then represented using a standard feature vector representation that included noise and target words randomly selected from a "bag" of words.

Figure 4.2: Illustration of the ADG generation. (a) shows the host graph created using the preferential attachment model [112], (b) shows the clusters created using the host graph in (a)


(a) Host graph (b) Clusters C_T (black) and C_N (grey)

Preferential Attachment Model The host graph generation process was founded on that described in [112] however case 3 (below) was derived from [114]. Given a positive integer m, the process generates a synthetic model of the WWW starting from a graph $G_0^{K,m}$ having a single vertex with m links pointing to itself. Then, for $t \geq 1$, $G_t^{K,m}$ is derived from $G_{t-1}^{K,m}$ according to the following procedure (here α, β, and ν are real numbers between zero and one):

1. With probability $\alpha \times \beta$ add a new vertex to $G_{t-1}^{K,m}$ with m links pointing to itself.

2. With probability $\alpha \times (1-\beta)$ choose a random edge in $G_{t-1}^{K,m}$ and make its source point to P.

3. With probability $(1-\alpha) \times \beta$, pick a random copying vertex Q_c; a new vertex P will point to m vertices chosen as follows:

   uar with probability ν, choose a random vertex Q and add (P, Q) to the graph.

   pa with probability $1-\nu$, add (P, R) to the graph, where R is a random neighbour of Q_c.

4. With the remaining probability $(1-\alpha) \times (1-\beta)$ no new vertex is generated and a random edge is added to $G_{t-1}^{K,m}$.

In the artificial data sets generated for evaluation of the proposed approaches presented in this thesis, the value of m was selected so as to mirror the fact that the average page on the WWW contains some 40-50 links [45]. Also $\alpha = 0.01$, $\beta = 0.9$ and $\nu = 0.2$. So, most of the time, new vertices will be generated to which 50 neighbours will be linked according to the procedure described in step 3 above. The resulting graph shared many features with the real web, in particular: (i) in and out degree distribution, (ii) the diameter, and (iii) the presence of small bipartite "cliques". This can be verified by conducting comparison with the Real Data Graph data sets considered later in this chapter (see section 4.3). For the purpose of the experiments, $G_T^{K,m}$ was first generated and then vertex $G_0^{K,m}$ was removed from it (the initial vertex created). Because of the copy mechanism which adds edges to $G_{t-1}^{K,m}$, the single page in $G_0^{K,m}$ contains a large number of links and is linked by a large number of "younger" pages. Removing it from the graph produces a resulting graph, which is denoted by H_T, which is more realistic.

Generating clusters The use of a synthetic version of the web has many advantages. However Kumar's graphs have one disadvantage: the lack of the clusters that are exhibited in the real web. To complete the definition of our artificial graphs a website of target pages C_T needs to be defined within H_T (and then the definition of G completed by adding some noise cluster C_N). To this end we performed the following steps:

1. Given H_T, we picked a random node X from this graph. Such node will represent the home-page in w_s.

2. To create the set of pages in C_T we then performed a breadth-first crawl of H_T starting from X, up to a certain maximum depth. The nodes visited by such process are then added to C_T with some fixed probability p.

3. The noise cluster C_N contains all nodes reachable from X that have not been added to C_T in the previous step.

4. The graph G is then defined as the subgraph of W induced by the set $C_T \cup C_N$.

Given a particular graph structure G, several instances of this structure may be generated by changing the feature vectors that are associated with the various pages (vertices). In general, for any page $P \in C_T$ (or C_N), the feature vector $f(P)$ can be chosen from the superposition of two k-dimensional normal multivariate distributions, having different means, μ_{C_T} and μ_{C_N}, and equal deviations σ. The vectors associated with the elements of C_T may be chosen from the $N_k(\mu_{C_T}, \sigma)$ distribution, those for C_N from $N_k(\mu_{C_N}, \sigma)$. Changing the relative position of the means results in data sets of varying complexity.

This process was simulated in a very simple way. Given P in $V(H_T)$, $f(P)$ contains $k = 20$ integer numbers, corresponding to the number of occurrences of the elements of a pool of words S in a bag associated with P. The pool S is split into two disjoint groups of equal size, S_{C_T} and S_{C_N}. The bag of P is defined by sampling, independently, k elements of S. If P is in C_T, each chosen word has a chance of $\pi = 0.7$ of belonging to S_{C_T}, and chance of $1 - \pi$ of belonging to S_{C_N}. The selection is made uniformly at random with replacement, from the chosen group. If $P \in C_N$, the selection is symmetrically biased towards S_{C_N}.

4.2.2 Artificial Data Graph Sets

The evaluation of the approaches presented in this thesis use two types of artificial graph data set (in addition to BRG data sets), termed Set1 and Set2. Both sets were generated from copies of H_T. Sets of type Set1 were derived from graphs generated using $m = 50$, keeping in the set C_T all nodes at a distance of at most three links from the initial page X. Sets of type Set2 were generated using $m = 30$ and $p = 0.5$ and breadth-first exploration up to distance five. Four data sets of each type were generated

Table 4.4 reports some graph-theoretic statistics of the resulting web graphs. Note that the two clusters have roughly the same number of elements but the class cluster C_T has many more internal links than links pointing to the noise cluster (modelling the fact that the elements of C_T represent homogeneous web-pages). Furthermore C_T is *popular* in the sense that many links from the noise cluster C_N point back to C_T. There are two main differences between the graphs in Set1 and those in Set2. First, typically graphs of type Set2 contain fewer edges and, individually, their nodes contain fewer out-links. Second, for graphs in Set1, the class C_T coincides with the set of all pages at distance at most three from X (also known as the *ball of radius three around* X). For the graphs in Set2 this is not the case. The web-site target pages (C_T) of X, although much bigger than in the case of Set1, is typically more "stringy". Only half of the neighbours of a vertex in C_T are in the cluster.

Type	Name	Cluster C_T			Cluster C_N		
		Size	Intra edges	Inter edges	Size	Intra edges	Inter edges
ADG Set1	G1	181	2443	53	176	470	2353
	G2	124	1581	48	182	591	2335
	G3	102	978	89	128	374	1463
	G4	149	2046	77	109	259	1665
ADG Set2	G1	208	4783	46	201	699	5006
	G2	262	6616	80	199	623	5570
	G3	254	5956	76	212	706	5554
	G4	293	6108	128	237	766	5500

Table 4.4: Graph-theoretic statistics for artificial graphs of type Set1 and Set2. *Intra edges* are links connecting two pages in the same cluster. *Inter edges* are links connecting pages in different clusters.

4.3 Real Data Graphs (RDGs)

This section presents the method used to create the Real Data Graphs (RDGs) used for evaluating the approaches in this thesis. The four RDG data sets (LivChem, LivHistory, LivMath and LivSace) were created from the departmental web pages hosted by the University of Liverpool. The method to generate the four RDG data sets first involves collecting the data from the WWW. The web pages collected were then manually labelled by an assessor relative to the definition proposed in section 3.2. Details of the methods are presented in section 4.3.1. Analysis was then subsequently performed to give some in depth graph theoretic statistics based on the potential target pages (C_T) contained in the website and the surrounding noise pages (C_N). The analysis and details on the data sets is presented in section 4.3.2.

4.3.1 Real Data set generation

The method used to generate the RDG data sets are presented below. Details are given on the two main processes used. The first is data collection, where WWW pages are downloaded using a given seed home page (w_s) and a size estimation. The second is data labelling where labels are assigned to the web pages that were downloaded.

Data Collecting The data collection phase commences with an estimation of the number of web pages that should be "grabbed". It is necessary to grab a sufficiently large portion of the web so as to ensure (with some degree of certainty) that the target pages of a website of interest are contained within the collection. Creating a collection of web data is a process of first estimating the amount of data to grab, and then the method by which to grab this data.

The size of the data set to be grabbed can be defined in terms of either: (i) a number of pages n, or (ii) a depth of crawl de. The two values can be argued to be related in that n can be defined in terms of both the depth and width of crawl (wi) such that $n = de \times wi$ (Figure 4.3). An estimation for de can be obtained manually by conducting an experimental crawl from w_s till the pages of the website are no longer traversed. An estimation can also be obtained using the value w from the number of links contained in w_s. An example formula to calculate nodes is: $n = wi^{(de-1)} \times alpha$. An example formula to calculate depth is: $de = wi \times beta$. In both cases a substantial over estimate is required to ensure that nothing is missed.

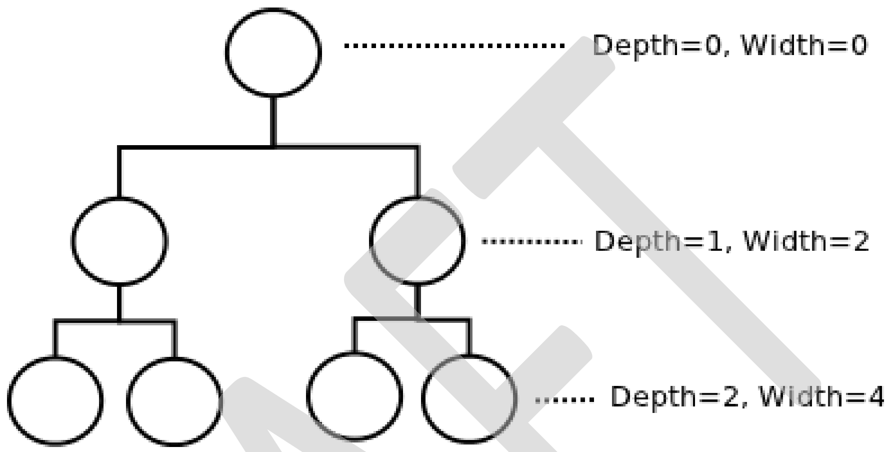


Figure 4.3: Example graph with depth and width values for each level.

Once an appropriate value for n or de has been derived the next step is to gather the desired collection of web pages. There are two broad techniques that can be adopted: Breadth First (BF) or Depth First (DF) web traversal. The distinction is that a BF-crawl can provide a set of pages limited by de or n, while a DF-crawl is limited by de (if DF is limited by n there is no depth limit, and the collection gathered will be a collection of pages in order of depth). Note that the crawling should not be limited to (say) URL domain or file size; the search should be able to span multiple physical domains. A BF web crawl was used to gather upto $n = 500$ pages for each of the four RDG data sets presented in section 4.3.2.

Data Labelling A common technique used to label data is to perform a manual analysis to label pages given some criteria. This technique was adopted to label the pages in each collection that corresponded to target pages. By definition, if a web page is not labelled as a target page, it is a noise web page. The web pages collected for each department were manually labelled by an assessor relative to a given seed home page (w_s) in each case and the definition proposed in section 3.2.

4.3.2 University of Liverpool Departments

The RDG data sets used were generated using web pages hosted by the University of Liverpool. Four departmental home pages were identified and web graphs generated. Four similar departments from the same university were selected so as to provide non-trivial examples of the website boundary detection problem. This was in contrast to selecting very dissimilar web pages in which the boundary detection problem becomes a trivial task. The four departments were: (1) the Department of Chemistry (LivChem), (2) the Department of History (LivHistory), (3) the Department of Mathematics (LivMaths) and (4) the Department of Archaeology, Classics and Egyptology (LivSace).

To analyse the nature of the clusters featured in the real data a number of metrics were used. The measurements included some basic graph properties, for instance number of vertices and edges. Of these edges the number of unique, duplicated and self loops were also recorded. The maximum diameter and average geodesic distance shows the longest shortest path between vertex pairs and the average distance between these pairs in the graph. The number of connected components shows the amount of connected sub-graphs that are not connected to any other components. If the value is one, this implies that all components are connected in some way, which means that there exists a path from every node to every other node in the graph, in other words all nodes are reachable by traversing edges. If the value is more than one (assuming vertices >1) then this implies that the graph contains a number of disconnected sub-graphs.

The degree value for a vertex shows how many edges that a vertex has. In this case the maximum and minimum values are reported. The average degree shows a value for the connectivity of the vertices across the cluster, the higher this value, the more dense the graph. The density is a value that is used to measure how dense or sparse a graph is, or how connected or not connected the vertices are. The value can range from 0 to 1, 1 representing the density or the complete graph.

The number of vertices in the target cluster in each case was found to be more than 5 fold that of the noise cluster. It was also discovered that there were a large number of duplicate and self looping edges in each of the clusters. The diameter value of the entire graph was consistent with the diameter of the target and noise clusters, this implied that there was an even spread of noise and target items in the graphs. Which essentially demonstrated that there was virtually a uniform diameter in all clusters.

The whole graph with respect to each data set had one Connected Component (CC). This included all vertices of the graph. This demonstrated that there existed a path from all pairs of vertices, thus all vertices are connected in some way. The single vector CC essentially shows how many instances there are of a single node that is not connected to any other nodes, in the case of the whole graph, this is generally none.

In the case of the elements in the target and noise clusters, the “picture” is pretty much consistent. The number of vertices in a connected component is equal to how many elements are in that cluster. These are single vector connected components, that are connected as leaf nodes to the target cluster, but are not related in terms of the website boundary. They are also not strongly connected to the noise cluster in which they are classified to belong.

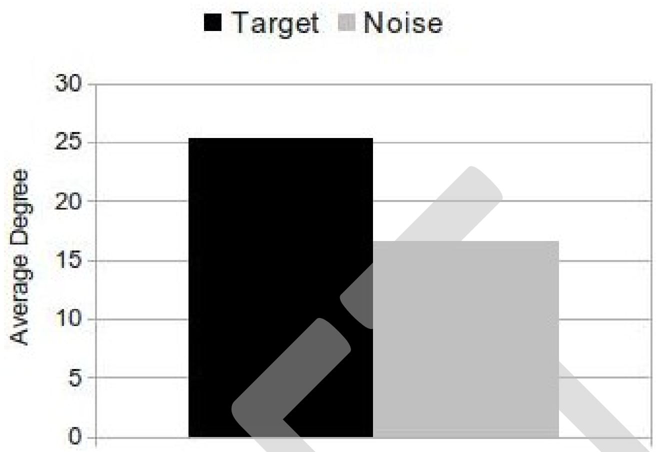


Figure 4.4: Average degree for target (C_T) and noise (C_N) pages for the RDG data sets.

Although the RDG data sets featured different maximal and minimal values of degree across the clusters, the average degree values were in line with the distance values, thus there is a consistent value with respect to all the clusters which shows that the target cluster (C_T) is more connected than that of the noise cluster (C_N), as illustrated in Figure 4.4. Density analysis illustrates that there is in fact a larger connectivity within the target cluster than in the noise cluster. This shows that there is some relationship between nodes of the target cluster that is not exhibited in any other grouping of the elements in the graph (whole or noise clusters).

Tables 4.5, 4.7, 4.9, and 4.11 present the statistical analysis for each of the four RDG data set LivChem, LivHistory, LivMath and LivSace respectively. The confusion matrix which presents the connections of the target and noise clusters are given in Tables 4.6, 4.8, 4.10 and 4.12 for the data sets LivChem, LivHistory, LivMath and LivSace respectfully. Further illustrations of the four RDG data sets are presented in Appendix A showing the relationships between clusters as explained above.

4.3.3 Variations

The total number of web pages that were collected to create the data set presented in the foregoing sub-section was considered to be complete snapshots. In some approaches used in this thesis it is desirable to test the performance of an approach using portions of a complete snapshot. The method used to do this is to replicate various web crawling

Table 4.5: Graph metrics for the LivChem web graph. This includes values measured relative to the whole graph (including all clusters) and also measured relative to the target and noise clusters.

	Whole Graph (W)	Cluster1 (Target C_T)	Cluster2 (Noise C_N)
Vertices	428	59	369
Unique edges	6351	796	4919
Edges with duplicates	1452	174	1095
Total edges	7803	970	6014
Self loops	154	55	99
Max Diameter	4	3	4
Avg geodesic distance	1.977	1.570	1.948
Connected Components (CC)	1	1	5
single vertex CC	0	0	4
max vertex in CC	428	59	365
max edge in CC	7803	970	6013
Min Degree	1	1	1
Max Degree	402	78	402
Avg degree	18.231	29.932	16.360
Density	0.037	0.242	0.039

Table 4.6: Confusion matrix for the connections between clusters of the LivChem graph.

	Target	Noise
Target	970	796
Noise	23	6014

Table 4.7: Graph metrics for the LivHistory web graph. This includes values measured relative to the whole graph (including all clusters) and also measured relative to the target and noise clusters.

	Whole Graph (W)	Cluster1 (Target C_T)	Cluster2 (Noise C_N)
Vertices	424	57	367
Unique edges	5738	300	4921
Edges with duplicates	1497	92	1232
Total edges	7235	392	6153
Self loops	137	38	99
Max Diameter	4	5	4
Avg geodesic distance	1.967	1.951	1.951
Connected components(cc)	1	1	1
single vertex cc	0	0	0
max vertex in cc	424	57	367
max edge in cc	7235	392	6153
Min Degree	1	1	1
Max Degree	402	78	402
Avg degree	17.063	18.894	16.779
Density	0.034	0.093	0.040

Table 4.8: Confusion matrix for the connections between clusters of the LivHistory graph.

	Target	Noise
Target	392	685
Noise	5	6153

Table 4.9: Graph metrics for the LivMath web graph. This includes values measured relative to the whole graph (including all clusters) and also measured relative to the target and noise clusters.

	Whole Graph (W)	Cluster1 (Target C_T)	Cluster2 (Noise C_N)
Vertices	428	64	364
Unique edges	6351	1035	3634
Edges with duplicates	1452	350	279
Total edges	7803	1385	3913
Self loops	154	61	93
Max Diameter	4	3	4
Avg geodesic distance	1.977	1.493	2.140
Connected components(cc)	1	1	6
single vertex cc	0	0	5
max vertex in cc	428	64	359
max edge in cc	7803	1385	3912
Min Degree	1	1	1
Max Degree	402	78	402
Avg degree	18.231	35.265	15.236
Density	0.037	0.279	0.028

Table 4.10: Confusion matrix for the connections between clusters of the LivMath graph.

	Target	Noise
Target	1385	872
Noise	1633	3913

Table 4.11: Graph metrics for the LivSace web graph. This includes values measured relative to the whole graph (including all clusters) and also measured relative to the target and noise clusters.

	Whole Graph (W)	Cluster1 (Target C_T)	Cluster2 (Noise C_N)
Vertices	411	14	397
Unique edges	5907	35	5727
Edges with duplicates	1522	10	1447
Total edges	7429	45	7174
Self loops	144	2	142
Max Diameter	4	3	4
Avg geodesic distance	1.967	1.643	1.952
Connected components(cc)	1	1	4
single vertex cc	0	0	3
max vertex in cc	411	14	394
max edge in cc	7429	45	7174
Min Degree	1	1	1
Max Degree	402	78	402
Avg degree	18.075	17.142	18.108
Density	0.038	0.209	0.039

Table 4.12: Confusion matrix for the connections between clusters of the LivSace graph.

	Target	Noise
Target	45	195
Noise	15	7174

strategies that collect a smaller amount of the web. To create the complete snapshot, starting at the seed page, a BF crawl of the web was conducted until a maximum number of pages was crawled. To create a smaller portion of the data in a snapshot the BF crawl was performed from the seed page, but the maximum number of pages was reduced. What these shortened crawls represent is an instance where the crawl strategy is such that a smaller portion of data is collected, which in fact can include fewer noise web pages when compared to the complete snapshot.

A systematic method starting at 50 and increasing by the same amount was used as the maximum pages collected in varying snapshot until the total number of pages in a complete snapshot was reached (50, 100, . . . Complete). The varying snapshots were created after the initial complete snapshot data had been acquired and labelled. At this point, the amount of target and noise web pages were known, thus a more informed decision could be made on the values to obtain smaller variations of the snapshots. In contrast to using a smaller initial estimate when creating the complete snapshot before labelling, and thus missing out on target data.

4.3.4 Comparison of data sets

It is shown in Table 4.13 that for each data set, the density of the target cluster is greater than that of the noise cluster. The noise cluster has similar density to that of the whole graph, which shows that there is clear structural relationship between the vertices of the target cluster, in comparison to the base line density of the whole graph, and of the noise cluster. It should also be noted that the LivChem and LivMaths data sets show a much higher density value for their respective target clusters.

With respect to the degree of the data sets, the LivHistory and LivSace collections have a fairly uniform degree with respect to each identified cluster. The target and noise clusters have similar values of degree in comparison to the whole graph. LivChem and LivMaths however exhibit an increased average degree for the target cluster in comparison to the noise and whole graph clusters. It can also be seen that the noise cluster has a slightly lower average degree when compared to the average for the entire graph. This suggests that both LivChem and LivMaths have a target cluster that is more cohesively connected in terms of hyperlink structure than that of the LivHistory and LivSace data sets.

The higher connectivity of LivChem and LivMaths is further corroborated by the values associated with the connected components of each graph. The total number is higher for the target clusters of LivChem and LivMaths in comparison to LivHistory and LivSace. It can also be observed that there are more connected components present in the noise clusters of LivChem, LivMath and LivSace than that of the LivHistory data set. In certain cases, particularly if traversing the graph structure, an increased number of connected components can lead to an increased visitation frequency of vertices.

Table 4.13: Comparison of four University of Liverpool department data sets.

	LivChem	**LivHistory**	**LivMaths**	**LivSace**
Density				
Whole (W)	0.037	0.034	0.037	0.038
Target (C_T)	0.242	0.093	0.279	0.209
Noise (C_N)	0.039	0.04	0.028	0.039
Avg Degree				
Whole (W)	18.231	17.063	18.231	18.075
Target (C_T)	29.932	18.894	35.265	17.142
Noise (C_N)	16.36	16.779	15.236	18.108
Connected components (CC)				
Whole (W)	1	1	1	1
Target (C_T)	1	1	1	1
Noise (C_N)	5	1	6	4
Max edges in CC				
Whole (W)	7803	7235	7803	7429
Target (C_T)	970	392	1385	45
Noise (C_N)	6013	6153	3912	7174

4.4 Summary

This chapter has presented the data sets that were used to evaluate the approaches to the WBD problem proposed in this thesis. Three kinds of data set were used: (i) Binomial Random Graphs, (ii) Artificial Data Graphs and (iii) Real Data Graphs. The first two used synthetic methods to produce data sets that model the WBD problem according to various scenarios. In total 3 BRG data sets and 2 ADG data sets were generated. The 4 RDG data sets were created by collecting and labelling web pages hosted by the University of Liverpool. A statistical analysis concerning each of the data sets (artificial and real) was also presented.

Chapter 5

Static Technique 1: Feature Analysis

This chapter presents the investigation of the WBD problem in the static context. In the static context all the web data is available prior to the start of analysis (as previously explained in section 3.4.2). The approaches in this chapter use various attributes extracted from web pages to represent features, and these features are then subsequently grouped using a variety of clustering algorithms, and a WBD solution produced. A methodology for providing a solution to the WBD problem in a static context is discussed, and two techniques are proposed; (1) the first uses n-feature types to provide a solution to the WBD problem, (2) the second using feature discrimination to produce the "best" set of features with respect to the WBD problem. A range of possible attributes to represent the web pages, extracted from content and associated meta-data are considered. Clustering algorithms were applied to the various features and website boundary solutions produced. The most relevant features that can be used to model web pages in terms of producing an optimal solution to the WBD problem are identified, along with the most appropriate clustering algorithms. From the experiments presented in this chapter it was found that the most effective features used to represent web pages with respect to WBD are: links to scripts (e.g Java script code), image links (e.g links that reference .png, jpg images) and resource links (e.g links that reference CSS styling files). The most appropriate clustering algorithms for WBD in the static context were found to be DBScan and kmeans from the specified selection used in this work.

The rest of this chapter is organised as follows, in section 5.1 some formal description is given, followed by details on the feature representation in section 5.2. The static approach is presented in section 5.3, with two techniques whereby this approach can be implemented is presented in section 5.4.1 and 5.4.2 respectively. The evaluation of the feature representation and clustering algorithms used is presented in section 5.5. This chapter is concluded in section 5.6

5.1 Formal Description

Recall the general WBD problem's formal description (see section 3.4). Given a collection of web pages W comprising n individual pages, such that $W = \{w_1, w_2, \cdots, w_n\}$, where the seed page is w_s. The website boundary (ω) is said to be the bounded subset of pages in W that form the website given by w_s. Each of the individual web pages $W = \{w_1, w_2, \cdots, w_n\}$ can be described using a dimensional vector length m, such that $V = \{v_1, v_2, \dots, v_m\}$. Each dimension of the feature space $v_1, v_2, \dots, v_m$ describes a single feature.

In contrast to the general WBD formal description, in the static context considered in this chapter, there is a key characteristic in the composition of the set of features V. The set V is constructed from some arbitrary concatenation of global features in set $F = \{f_1, f_2, \dots, f_b\}$, where b is the total number of sub features. The set F serves the purpose of defining the actual sub features making up a particular composition of set V.

The sequence (set) of possible values for a feature f_i is then described by the set of values $\{\varphi_{i_1}, \varphi_{i_2}, \cdots, \varphi_{i_k}\}$. The value of k will depend on the nature of the feature; given a binary valued feature $k = 2$ the value set will be $\{0, 1\}$. Given (say) a numeric feature the value of k may be substantial. An example is shown in Figure 5.1.

$$VSM = \left[\begin{array}{c|cccccc} & \multicolumn{2}{c}{\overbrace{v_1 \quad v_2}^{f_1}} & \overbrace{v_3}^{f_2} & \multicolumn{3}{c}{\overbrace{v_4 \quad \dots \quad v_m}^{f_3}} \\ \hline w_1 & 1 & 0 & 0 & 1 & \dots & 1 \\ w_2 & 1 & 0 & 1 & 0 & \dots & 1 \\ w_3 & 0 & 1 & 1 & 1 & \dots & 1 \\ w_4 & 1 & 0 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ w_n & 1 & 0 & 0 & 1 & \dots & 1 \end{array}\right]$$

Figure 5.1: The vector space model for the WBD problem using some formal notation. In this example $k = 2$ for all features and thus a binary value of $\{0, 1\}$ is allocated to each feature f_1, f_2 and f_3.

The complete set of features F, comprises one or more sub-sets of features (each describing a different aspect of a www page, see section 5.2). Set F gives the maximum dimensions of possible values that can be used to describe a web page. Feature vectors for individual www pages are thus created by concatenating together sub-vectors $(f_1, f_2 \dots)$ representing individual features. An illustration is given in Figure 5.2.

Each web page in the collection W can be represented as a feature vector. What web page features to include in the feature space is a subject for debate that will be considered later in this chapter (see section 5.2). The more features that are included the greater the computational overhead. Clearly it is also not desirable to include sub features that are not good discriminators; the question is what are the features that make good discriminators?

Once a set of features has been selected to represent the web pages as a vector space, the WBD clustering paradigm as described in section 3.4.1 is applied. Recall that the clustering paradigm essentially groups web pages into two sets, K_T and K_N, representing the target cluster and noise cluster respectively.

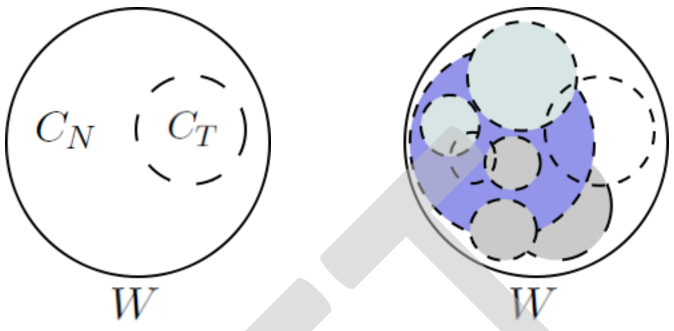


Figure 5.2: An example data set W (left), showing a possible cluster configuration using different features (right). Each coloured area represents a related set of pages using a particular sub feature (f_i), overlap of clusters occurs as different features express similar relationships.

5.2 Representation

The formalisation presented in section 5.1 above prescribed the construction of feature vectors from a concatenation of values corresponding to a set of chosen features. It was also noted that the set of features comprises one or more groupings (subsets) of particular types of features. There are a number of different types of feature that can be included in the representation. The different categories considered in this thesis are as follows:.

Hyper links Hyper links are an obvious candidate for inclusion in any feature space describing a collection of www pages. The theory is that web pages that are related may share many of the same hyper links. The shared links may be other pages in the same website (e.g. the website home page) or significant external pages (most links a point to a related set of pages with some common topic). The hyper link based features were constructed by extracting all of the hyper links from the collection of web pages. Each hyper link was then considered to be an individual binary-valued feature.

Image links The use of image links was prompted by the observation that web pages that link to the same images were likely to be related, for example a common set of logos or navigation images could imply a relationship. The image links were processed in a similar fashion to the hyper-links, as described above.

Mailto links If a set of web pages includes identical mail links this might indicate a relationship between these web pages. The links were extracted from the HTML code using the same method as described above, but by searching for the $Mailto$ tags.

Page Anchor links Page anchors are used to navigate to certain places on the same page, these can be helpful for a user and can very often have meaningful names. It was conjectured that if the same or related names are used on a set of web pages it could imply related content. The Page Anchor Links were extracted by parsing the HTML code as above and identifying the number of possible occurrences.

Resource links The motivation for using resource links was that the styling of a page is often controlled by a common Cascading Style Sheet (CSS) which could therefore imply that a collection of pages that use the same style sheet were related. In this case the feature subset was obtained by extracting the appropriate resource links from the HTML code.

Script links It was observed that some scripting functions that are used in web-pages can be written using some form of common script file; if pages have common script links then they could be related. The script links vector was constructed by extracting all of the script links (for example Java script links) from each of the pages in W.

Title text It is conjectured that the title text used within a collection of web pages belonging to a common web site is a good indicator of "relatedness". The title group of features was constructed by extracting the title from each of the given web pages. The individual words in each title were then processed to produce a "bag of words" (a common representation used in text mining). Each word represented a binary valued dimension in the feature space. Note that when the textual information was extracted from the *title* tag non-textual characters were removed, along with words contained in a standard "stop list". This produced a group of feature values comprised only of what were deemed to be the most significant title words.

Body text Another key indicator of web page "relatedness" was considered to be content, as reflected by the text contained in WWW pages. Textual content was extracted from each web page using a html text parser/extractor (`http://htmlparser.sourceforge.net/`). This type of tool extracts the text as it would be rendered by a web browser. This was deemed to be the same text that a user would use to judge a pages topic/subject. Stop words (same list as used to process the title text as described above) were then removed and a bag of words produced similar to that used in the case of the title feature sub-set.

URL Web page URLs are likely to be an important factor in establishing whether subsets of web-pages are related or not. URLs should not be considered to be a unique indicator for establishing a web site boundary (referring to URL filtering methods, see section 2.2.1). In each case the page URL was split into "words" using the standard delimiters found in URL's. For example the URL `http://news.bbc.co.uk` would produce the sub-set of features $\{news, bbc, co, uk\}$. Non textual characters were removed (no stop word removal was undertaken).

The individual feature subsets listed above could be used, either in isolation or in various combinations, to produce a feature space which in turn could be used to describe individual web pages. It should be noted that the combination, for example, of the hyper links and the image links feature could created a very large vector space, in which potentially useful information could be swamped. The use of certain elements of these features could provide useful information; a hyper link to the home page, and image links contained on the home page, could be the most important sub features of this concatenation. Although the further processing of sub elements of features could be explored with the use of feature extraction techniques, such techniques are beyond the remit of this thesis.

5.3 The Static Approach

In the static context, the WBD problem is concerned with identifying website boundaries in a predetermined collection of web pages. These web pages are gathered prior to performing any analysis to detect the desired website boundaries. The static context is therefore defined as the situation where all the data to be processed is available at the commencement of the process (as will become clear later in this thesis, this is not the case in the dynamic context).

In this section a general static WBD process is described. The process may be implemented using a variety of different techniques, two of which are proposed in this chapter and are discussed in further detail in section 5.4.1 and 5.4.2 below.

The set of steps involved in the static WBD process are shown in table 5.1. From the table it can be seen that the static process comprises three phases: (i) data collection, (ii) data modelling and (iii) data analysis. Each phase is described in further detail in the following three sub-sections below.

1. Data Collection
    - Identify and gather WWW data
2. Data Modelling
    - Select features
    - Feature extraction
    - Representing as vector space
3. Data Analysis
    - Apply clustering algorithm
    - WBD solution

Table 5.1: Overview of processes involved in a general static approach to the WBD problem.

5.3.1 Data Collection (Phase 1)

The web page data needs to be collected prior to any analysis being performed. In the work described in this chapter, the RDGs data sets (LivChem, LivHistory, LivMath and LivSace) are used (section 4.3.2). A breadth first web crawl is performed from a given seed page (w_s) until an estimated stopping criteria is reached (further details are given in section 4.3.1).

When the dataset W has been collected (downloaded) some pre-processing is applied ready for the application of feature extraction (Phase 2). Depending on the content of the web page (html/xhtml/xml and so on), various parsing steps were applied including the validating or re-requesting of pages where content was found to be incomplete or corrupt.

5.3.2 Data Modelling (Phase 2)

This section explains the data modelling phase of the static approach for a given collection W. This includes three distinct sub-phases, namely: (1) selecting the features, (2) pre-processing to extract the selected features and (3) creating the vector space to model the web pages W.

Select Features: During the Select Features sub-phase the set of features F to be used to model the collection of web pages W are identified. In Section 5.2 a number of feature types (groupings) were identified. Any number of these groupings may be combined to form F and consequently define a feature space.

Value Extraction: Using the set of selected features F, the next step is to determine the set of values for each feature by parsing W. The result is a definition of the feature space to be used to generate the desired feature vector representation for the contents of W. The preprocessing of each resource in a collection W can be costly. Data from the web is often malformed, and often needs to be parsed and validated prior to any features extraction.

Vector Space Representation: During this last step in Phase 2 (Table 5.1) the web pages contained in W are described in terms of feature vectors using the identified feature space. At the end of Phase 2 each of the pages j in W will have an associated vector v_j representing its content.

5.3.3 Data Analysis (Phase 3)

This section explains the data analysis phase of the proposed static approach to the WBD problem. Recall that the challenge of the data analysis phase is to identify a sub set of web pages ω ($\omega \subset W$) such that ω holds all the pages included in a website and consequently also defines the boundary of that website. Recall also that W is modelled using a set of identified features F which in turn defines a feature (vector) space. To achieve this a two stage process was adopted: (1) web page clustering and (2) website boundary identification. During the first stage the web pages in W were grouped according to some notion of similarity to produce K clusters, one of which will include w_s and will thus be identified as the target cluster K_T. The target cluster is considered to be all the pages that are part of a website, as defined by the given solution. The pages in cluster (K_T) thus form the boundary of the website with respect to the remainder of the web pages in W.

5.4 Static Technique Implementations

This section describes two techniques whereby the above static approach to WBD may be implemented: (1) n-feature types (section 5.4.1), or (2) feature discrimination (section 5.4.2). The distinction between the two techniques is in the way the identified features are used and the techniques are applied.

The n-feature technique describes an implementation of the static approach that can be practically applied to produce a WBD solution given a set of features. The n-feature technique is given a set of parameters; including the seed page, size estimation and selected features to extract from pages, and will conduct all processes involved in producing a WBD solution. The WBD solution produced is subject to the input features used to model the web pages.

The feature discrimination method, on the other hand, has an automated process to iteratively select and test combinations of features from a given list of input features. This technique uses labelled data (target and noise web pages) to measure the performance of the WBD solutions produced from a particular combination of features. Therefore the feature discrimination technique is not a suitable approach to provide a WBD solution (as is the case of the n-feature technique), but instead is used to produce the "best" set of features in terms of WBD performance for a particular WBD problem. Both mechanisms are described in further detail in the following two subsections.

5.4.1 Static Technique: using n feature types (n-features)

This section describes the technique to providing a solution to the WBD problem using n-feature types. The n-features technique can be applied in a static context to produce solutions to the WBD problem given a set of features. The details of the process steps are formally described in table 5.2, and illustrated in Figure 5.3.

The basic processes involved in this technique are to gather the data from the web, given a size estimate and seed page (w_s). This data is then represented using the given set of n-features. This set can include any number of the feature types described in section 5.2 which are then considered in combination. The given clustering algorithm is then applied in order to produce clusters from which a website boundary can then be identified. The static process requires four parameters:

- Feature - The set of feature types that is going to be used to represent the web pages.
- Clustering Algorithm - The user specified clustering algorithm to be used to group pages in the collection.
- Size - The size estimate of the collection to grab, in terms of number of pages n or de depth (see section 4.3.1 for more details).
- Seed Page - A seed page (w_s), which is commonly the homepage of the web site.

The main processes for the n-feature techniques are presented in Table 5.2. A crawl of the web to collect data set W is first performed using the given seed page and size estimate. The set of features f is then extracted with respect to the data collected. A vector space V is then constructed and a clustering algorithm applied to produce a set of K clusters. This set of K clusters is then used to identify the target cluster K_T with respect to the seed page w_s in order to produce a website boundary.

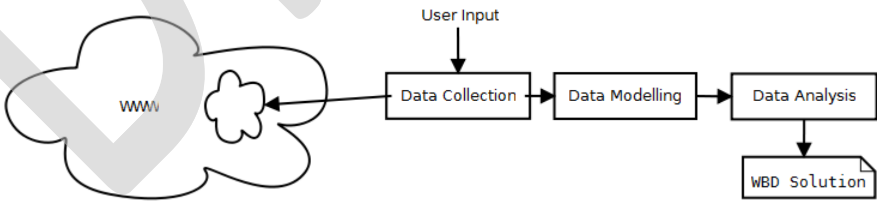


Figure 5.3: An illustration of the n-feature type technique.

5.4.2 Static Technique: using feature types discrimination (feature discrimination)

The second proposed technique to implement the static approach to WBD is the feature discrimination technique; which, as the name suggests, adopts the concept of feature

```
**Process** n-feature (FeatureSet $F$, ClusteringAlgorithm $C$, SizeEstimate $s$,
SeedPage $w_s$)
    $W$ = BF_Crawl($w_s$, $s$);
    $A$ = Extract_Features($F$, $W$);
    $V$ = Build_Vector_Space($A$);
    $K$ = Clustering_Algorithm($C$, $V$);
    $\omega$ = WBD($K$,$w_s$);
    **return** $\omega$;
**Process** Evaluate ($w$);
```

Table 5.2: The n-feature algorithm.

discrimination. This technique iteratively selects features based on the "best" WBD solution produced for each combination of features. This technique uses labelled data (target and noise web pages) to measure the performance of WBD solution produced from a particular combination of features. Therefore the feature discrimination technique is not an approach to provide a WBD solution (as is the case of the n-feature technique), but instead is used to produce the "best" set of features in terms of WBD performance for a particular WBD problem. The feature discrimination technique is used to provide further insight into the performance of features using an information gain style technique of discriminating features for a particular WBD problem.

The technique described in this section uses the same three phase processes described in section 5.3. The difference between the feature discrimination technique and the n-feature technique described in the foregoing sub-section is in the choice of features that are used to represent web pages in the collection. The feature discrimination technique uses the user specified set of features F, and then subsequent combinations are produced and tested based on the highest score performance of the WBD solution with the addition of each new feature in the combination. This produces a combination hierarchy that can be used to access the best combination of features for a WBD problem (as shown in Table 5.4).

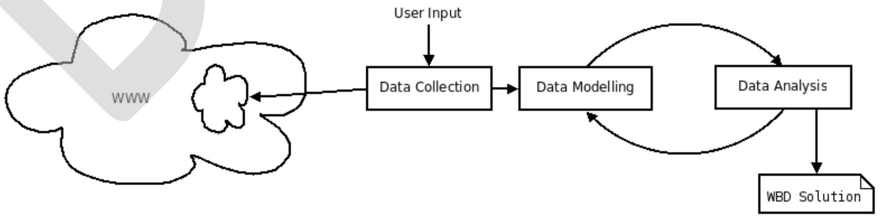


Figure 5.4: An illustration of the feature discrimination technique.

As shown in Figure 5.4 the feature discrimination technique has similar process as shown in the previous technique (Figure 5.3). The important difference is the transition between the Data Modelling phase and the Data Analysis phase. This transition backwards and forwards is depicted for the purpose of discovering the best combination of features. Thus the data modelling phase in this case consists of the construction of feature sets from individual features, the data analysis phase measures the quality of the produced WBD solution using the given feature set.

```
Process
feature discrimination (FeatureSet $F$, ClusteringAlgorithm $C$, SizeEstimate $s$,
SeedPage $w_s$)
  $MAX = |F|$;
  $f = \{\}$; current Feature Set
  $W$ = BF_Crawl($w_s$, $s$);
  while $f < MAX$ do
    $tmp\_f = \{\}$;
    $bestPerformance = 0; performance = 0$;
    generateCombinations($|f|$,$F$);
    while nextCombination() do
      $tmp\_f$ = nextCombination()
      $A$ = Extract_Features($tmp\_f$, $W$);
      $V$ = Build_Vector_Space($A$);
      $K$ = Clustering_Algorithm($C$, $V$);
      $\omega$ = WBD($K$,$w_s$);
      $performance$ = Evaluate($\omega$);
      if $performance > bestPerformance$ then
        $bestPerformance = performance$;
        $f = tmp\_f$;
      end if
    end while
    Print: FeatureSet $f$. Performance $bestPerformance$;
  end while
```

Table 5.3: Feature discrimination algorithm.

The feature discrimination technique focuses on what features to use in terms of optimising the WBD solution in a static context. The table 5.3 shows the processes that take place in this method. The main point to notice about this approach is the collection W is created once, at the start of the process. Upon each iteration a possible set of combinations are produced given the starting feature set F and the cardinality of the current feature set f_i. The features in each of the generated combinations (generateCombinations($|f|$,F)) are iterated from an n-element selection, which is based on $|f_i|$. The combinations in each of these n-element subsets model the collection and produce a WBD solution to the given problem. The highest performing combination is then used to generate the next set of $n+1$-element combinations until $|F|$-elements is reached.

Table 5.4 shows an example of the hierarchy that is produced when a 5-element feature set is used with respect to the feature discrimination technique. The collection

$F = \{a, b, c, d, e\}$ Set of 5-element features.

1:	Best {ac}	:	{a,b} {a,c} {a,d} {a,e} {b,c} {b,d} {b,e} {c,d} {c,e} {d,e}
2:	Best {acb}	:	{ac,b} {ac,d} {ac,e}
3:	Best {acbe}	:	{acb,d} {acb,e}
4:	Best {acbed}	:	{acbed}

Table 5.4: Example hierarchy generated using the feature discrimination technique.

of web pages W is first represented using the features in a 2-element combination (line 1). The WBD solution performance of each combination is calculated. The best combination in terms of producing the highest WBD performance score is then carried forward and used to produce the combination in the next step. Thus the best performing 2-element combination in the example is the pair {a,c} which is then used to generate 3-element candidate combinations and so on. The process is repeated until all combinations of features are contained in the set (F). The final representation contains the feature subset with the highest WBD performance, for example this could be (line 3) $\{acbe\}$.

5.5 Evaluation

This section presents the evaluation directed at the (1) identification of the most appropriate features with which to represent WWW pages with respect to the solution of the WBD problem, and (2) the 9 single feature representations were described earlier in section 5.2. These representations can be used in isolation (singles), or in various combinations (doubles, triples) to represent a collection of pages before attempting to discover the website boundaries. The fact that there are a large number of combinations that can be produced from the representations is acknowledged, and is reflected in the evaluation conducted.

The evaluation was set up as follows; firstly, an investigation into the performance of the single representations was conducted using the n-feature technique; this is presented in section 5.5.1. An evaluation that uses double and triple combinations is then presented in sections 5.5.2 and 5.5.3 respectfully; again using the n-feature technique. Comparison of the single, double and triple combinations shows that as the combinations increase, so does the WBD solution performance. This evidence leads to the intuitive notion that a composite representation of all features would provide the best WBD solution (see section 5.5.4). However, this is proven not to be the case, as use of the feature discrimination technique indicated that the "best" combinations comprised of a number of features of between one and the maximum available number (see section 5.5.5).

The remainder of this section presents an explanation of the details of how each experiment was set up and run, including the parameters used. As stated above, two techniques were used, n-feature and feature discrimination. To evaluate the performance of each of the feature representations with respect to a WBD problem, the techniques were run multiple times with a varying set of parameters. Recall that the identified parameters for both the n-feature type and feature discrimination technique were:

- FeatureSet (F)
- Clustering Algorithm
- Size Estimate (max nodes n or depth de)
- Seed Page (w_s)

To simplify the testing process and to be able to comparatively analyse the results more easily, the four RDG data sets were used (see section 4.3.2). The breadth first method of crawling the web, which is used in this work, is deterministic. This means the same web data is produced from various crawls of the same web content using the same seed page and size estimate. What is subject to change is the content of data hosted on the web itself. To remove this variation, the web data used in this evaluation was fixed. In practical terms, to do this, a cache of web data was produced by crawling the web once using the size estimate and seed page.

The techniques in this chapter focus on using clustering algorithms to produce clusters from the data. A selection of clustering algorithms were used in conjunction with the two feature techniques to produce WBD solutions, as follows:

1. Kmeans
2. Bisecting kmeans
3. Dbscan
4. KNN

Details on how they work and justification of their use was given in chapter 2.6.1.

5.5.1 Single Features

This sub-section reports on the evaluation conducted with respect to the use of single features. These were listed in section 5.2. The hyperlinks, resource, imagelinks and scriptlinks features were tested as follows: (1) "tokenised" and (2) "whole". In the first case (as indicated using labels: hyperlinks - h, resource -r, imagelinks -i and scriptlinks -s) the features were tokenised based on certain delimiters. In the second case (as indicated using labels: hyperlinks - hw, resource -rw, imagelinks -iw and scriptlinks -sw) the features were used in their entirety. The average WBD performance score for each of the features is presented in Figure 5.5. From the Figure it can be seen that the resource, imagelinks and scriptlinks features produced the best performance when they are tokenised (resource-r, imagelinks-i and scriptlinks-s). Where as the hyperlink features performed best when used as whole features (hyperlinks - hw). Figure 5.6 presents the WBD performance for each of the features with respect to all four RDG data sets. It is suggested that the hyperlink feature performed worst when tokenised because of the dense feature space of common terms that are created, in contrast to the resource, imagelinks and scriptlinks features, which are more reflective of a website boundary when this feature is tokenised. In the remainder of the evaluation reported here, only the best performing versions of the features, either tokenised or whole, were used with respect to further combinations.

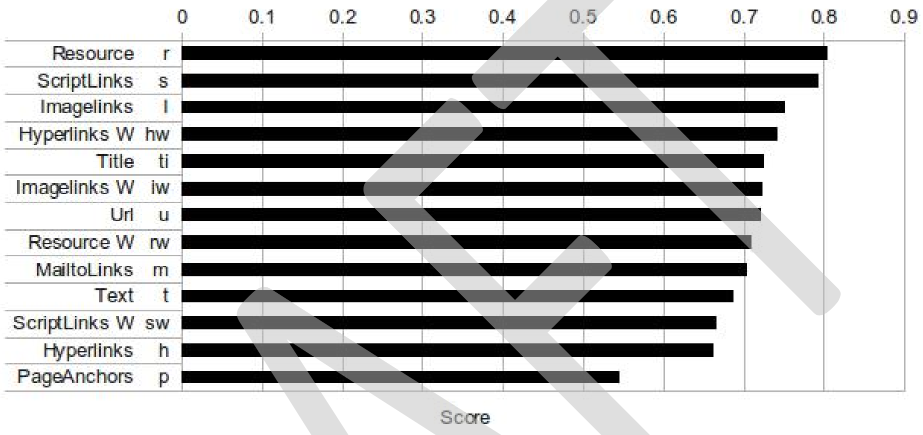


Figure 5.5: The average performance score of each feature using the four RDG data sets.

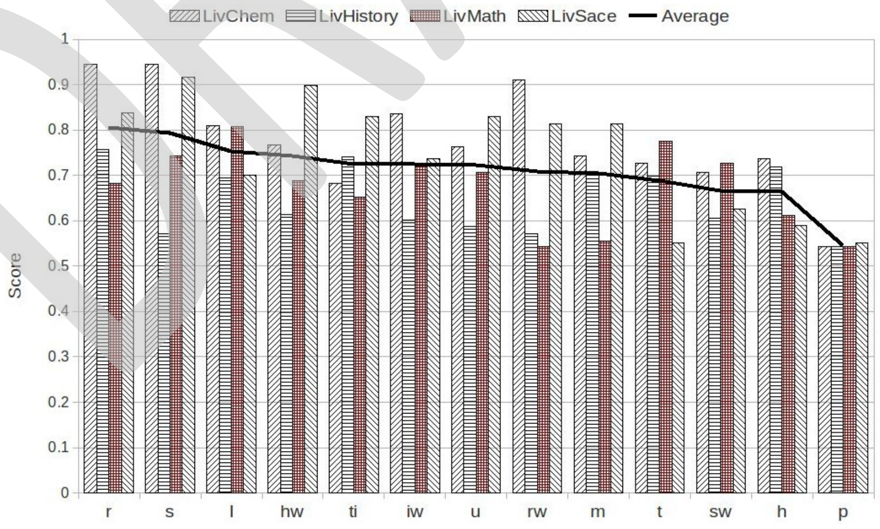


Figure 5.6: The performance score of each feature for RDGs data set.

Figure 5.7 lists the nine top performing features. From the Figure it can be seen that the top performing features are resource, script and image links. These features

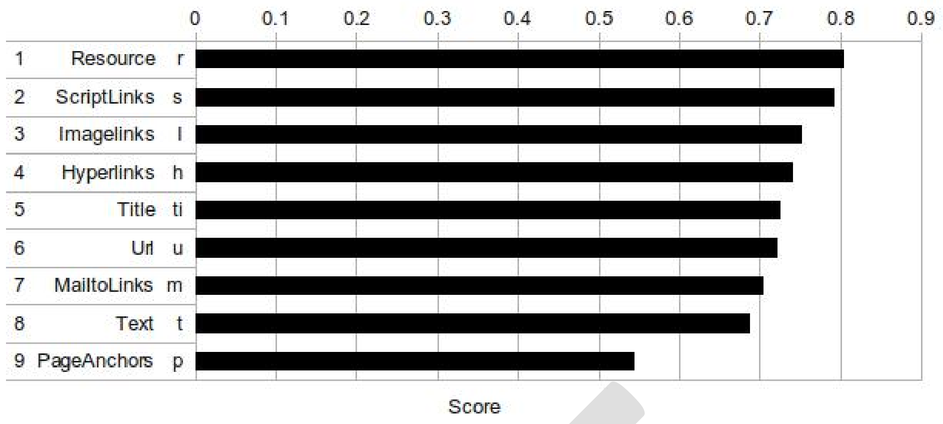


Figure 5.7: Performance of the top nine single features, ordered from best performing 1, to worst performing 9.

are intuitively indicative of the authors intent to express relationships in terms of a website. Resource and script links are used to express an underlying similar "look and feel" of pages in the same collection. It is very common for shared scripts, or blocks of code providing functionality, to be stored once, in a common location, and referred to many times by different resources. A similar practice is used in the styling of pages, which may link to common CSS or shared design elements.

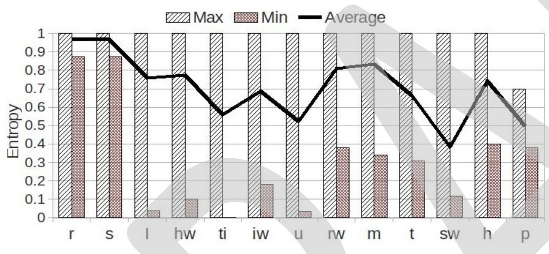
Image links can also be shared when referring to commonly used logos embedded in web pages, or content that is referred to on a regular basis with accompanying images or other media. For example if a set of pages refers to content that is regularly published and hosted on (say) `www.youtube.com/username`. A set of links in the form of `www.youtube.com/username?<video-id>` would tokenise to a set of features that refer to `youtube`, `com` and `username`. This is the reason for tokenised forms of script, resource and image links outperforming their "whole" feature counter parts.

In the case of the hyperlinks feature, when tokenised to the form used above, the feature provides a very dense feature space. This dense feature space is made up of common elements of a hyperlink, for example domain names and directories etc. The common elements can be used many times throughout a collection of pages, which can cause the distinction between pages to become blurred.

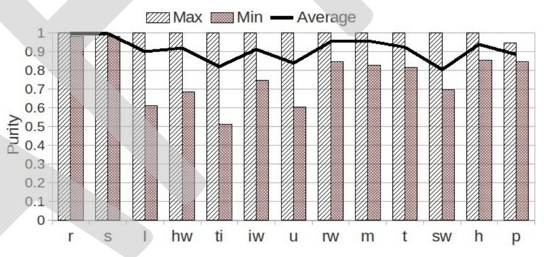
The features that do not seem to provide as good an indicator of a WBD solution are: page anchors, text and mailtolinks, as they are lower in the ranking of features as shown in Figure 5.7. Although, further experimentation using combinations of features, indicated that the presence of an under performing feature when used in combination with a high performing feature can produce an improved WBD solution. It will later be shown that features that do not perform well when used on their own can act as good discriminators when coupled with other features.

Figure 5.8 presents the values for each of the measures used to evaluate the single features. From the Figure it can be seen that a derived WBD solution can be considered a good or bad WBD solution according to the different metrics used to measure the performance, this can be due to the limited scope of the individual measurements. Looking more closely at the performance of the Page Anchors feature, which is the last column shown in the plots presented in Figure 5.8, it can be seen that the purity and recall (Figures 5.8b and 5.8c) values are high, indicating that these features produced a good WBD solution. While the remaining metrics indicate otherwise (see the last columns of Figures 5.8a, 5.8d, 5.8e and 5.8f). This phenomenon can occur when a cluster contains a small number of web pages, and these pages are target pages. In this scenario the cluster is essentially deemed as being a good website boundary, as the notion of how many target web pages existing is unknown to certain metrics. This variation of some of the metrics leads to the justification of the combined notion of an average "score" used to evaluate the performance of WBD solutions, which is used in the remainder of this chapter.

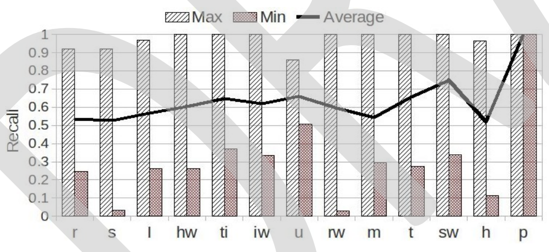
Figure 5.8: The minimum, maximum and average for each of the evaluation measures used to evaluate the single features.



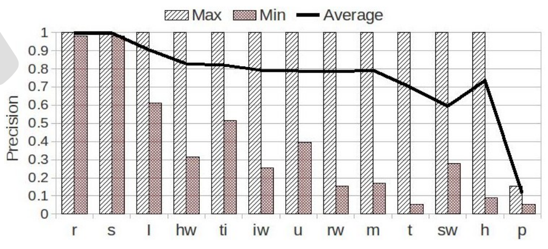
(a) Entropy



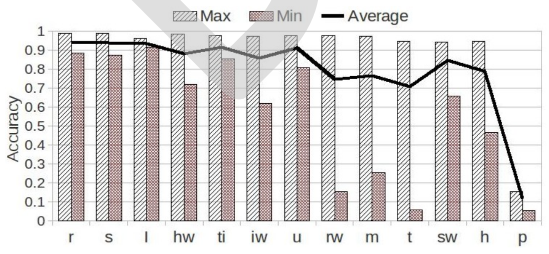
(b) Purity



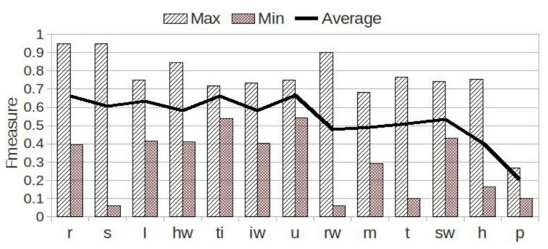
(c) Recall



(d) Precision



(e) Accuracy



(f) Fmeasure

5.5.2 Double Feature Combinations

The previous results were directed at evaluating the use of single features to produce a WBD solution. The evaluation presented in this sub-section considers combinations of pairs of features. The best performing pairs were found to be made up of combinations of a strong single feature with a weaker single feature. The evaluation shows that with respect to WBD, such pairs outperform the use of single features. The performance of each feature "double" across the four data sets is shown in Figure 5.9. The overall top performing feature combinations are shown in Figure 5.10 over all the data sets.

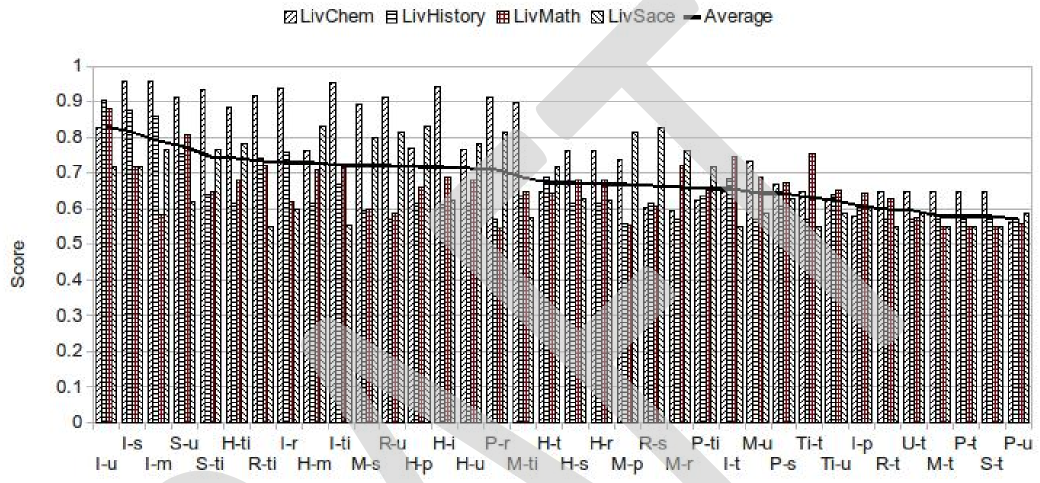


Figure 5.9: The performance of double feature combinations for RDG data sets.

With respect to the double combinations, script, image and resource links appear in the top performing combinations. In contrast to what might have been predicted, combining the top performing single features does not necessarily yield top performing combinations. The features that performed the best with respect to the single feature experiments, do however, seem to perform much better when combined with features that appear lower in performance list in the single feature analysis. The lower performing single features when used in combination with higher performing single features appear to be url, mailtolinks and title features.

5.5.3 Triple Feature Combinations

Experiments were also conducted using feature "triples". The performance of each of the triple features across the four RDG data sets is shown in Figure 5.11. The overall top performing feature triple combinations are shown in Figure 5.12 averaged over all the data sets. The results obtained with respect to the combinations of features seems to be consistent with the results obtained using doubles; under performing features appear to increase the performance of the high performing features when used in combination.

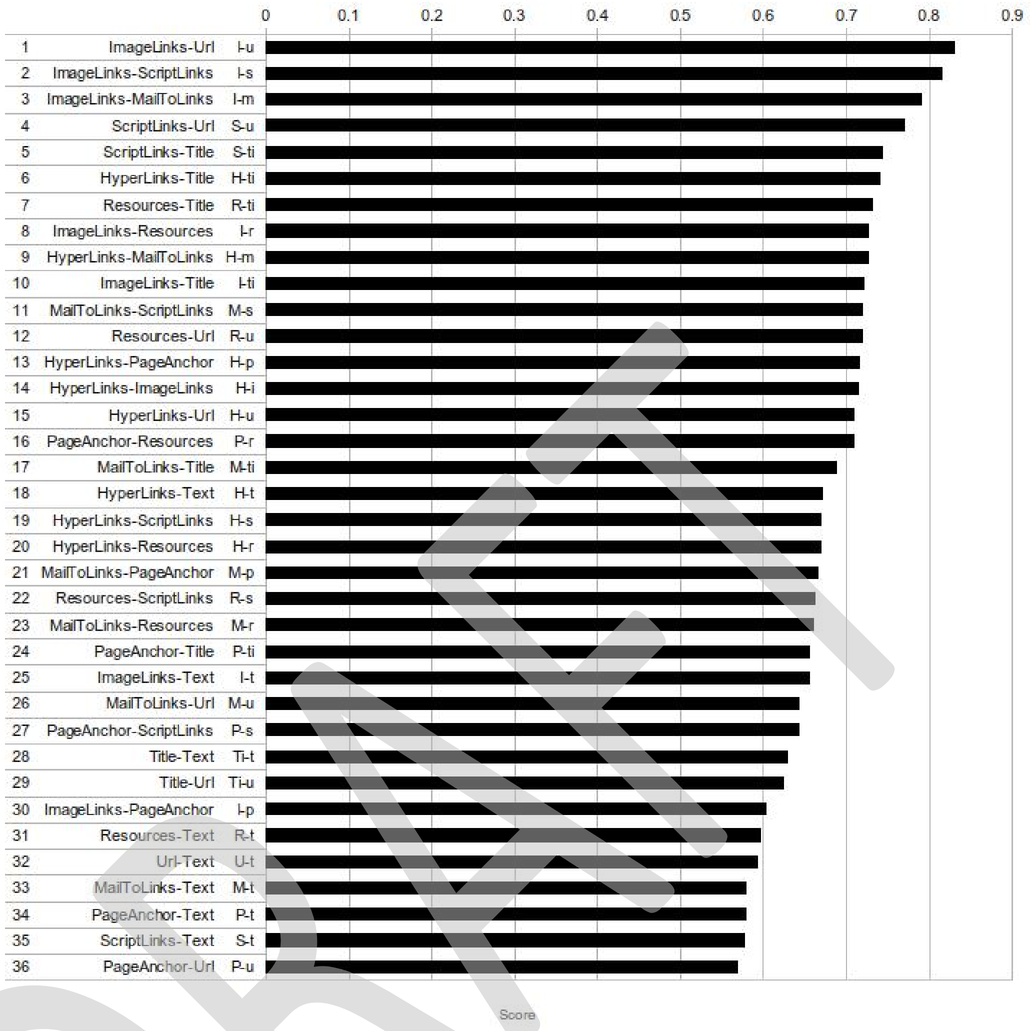


Figure 5.10: The performance of double feature combinations, ordered from best performing 1, to worst performing 36.

This can be observed from inspection of Figure 5.12.

5.5.4 Composite (All) Features

This sub-section reports on the experiments which were used to evaluate the composite of all features for each of the four RDG data sets. Figure 5.14 presents the performance of all features for each data set. The figure shows that using all features in combination produced the worst performing WBD solution. Using all features is consistently the worst performing feature combination in terms of score performance for each of the four data sets.

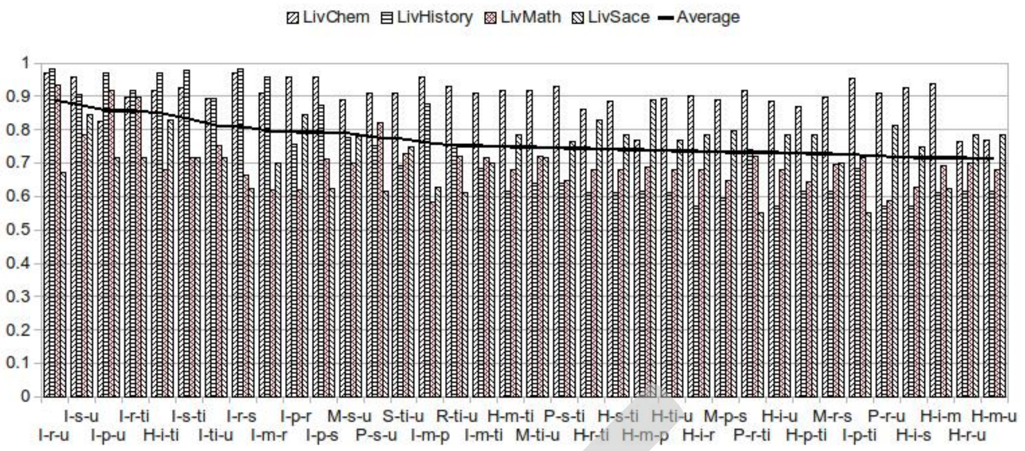


Figure 5.11: The performance of triple feature combinations for each data set.

5.5.5 Discriminated Features

The previous sub-sections evaluated the best single, double and triple combinations of features for the WBD problem. The use of these combinations, as described above, are examples of an exhaustive technique. The number of combinations increases significantly as the size of the combinations (number of elements) increases.

A non-exhaustive approach is to use the feature discrimination technique (see section 5.4.2). The aim of this technique is to produce a WBD solution using the best combination of a given set of features in terms of WBD solution produced without considering all combinations. The feature discrimination technique is used to combine features on each iteration until a composite feature vector which contains all features is produced. On each iteration, the combination that provides the biggest increase in performance with respect to the WBD solution is chosen to be the “feature set so far”.

The average result for each data set is shown in Figure 5.13, which indicates that the highest scoring number of features in a combination is 3 for LivChem and LivSace, 4 for Livhistory and 7 for LivMaths. The details of the features in each combination are shown in Figure 5.14. It can be seen that there exists a combination that gives the highest WBD score, and it is found somewhere between using a single feature, and all features. To find this “best” combination, an exhaustive enumeration of all possible combinations would have to be used, to ensure the best combination was found, if the feature discrimination technique was not used.

The features in the highest scoring combinations are consistent with the results found in previous experiments. Namely that a combination of a high and mid performing feature produces the most accurate WBD solution. The issue in finding this combination in practice is that all combinations have to be exhaustively executed and evaluated. The feature discrimination technique, however, provides a fast way of dis-

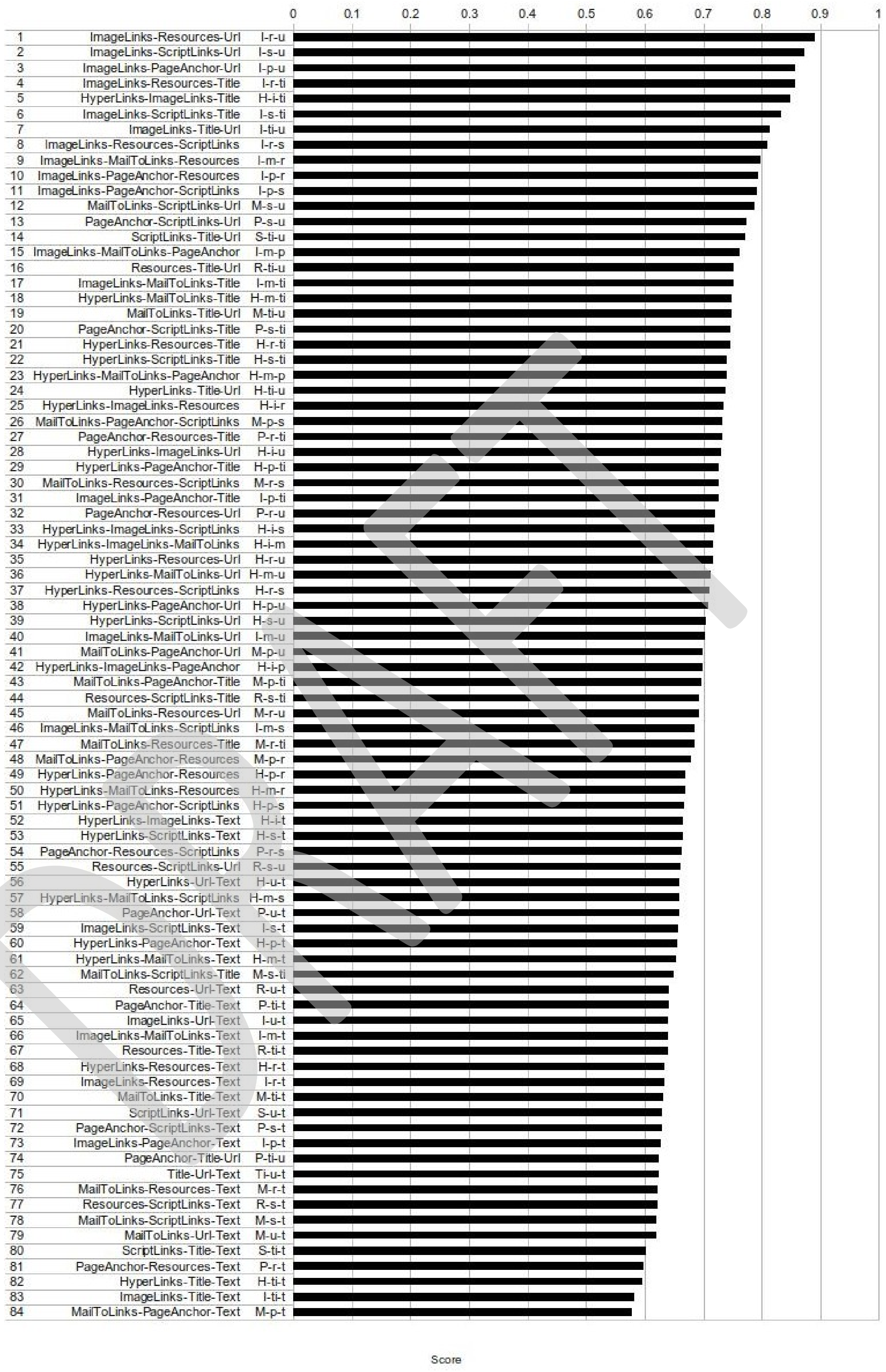


Figure 5.12: The performance of triple feature combinations, ordered from best performing 1, to worst performing 84.

covering a likely set of candidate features to provide a high quality WBD solution.

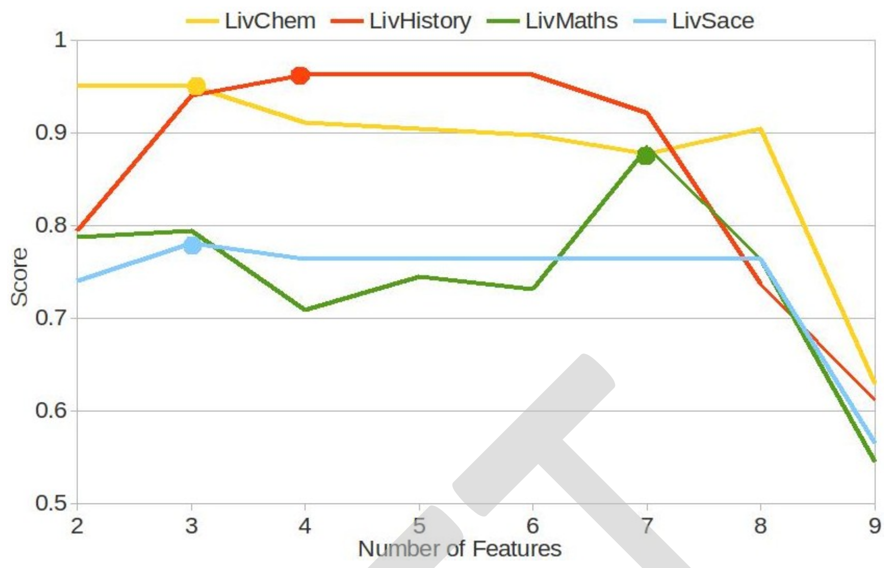


Figure 5.13: The average WBD solution score is plotted against the number of features in the combination, the highest scoring combination in the case of each data set is highlighted by a dot.

5.5.6 Comparisons

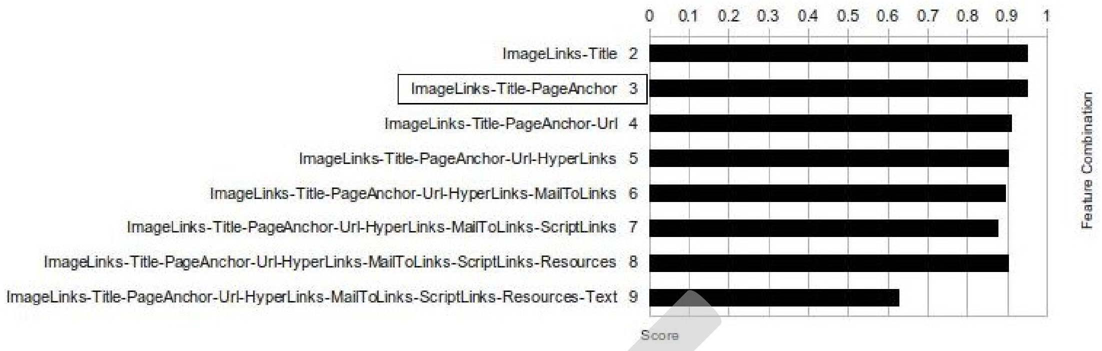
This sub-section compares the performance between singles, doubles, triples, all and discriminated combinations of features, that have been presented in the previous sub-sections. This sub-section also evaluates the performance of the four clustering algorithms with respect the WBD solutions produced using each feature combination. In this sub-section it is shown that as the number of features in a combination is increased, the top performing result increases also; as shown in Figure 5.15. In the previous single, double and triple evaluations, the top performing WBD solution was a triple feature combination, followed by a double and a single. The worst performance was recorded in the case where all features were used.

In order to establish which features performed the best, a ranking was produced based on the performance of each feature in the various feature combinations (single, double, triple, all). This is shown in Figure 5.16. The top performing features were found to be image, script and resource links; while the worst were mailto, page anchors and text.

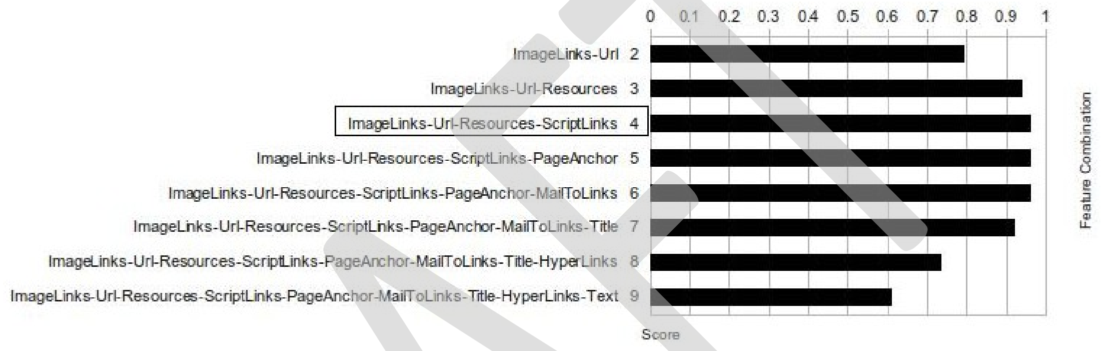
Although some lesser performing single features help to boost the performance of higher performing single features when used in combinations, the lesser performing features do not provide a good enough website boundary solution when used in isolation or in combination with other lesser performing features. This makes sense, and they appear further down the ranking of all features (Figure 5.16).

The main problems with clustering the types of feature space used in this chapter

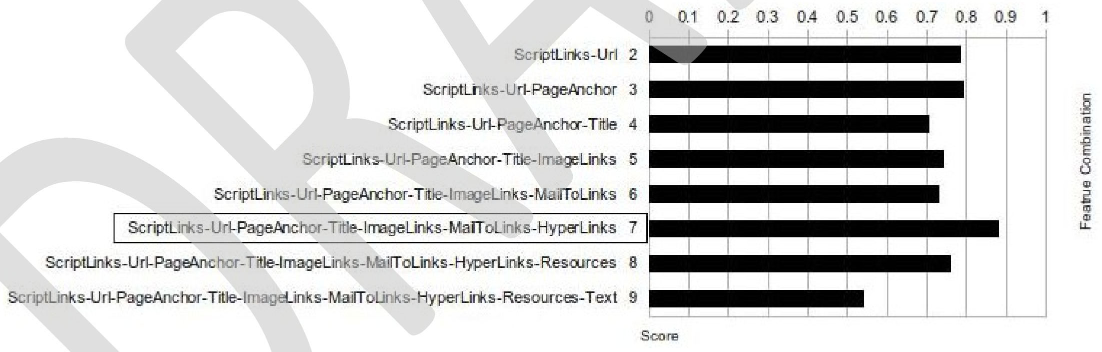
Figure 5.14: The feature combinations and average scores for each data set. The highest scoring feature combinations in each case has been highlighted.



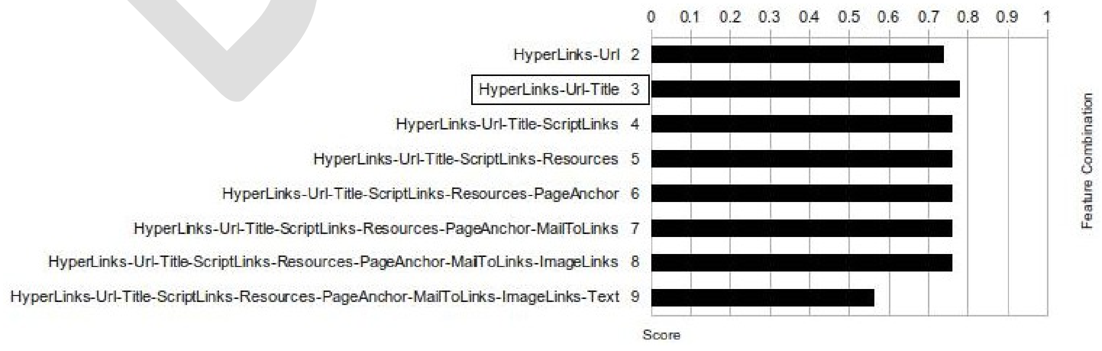
(a) LivChem



(b) LivHistory



(c) LivMaths



(d) LivSace

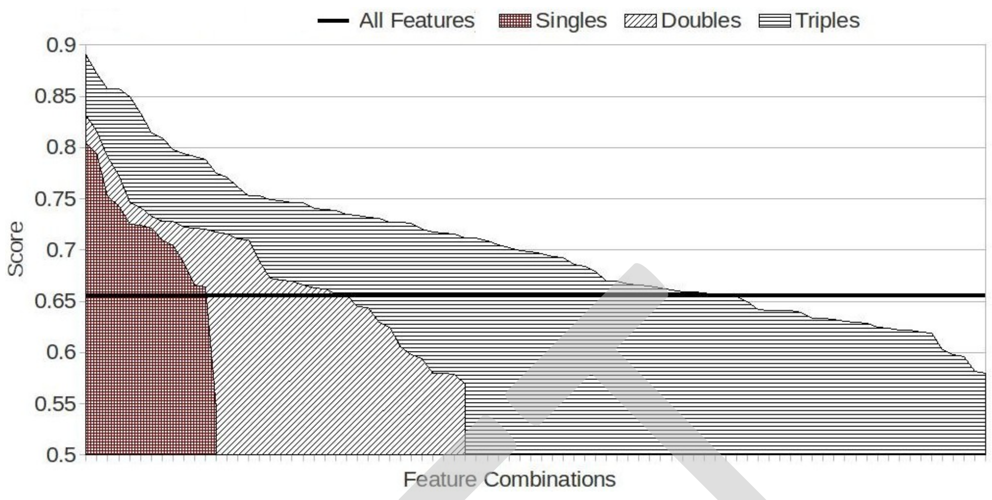


Figure 5.15: The average scores of the single, double, triple and all feature combinations. The x-axis represents the feature combination, while the y-axis illustrates the performance of that particular feature combination.

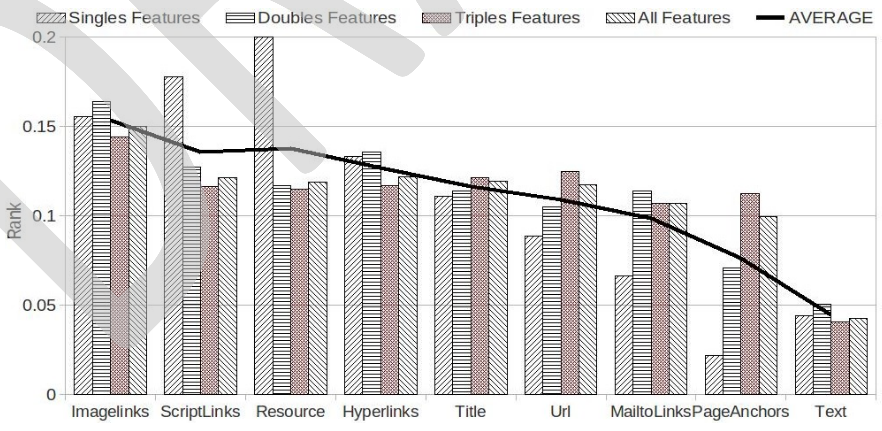


Figure 5.16: The rank of each of the features in order of best performance over singles, doubles, triples and all feature combinations.

is the so called “curse of dimensionality”. As the number of dimensions increased it became problematic to evaluate distance functions with respect to this feature space. The fact that the feature space becomes very sparse as the number of dimensions increases can be considered to be a contributing factor to the poor performance when all features are used. It has been shown that as the number of features in a combination is increased from singles, doubles to triples, the performance also increases. The intuitive thought would be that this trend of increasing performance will continue until all features are combined (the composite feature). However, it has been shown that this is not the case, using all features produces the worst performance in comparisons with the other feature configurations considered. From the results described here, it has been established that there is a high performing number of features that will perform the best; however, the only way to find this combination is to perform an exhaustive search of all combinations. The feature discrimination technique however, iteratively searches for this feature combination, and shows that it lies somewhere between 3 and 7 combinations of features with respect to the data sets used to evaluate the work described in this thesis.

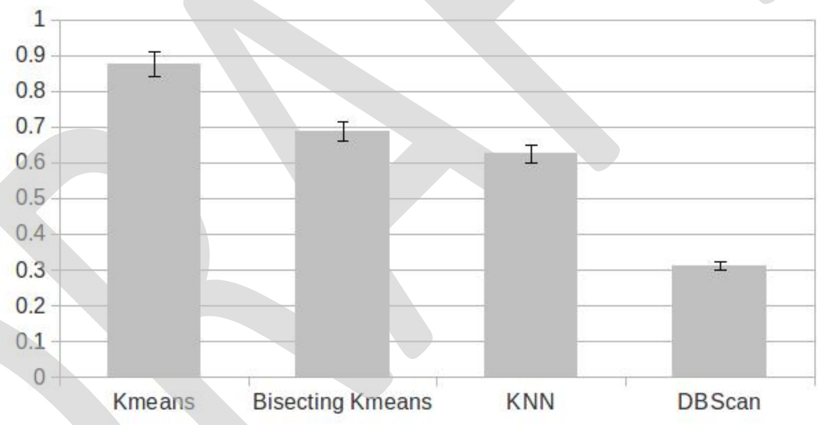


Figure 5.17: Each clustering algorithm ranked with respect to average WBD score performance for each ADG data set for the single, double, triple and all feature combinations.

Figure 5.17 shows the ranked WBD score performance of the clustering algorithms averaged with respect to the four datasets using single, double, triple and all feature combinations. Both the kmeans clustering algorithm and the bisecting kmeans algorithm require the number of clusters to be specified. In contrast, KNN and DBScan do not use such a value, instead they rely on user provided threshold values to determine a cluster configuration. In the evaluation presented in this chapter the parameters for each algorithm were systematically varied and the best performing output reported. The Figure shows that the Kmeans algorithm performs best when ranked with respect to the algorithm that produced the highest performing WBD solutions using each feature combination. It was suggested that the Kmeans algorithm outperformed the other algorithms tested due to its robust operation, allowing it to best adapt to the various feature spaces.

The kmeans algorithm used in combination with a high and mid performing feature combination (as reported in section 5.5.5) was found to be a good performing technique for WBD. It is suggested that due to the robustness of the kmeans algorithm, combined with a feature combination that is also robust can generalise to additional data sets. The robustness of the feature combination is due to the features providing a wide range of elements that can be used to capture the similarity between web pages, even in the event that certain elements are missing/ not present. The kmeans algorithm is known as a robust clustering algorithm, it lends itself to a wide variety of application, which are inherently indicative of varying feature spaces. For the reasons mentioned it is suggested that the WBD technique presented would generalise for data sets beyond that of those that were evaluated in this chapter.

5.6 Conclusions

This chapter has presented an approach to the WBD in the static context. A methodology for providing a solution to the WBD problem has been presented, and two static techniques implementing this approach to WBD have been proposed. The static methods were evaluated using four RDG data sets (section 4.3.2). Methods to model the web data, in a static context, using features extracted from web page content and associated meta-data have also been described. Using a systematic testing of combinations and single feature models, the most significant features to represent web pages in a static WBD context are shown to be script, image and resources links.

Using the clustering paradigm proposed for WBD, a number of selected clustering algorithms were applied to the static model and website boundary solutions produced. The most appropriate clustering algorithms for WBD, in the static context, were found to be kmeans and dbscan.

The static approach to WBD is shown to have two main issues. Firstly, it involves using a size estimate to gather a collection of pages prior to performing any modelling or analysis. This size estimate has to exceed the amount of pages that are estimated to be within the website boundary, and thus inherently gathers a disproportionate amount of noise to target pages. The second issue is the cost of processing, associated with each of the data items, which increases as the presence of noise increases. When this is combined with an unknown set of attributes used to model a web page, the discovery of a WBD solution can become computationally expensive. Because of these issues it can take a substantial amount of processing resources to discover website boundaries in the static context. The next chapter presented an approach, again in the static context, that concentrated on the hyper-linked structural properties of the graph to generate WBD solutions.

Chapter 6

Static Technique 2: Graph Structure Partitioning

This chapter presents the second proposed solution to the WBD problem considered with respect to the research described in this thesis. More specifically the research presented in this chapter reports on two graph partitioning approaches to investigate if the structural properties of a web graph can provide accurate solutions to the WBD problem. The graph partitioning approaches concentrate on segmenting (dividing) a web graph solely according to the structural connectivity of the hyperlinks of the graph. Both of the methods presented in this chapter are considered to be static WBD methods, as they both require the entire web graph apriori. In order to justify the usage of graph partitioning methods, an assumption has to be made about the structure of a website. The assumption is that a collection of pages making up a web site is in some way more structurally cohesive than the connected noise pages, and that this property is encoded and detectable in the web graph, thus making it possible to provide a solution to the WBD problem using graph partitioning approaches.

The first of the two graph partitioning approaches presented in this chapter is a hierarchical graph clustering approach based on the Newman algorithm [144, 143]. The second is a minimum cut (mincut) approach based on the Ford Fulkerson algorithm founded on the *mincut-max flow* theorem [83, 84], which in turn is founded on the concept of flow networks.

The results show that it is indeed possible to provide a representative WBD solution using graph partitioning approaches. The mincut method of graph partitioning yields the best WBD solution with respect to the experimental analysis performed. A major down side of the mincut approach, however, is that it is not clear, at least in the initial stages, which vertices are best chosen to perform a mincut of the graph. As a result the approach provided in this chapter is an exhaustive approach, which has to iterate over all combinations of vertices.

The rest of this chapter is organised as follows. Some background theory on hierarchical graph partitioning and details of the Newman algorithm are presented in section

6.1. This is followed by a report and analysis of the experimental results obtained during evaluation of the hierarchical graph approach (section 6.2) and a summary. An explanation of flow networks is given in section 6.3, the explanation includes a description of the Ford Fulkerson algorithm. The analysis of the experimental results obtained during the evaluation of the minimum cut method is presented in section 6.4. Some conclusions drawn from the work described in the chapter are presented in section 6.5.

6.1 Hierarchical Graph Partitioning

This section describes the hierarchical graph partitioning method founded on the Newman method [144, 143] of hierarchical graph clustering. It uses an agglomerative hierarchical approach to produce clusters in a graph, based on the similarity of graph vertices. The algorithm produces k graph clusters, where k is determined by the algorithm itself (no need for user input).

Hierarchical graph partitioning has been used with respect to various applications to detect communities in networks [89], such that clusters of high inter-connectivity are grouped together while the connection between these clusters remains low. The algorithm will first be explained below, and then the experimental results obtained from the application of the technique to the WBD problem will be presented.

6.1.1 Newman's Modularity

This sub-section explains the hierarchical graph clustering method for identifying clusters in graphs based on the work by Newman [144, 143] centred around a modularity value Q. Hierarchical clustering is conducted by merging or dividing clusters in a top down (divisive) or bottom up (agglomerative) manner. The Newman method is an agglomerative method. This sub-section is commenced with some background as a precursor to the following explanation of the calculation of the modularity (Q) of a graph.

Given a directed graph $G = (V, E)$ such that V is the set of all vertices $V = \{v_1, v_2 \dots v_k\}$ and E is the set of all edges (v_i, v_j). The total number of edges is denoted as M ($M = |E|$). A cluster is defined as $g = \{v_1, v_2 \dots v_i\}$ such that $g \in G$. The total number of clusters for a given graph is denoted by k.

A function can be devised, $countEdges(s_i, s_j)$, that counts the number of edges from the set of vertices in s_i to the vertices in s_j. Let e_{ij} equal the fraction of edges from cluster g_i to g_j in a graph G.

$$e_{ij} = \frac{countEdges(g_i, g_j)}{M} \tag{6.1}$$

Thus e_{ij} represents the fraction of intra-cluster edges, with respect to two given clusters i and j, in terms of the total number of edges M in graph G. Note that e_{ii} is the fraction

of intra-cluster edges in a single cluster g_i.
Let T equal the total edges inside all clusters k, such that:

$$T = \sum_{g_i=1}^{g_i=k} e_{ii} \tag{6.2}$$

This value is essentially the sum of inter-cluster connections for all clusters.

Let a_i be the fraction of edges within graph G that end in the vertices of cluster i.

$$a_i = \sum_{j=1}^{j=k} e_{ji} \tag{6.3}$$

The value a_i^2 is the fraction of edges in cluster i if edges where connected uniformly at random in graph G. This value is simply calculated as $a_i^2 = (a_i)^2$.

The modularity Q is a measure of how "good" a certain cluster configuration is when imposed on G and is calculated according to the inter-connectivity of vertices in each individual cluster. The process of calculating Q proceeds by comparing the connectivity within each cluster division to the connectivity if the records in each cluster were connected randomly [145]. The modularity measure is thus an indicator of whether the proposed cluster configuration takes suitable account of the actual connectivity between records as opposed to some random connectivity. The assumption is that the proposed cluster configuration has been made based on decisions to merge clusters with high inter-connectivity, thus better than merging clusters based on random connectivity.

$$Q = \sum_{i=1}^{i=k} (e_{ii} - a_i^2) \tag{6.4}$$

The modularity value, as defined above, is essentially a measure of how good a cluster merge is in terms of how separated the vertices are with respect to the graph structure. Note that if all vertices of graph G are placed into a single cluster the value of Q will be 0, thus indicating a (very) poor clustering. Experimental results in [145] show that a value of $Q = 0.3$ or greater indicates that meaningful clusters (communities) exist in the target graph G (otherwise it is deemed that no meaningful clusters exists within the input data).

6.1.2 The Newman Graph Clustering Algorithm

The Newman clustering algorithm is an iterative hierarchical clustering method applied to graph networks (such as social networks); the aim being to generate a cluster configuration P comprised of clusters. Thus $P = \{\{g_1\}, \{g_2\} \dots \{g_k\}\}$, where each g is an individual cluster containing one or more vertices. The pseudo code for the algorithm is shown in Table 6.1. The input to the algorithm is a configuration P generated by placing each vertex in V in its own cluster. Thus on start up $|P| = |V|$. On each iteration a pair of clusters (g_i and g_j) from all possible combinations of clusters in P is selected to be joined. The selection is made by maximising the Q value (calculated as described above). The algorithm iteratively merges combinations of clusters so that either: (i) the modularity value (Q) is maximised for a cluster configuration, or (ii) a single cluster containing all input vertices V is reached. The resulting cluster configuration produced by the algorithm is thus the cluster configuration that has the highest value Q associated with it.

```
Algorithm hierarchical_clustering ($P$)
  $P'$={};
  $\theta'$ = Min_Value;
  repeat
    $\rho$={};
    $\theta$= 0;
    while moreCombinations($P$) do
      tmp = nextCombination();
      if Modularity(tmp)> $\theta$ then
        $\theta$ = Modularity(tmp);
        $\rho$ = tmp;
      end if
    end while
    $P' = \rho$;
  until $\theta < \theta'$
  return $P'$;
```

Table 6.1: Iterative Hierarchical Clustering algorithm based on Newman [144, 143].

6.1.2.1 Example of Newman Hierarchical Clustering

In this subsection Newman's hierarchical graph clustering method is illustrated by applying it to cluster an example graph of the form given in Figure 6.1. The cluster joins performed at each iteration of the algorithm are illustrated by the dendogram given in Figure 6.2b. Given a graph $G = (V, E)$ (Figure 6.1) with 4 vertices (labelled 0,1,2 and 3) and 5 edges, the initial cluster configuration will be $P = \{\{0\}, \{1\}, \{2\}, \{3\}\}$ (which has a Q value of -0.07). On termination of the process the highest value for Q is 0.28 to give cluster configuration $P = \{\{01\}, \{23\}\}$ (see Figure 6.2a). For an illustration of each iteration of Newman's algorithm using an example graph see Appendix C.

6.1.2.2 Application of Hierarchical Clustering Based Method to WBD

The application of Newman's clustering algorithm to a graph results in an output set of clusters. The number of clusters is not predetermined, it is derived by the algorithm according to the structure of the input graph. Thus the number of generated clusters

Figure 6.1: An example graph.

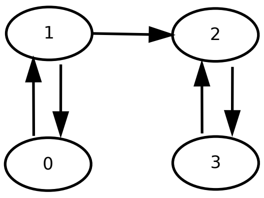
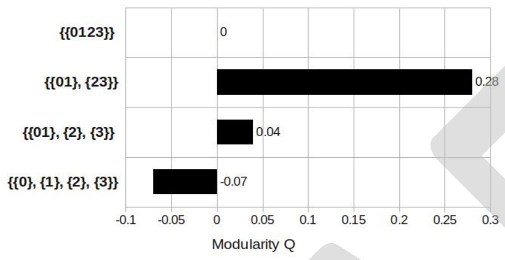
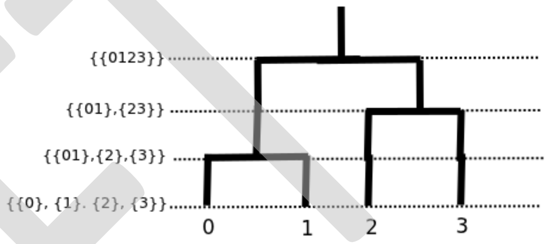


Figure 6.2: The Q values (a) and corresponding dendogram (b) for each iteration of Newmans hierarchical graph clustering algorithm with respect to the example graph given in Figure 6.1.



(a) Q values



(b) Dendogram

can not be predicted in advance. Application of Newman's algorithm to the WBD problem has similarities with the previous methods described in this thesis in that the number of clusters produced varies (recall the output of some of the clustering algorithms used in chapter 5). In the input graph there is a labelled seed page w_s. This seed page is located in one of the output clusters, this cluster is known as the target cluster K_T. The remaining clusters, regardless of their number, are aggregated and subsequently known as the noise cluster K_N. To produce a high performing WBD solution, given an input graph, the aim is to produce a graph partitioning such that the target cluster consists of a high number of target vertices C_T and a low number of noise vertices C_N (recall the description and notation from section 3.4).

6.2 Evaluation of Hierarchical Graph Partitioning Approach

Newman's hierarchical graph clustering method was used to partition each of the RDG data sets (described in section 4.3) in order to produce WBD solutions which were then evaluated. As already noted the proposed graph partitioning method is a static WBD method, this means that a snapshot of the web is needed apriori. As explained above, the method generates a complete "best" cluster configuration over all vertices, therefore all vertices to be considered have to be available "ahead of time".

The evaluation was conducted using two scenarios. The first was directed at measuring the performance of the Newman hierarchical graph clustering method on the complete set of snapshots of the RDG data sets. The second was directed at measuring the performance of the algorithm on varying sized snapshots, containing different amounts of noise and target web pages. The objective was to mimic the way that different web crawling strategies might collect the initial data. It was expected that, depending on the nature of the web crawling strategy that might be adopted, different quantities of noise web pages would be included. It will be shown that using a smaller snapshot size, than the size of the complete snapshot, produced the best in terms of providing a WBD solution.

6.2.1 Complete Snapshots

The performance of the Newman graph clustering method, using the complete snapshot of the data, is indicated by the bar charts presented in Figure 6.3. From Figure 6.3d it can be observed that the performance score is consistent for LivHistory, LivMaths and LivSace; while LivChem shows a higher score. This is also true for both the accuracy and Fmeasure values (Figure 6.3a). In contrast, the recall and purity values are consistently high for all four data sets (Figure 6.3b and 6.3c).

Inspection of the hierarchical clustering process reveals more detail about how the algorithm has segmented the graphs and produced the indicated performance values. By observing the composition of the clusters as the process proceeds (Figure 6.4) it can be seen that most of the target items are grouped with one another for all four data sets, hence the similar recall and purity values. However, the target items are grouped together with a lot of noise items, which produces overall low performance score values. The exception to this is for the LivChem dataset, where target items (pages) are grouped together with a little amount of noise contained in the same cluster.

The modularity value (Q) produced as the Newman algorithm proceeds gives an insight into the process of merging clusters (as illustrated in the Figure 6.5). As the algorithm merges clusters, the modularity value improves dramatically in the initial stages. This is because, as clusters are combined, the divisions of the graph are exhibiting an increasingly higher inter-connectivity. The steady increase of Q then tails off, and peaks at which point the "best" clustering configuration has been arrived at. In practice, the stopping criteria is the observation that the value of Q decrease from one iteration to the next. This consistently happens after a significant amount of cluster merges have been executed, compared to the total number of merges possible for each data set. Around 90% of the total merges possible are made on average for each case (LivChem 93.66%, LivHistory 92.41%, LivMath 85.98% and LivSace 87.28%). This requires a significant amount of resource while only yielding mediocre WBD solutions (as in the case of LivHistory, LivMath and LivSace).

The criteria for merging clusters in the algorithm is based on optimising the modularity value (Q) on each iteration. The effect this has on cluster merges is that the algorithm favours cluster merges that have a high number of edges between them. The algorithm essentially removes edges "between" communities, which results in these clusters being merged [89]. By selecting such clusters for a merger, the modularity value improves as the edges are now contained within a cluster, and not across clusters, thus increasing the inter-connectivity. In general, if two vertices have a high number of edges between them, these will be merged in the early stages of the algorithm.

Intuitively the hierarchical approach seems like a good strategy for WBD. If web pages from a website are assumed to link to each other often, then these vertices will be merged, thus creating dense clusters of highly connected and related pages representing a website. What this does not account for is highly connected noise pages which tend to also be merged first. This can subsequently create a scenario where a noise cluster has many links to the target cluster because either: (i) the noise items increasingly point to target web pages, as parts of the noise cluster is dense, thus increasing the probability of links to target pages or; (ii) the target pages link to many separate noise pages, which subsequently fall into the same cluster as target pages because of a high frequency of links between the target cluster and the noise pages. Both of these scenarios cause clusters of noise pages to be merged with target pages and vice versa. This characteristic is exhibited in the data sets used, and is a contributing factor to the adverse WBD solutions produced.

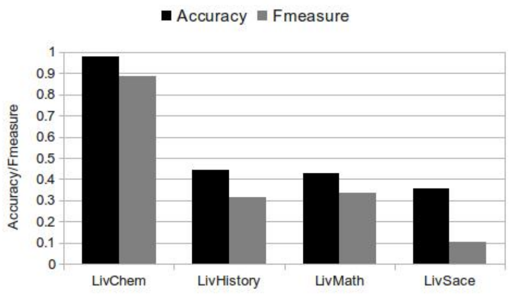
Therefore the proposed hierarchical clustering approach does not produce significant WBD solutions with respect to the RDG data sets tested. The additional set of experiments described below were directed at using various sized portions of each of the data sets. The aim was to evaluate the effect that different sized portions of the web graph might have on the WBD solutions produced.

6.2.2 Varying the "Snapshot" Size

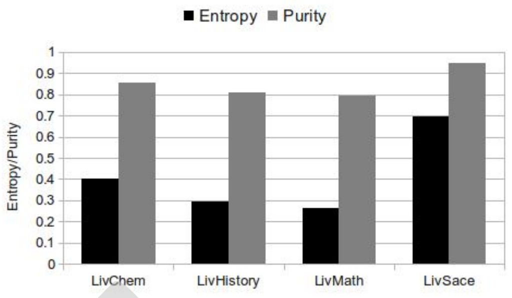
A systematic approach was used for the set of experiments that tested the hierarchical graph clustering approach with varying sized snapshots of the data sets. The testing was comprised of a number of rounds. Each round used a different sized snapshot of the RDG data sets. In the first round, the first 50 vertices, in breadth first order, from the seed page (w_s) were used as input to the algorithm. The number of vertices was increased in steps of 50 for each subsequent round, until a set equivalent to the "complete snapshot" used in the previous set of experiments was arrived at.

The results produced demonstrated that there were variations in the WBD solutions produced for each of the differing sized snapshots (Figure 6.6). From the results the most appropriate snapshot size is considered to be that associated with the solution that has the highest WBD performance over the various sized snapshots considered.

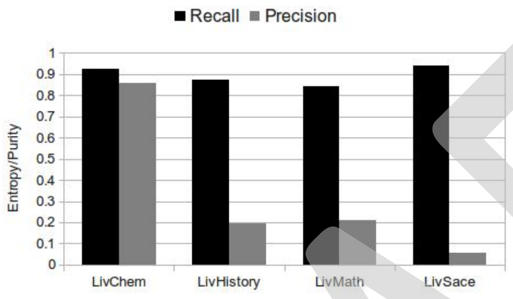
Figure 6.3: The performance of the hierarchical graph clustering algorithm on the four test datasets (LivChem, LivHistory, LivMath and LivSace).



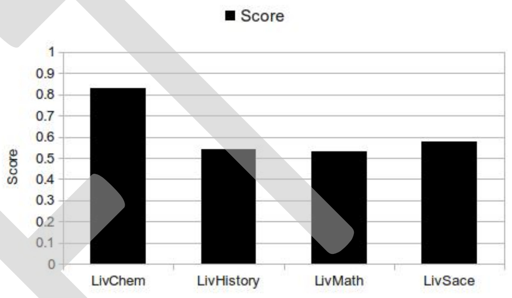
(a) Accuracy/Fmeasure



(b) Entropy/Purity



(c) Recall/Precision



(d) Performance Score

Figure 6.4: The compositions of the output clusters generated using the Newman algorithm for all four data sets. The tables show how many clusters were produced and the total number and target pages (vertices) in each of the clusters. The cluster containing the seed page (w_s) is the target cluster K_T.

	C1	C2	C3	C4	C5
Total	251	21	84	3	69
Target	0	0	0	0	59 (w_s)

(a) LivChem

	C1	C2	C3	C4	C5	C6	C7
Total	15	15	15	79	7	291	2
Target	0	1	0	0	0	56 (w_s)	0

(b) LivHistory

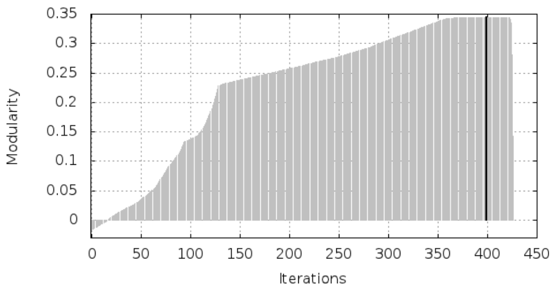
	C1	C2	C3	C4	C5
Total	11	86	307	12	7
Target	0	0	64 (w_s)	0	0

(c) LivMath

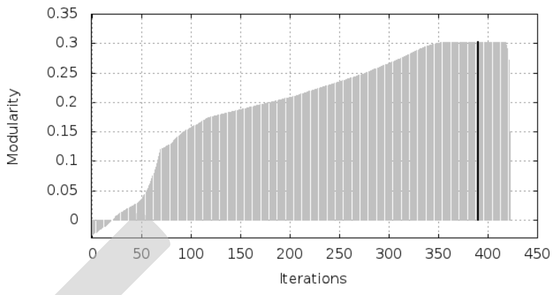
	C1	C2	C3	C4	C5
Total	16	279	15	100	1
Target	0	15 (w_s)	0	1	

(d) LivSace

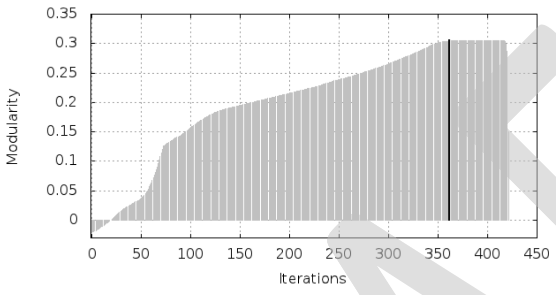
Figure 6.5: The modularity Q values obtained for each iteration of the Newman algorithm for all four datasets (LivChem, LivHistory, LivMath and LivSace). Iteration 0 has each vertex in an individual cluster. In the final iteration all vertices are in a single cluster. The dark line indicates the highest Q value, and subsequently the "best" cluster configuration, in each case.



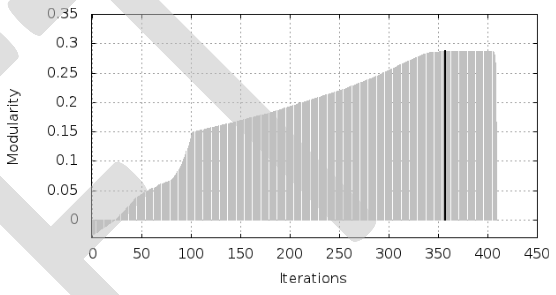
(a) LivChem



(b) LivHistory



(c) LivMath



(d) LivSace

Figure 6.6 shows the performance for all sized snapshots (x-axis represents snapshot size), while Figure 6.7 reports on the cluster compositions of the best solution.

As the snapshot size was increased, it was found that there was a corresponding improvement in the context of the target web pages grouped with the seed page. This is shown by the high recall values produced (Figure 6.6e). This demonstrates that the Newman algorithm is partitioning the graph in such a way that target pages are not being allocated to too many differing clusters, but are in fact being mostly grouped together. In contrast to the high recall values recorded, the precision values (Figure 6.6f) indicate that the ratio of target to noise pages grouped with the seed page is increasingly containing more noise pages, as the precision value decreases. This indicates that the algorithm is partitioning the graph in such a way that pages from the website are grouped together; but, as the noise increases (as snapshot size increases), the algorithm finds it increasingly difficult to distinguish noise pages from target pages.

As previously mentioned, the Newman algorithm does not have a predetermined number of output clusters (the number is determined by the algorithm during the computation). The number of clusters produced does not represent the two cluster labelling associated with the data (target pages contained inside the website boundary C_T and irrelevant/ unwanted noise web pages C_N). The number of clusters finally generated is never lower than 5 with respect to each test run where the snapshot size was varied (Figure 6.8). This suggests that the algorithm discovers a natural partitioning that is influenced by the underlying structure of the web graphs in each case. The cluster composition of the best results gives an insight into how the algorithm is partitioning the graph (Figure 6.7). The algorithm outputs a number of clusters, a selection of these clusters potentially may hold “outliers” contained in the data. This is shown in Figure 6.7d where clusters c3-c8 each contain a single noise web page.

The most appropriate WBD solution associated with each data set is not produced using a single particular snapshot size (in terms of number of vertices/pages). What is consistent is that the best WBD solution is arrived at using a smaller size snapshot than the complete snapshot. This intuitively indicates that reducing the amount of noise improves the WBD solution using the proposed hierarchical graph clustering approach based on the Newman algorithm.

As a direct comparison the best results indicate a significant improvement over the WBD solutions generated using complete snapshots (Figure 6.9). What this suggests in terms of the deployment of the Newman algorithm for WBD is that an improved crawl strategy to reduce the number of noise pages included in the snapshot will significantly improve the WBD solutions produced.

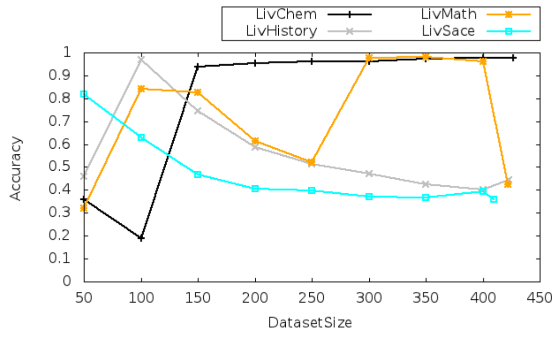
6.2.3 Evaluation Summary

The results for the hierarchical graph partitioning approach based on the Newman graph partitioning algorithm using the complete snapshot produced mediocre results, the exception to this was its application to the LivChem dataset. The contributing factor for the adverse results produced by the algorithm was the merging of highly connected clusters that were unrelated in terms of the WBD problem. The identified target cluster, generated by the algorithm was a grouping that contained a high number of target pages, but also contained a high proportion of noise pages.

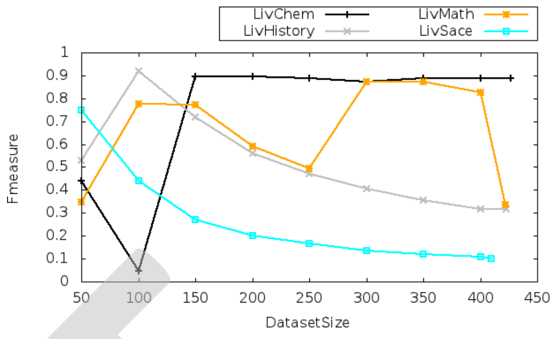
To further investigate the performance of the Newman algorithm varying sized snapshots were used, in contrast to the complete snapshot used in the initial evaluation. The varying sized snapshots of the web represented portions of the web that might be collected using different crawl strategies designed to minimise noise page collection. Using the Newman algorithm on such snapshots highlights the potential performance for the algorithm given a different snapshot of the web graph.

The results show that the best WBD solutions were found when the algorithm was applied using smaller sized snapshot than the complete snapshot. This is evidence that, in combination with a differing crawl strategy, the hierarchical graph clustering does have the potential to produce representative WBD solutions.

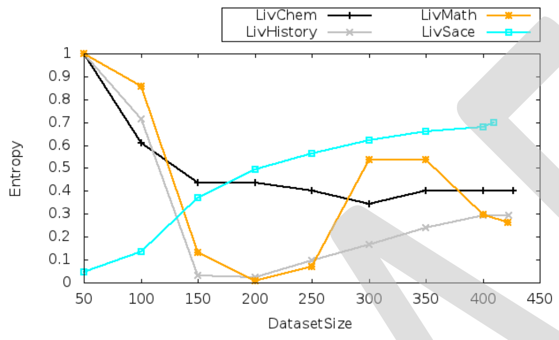
Figure 6.6: The results of running the Newman hierarchical clustering algorithm on the RDG datasets (LivChem, LivHistory, LivMath and LivSace).



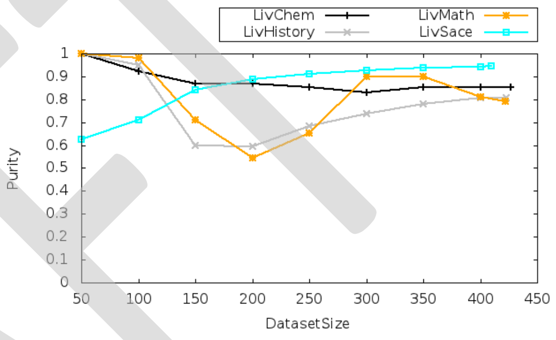
(a) Accuracy



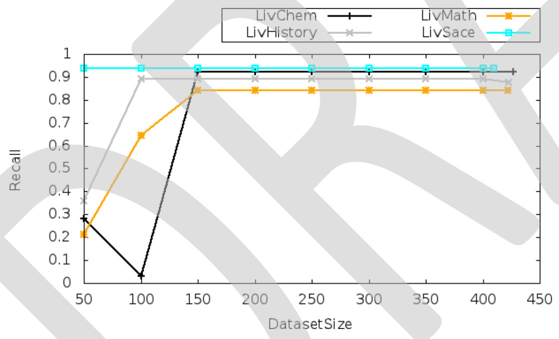
(b) Fmeasure



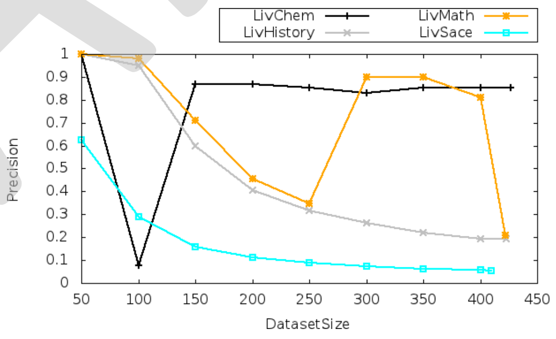
(c) Entropy



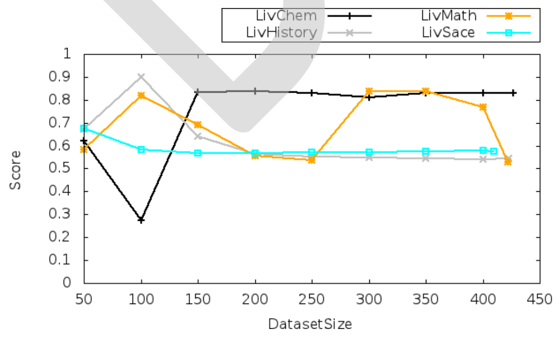
(d) Purity



(e) Recall



(f) Precision



(g) Score

Figure 6.7: The cluster compositions of the best results from all tests that applied the Newman algorithm on varying sized snapshots with respect to the four data sets (shown in Figure 6.6). The size of the snapshot in each case is indicated. The tables show how many clusters were produced and the total and target number (C_T) of vertices in each of the clusters. The cluster containing the seed page (w_s) is labelled as the target cluster K_T.

	C1	C2	C3	C4	C5	C6	C7
Total	3	3	2	68	71	52	1
Target	0	0	0	59 (w_s)	0	0	0

(a) LivChem (200 snapshot)

	C1	C2	C3	C4	C5	C6	C7	C8	C9
Total	60	3	31	1	1	1	1	1	1
Target	57 (w_s)	0	0	0	0	0	0	0	0

(b) LivHistory (100 snapshot)

	C1	C2	C3	C4	C5	C6	C7
Total	16	15	71	190	56	1	1
Target	0	0	64 (w_s)	0	0	0	0

(c) LivMath (350 snapshot)

	C1	C2	C3	C4	C5	C6	C7	C8
Total	24	20	1	1	1	1	1	1
Target	15 (w_s)	0	0	0	0	0	0	0

(d) LivSace (50 snapshot)

Figure 6.8: The number of output clusters from each round of applying the Newman algorithm on varying sized snapshots with respect to the four data sets. The plot shows the number of clusters (y-axis) for each snapshot size (x-axis), the average number of clusters is also shown.

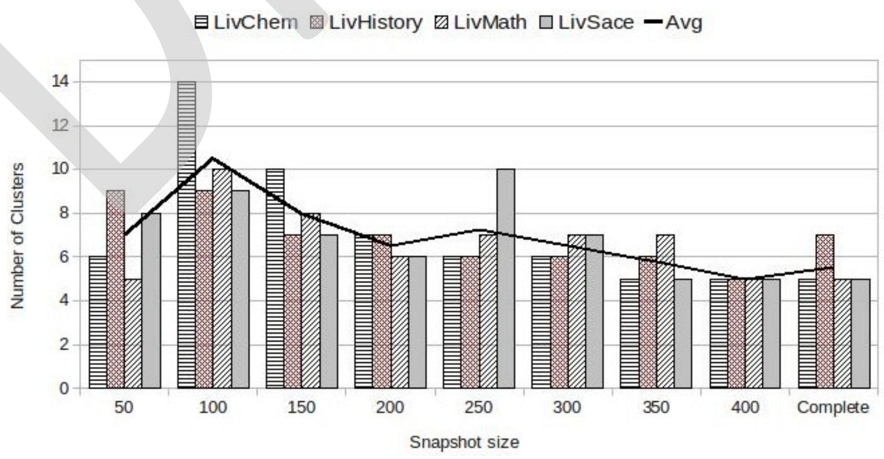
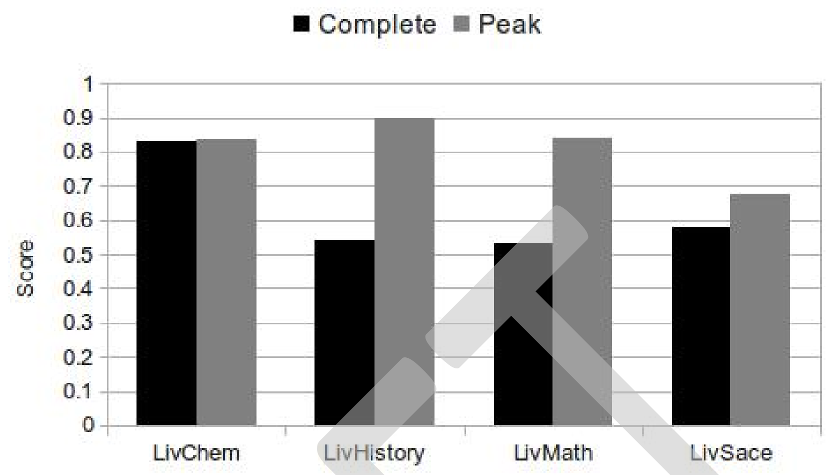


Figure 6.9: Comparison of the results of the application of the Newman algorithm to the complete snapshot and the best result obtained from varying the snapshot size. The plot shows the WBD solution performance score of both the best and complete snapshot results (y-axis) for each data set (x-axis).



6.3 Flow Networks

In graph theory, a flow network is a graph of vertices and edges where edges can have a *flow* associated with them, an indicator of *material* "flowing" between vertices. Flow networks can be used to model a variety of problems [23, 87]: in the context of the work described in this thesis they can be used to identify "cuts" (divisions) within a network in order to produce graph partitions [164].

A "minimum cut" is a cut that splits a network into two partitions such that a "minimum" number of edges are cut, thus forming two clusters comprising dense (or at least denser) connections [164]. The minimum cut of a network can be found effectively by applying the Max-Flow Min-Cut theorem. [57][83].

In this section the approach to addressing the WBD problem using a flow network to model the web graph is described. The process takes an input network, which in this case is a portion of the web, and creates a flow network based on this input. Using the flow network, a corresponding *residual network* (see section 6.3.1.1) is also created to keep track of how the flow is permitted to move across the network using the notion of *augmenting paths* (see section 6.3.1.2). Using this model cuts of the network can be performed (see section 6.3.1.3); in the case of the work described in this thesis, the Max-Flow / Min-Cut theorem was used with respect to the input graph to achieve the desired partitioning (see section 6.3.1.4).

The Ford Fulkerson method (section 6.3.2) is an algorithm used to discover the maximum flow in a network, in the work described in this thesis this algorithm is used to implement the Max-Flow / Min-Cut Theorem to find the minimum 2-cut of the network such that an output of $k = 2$ clusters is produced. The partition of the flow network is then transposed to the input network so as to provide a solution to the WBD problem.

6.3.1 Flow Network Model

The concept of a flow network model is formally described below, the description and examples are based on [57]. A flow network is a directed graph $G = (V, E)$ such that V is the set of all vertices, and E is the set of all edges. An edge between two vertices u and v ($u, v \subseteq V$) is indicated using the notation (u, v). Each edge (u, v) has an associated non-negative capacity c defined as $c(u, v) \geq 0$, and can also have an associated flow f of "material" from u to v ($f(u, v) \geq 0$). The value of the flow cannot exceed the capacity of an edge (see below). This model can be conceptualised as a network of water pipes, where by the *capacity* is constrained by the diameters of the pipes, thus restricting the amount of water that can flow between any two points.

One vertex of the flow network is considered to be the source $s \in V$ and another vertex is considered to be the sink $t \in V$; "material" is then considered to "flow" through the network from the source to the sink. For a graph G to be (typically) considered to be a flow network, some constraints about its structure are made as follows:

Disallowed reverse edges: If an edge (u, v) exists, then there must be no edge (v, u) that exists in the reverse direction.

Disallowed self loops: A vertex v cannot have an edge of the form (v, v) (an edge which points back to its self).

Typically a flow network will not have any edges into the source node s from other vertices in the set V as this would not be compatible with the idea of material flowing from the source to the sink. A path p in a flow network is simply an ordered set of vertices connected by edges. A path from s to t through (say) two intermediate nodes u and v is represented as $p = \{s \rightarrow u \rightarrow v \rightarrow t\}$. This path thus comprises of three edges (s, u), (u, v) and (v, t). The assumption is made that each vertex in V is on some path p connecting s to t. The notation $\rightsquigarrow$ is used to indicate a path from one vertex to another via some unspecified subset of vertices in V, for example $p = \{s \rightsquigarrow t\}$.

Using the definition of a flow network as described above, a flow f inside a network can be formally defined. A flow f in a flow network G, is considered to be a value which can be measured between vertices u and v. A flow $f(u, v)$ equals a non negative value such that the following constraints are met:

Capacity Constraint: Flow from one vertex to another must be non-negative, and must not exceed a given capacity. More formally, for all edges $(u, v) \in E$, a flow

f must be ≥ 0 and cannot exceed the capacity $c(u, v)$ of the edge along which it travels.

Flow Conservation: Total flow into each vertex v, must equal the total flow out of a vertex v, excluding vertex s and t. Formally:

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v) \tag{6.5}$$

Figure 6.10 shows an example network (Figure 6.10a) and its corresponding flow network G with a path p (Figure 6.10b). Notice the path $p = s \rightarrow v_1 \rightarrow v_3 \rightarrow t$ is said to have *admitted* one unit of flow along it. The maximum amount of flow that can be admitted along this path is equal to $c(v_1, v_3) = 12$, as this is the lowest capacity constraint for the path p. As noted above, the flow f cannot exceed any given capacity c. Formally, given a path p, the maximum amount of flow admitted along path p is equal to the lowest minimum capacity of the edges contained in p. Using the water pipes analogy, the amount of water flow that can be admitted in a network of pipes is restricted by the smallest pipe (essentially causing a bottle neck).

The value of a flow f in a flow network G can be described (although in a contradictory manner) as the total flow out of the source, minus the total flow into the source. formally defined as:

$$f(G) = \sum_{v \in V} f(s, u) - \sum_{v \in V} f(v, s) \tag{6.6}$$

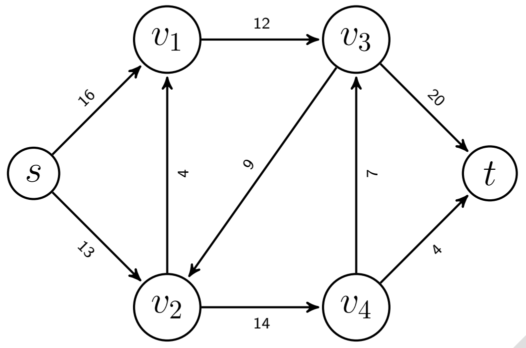
This equation contradicts the fact that no edges point to the source, but it is important in the max flow min cut theorem below. This is explained in the following sub-section 6.3.1.2, in which a (residual) network is used to keep track of the flows and capacities of edges, and where the notion of "back flow" can be used to admit flow in an opposite direction to an existing flow, in order to maximise flow in the entire network.

6.3.1.1 Residual Networks

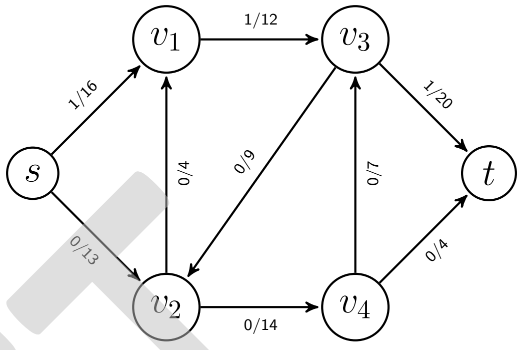
A residual network G_f consists of a network with edge capacities that show how the flow can be changed in a given flow network G. Given a flow network $G = (V, E)$ and a corresponding flow f, the residual network of G induced by f is $G_f = (V, E_f)$. E_f is a set of edges representing the "residual capacity" c_f of G given the flow f. E_f may also contain edges that are not contained in G. This occurs when back flow is admitted in an opposite direction to some existing flow. An edge (u, v) has a positive value in E_f if there is a corresponding available residual capacity $c_f \geq 0$.

Given an edge (u, v) in G, an edge (u, v) exists in G_f such that $(c_f(u, v) = c(u, v) - f(u, v))$. An edge's residual capacity c_f is equal to the capacity c in G minus the flow f in G, which gives the *residual* amount of capacity left (c_f) for a maximum amount of

Figure 6.10: The image in (a) shows an example network of vertices with weighted edges. The image in (b) is a corresponding flow network $G = (V, E)$ of the network in (a). The edges are labelled as $f(u, v)/c(u, v)$ to indicate the amount of flow f and the maximum capacity c of edge (u, v). The flow network (b) depicts edge weights the network given in (a) as capacity constraints $c(u, v)$. A flow f of 1 unit is shown along path $s \to v_1 \to v_3 \to t$ in flow network G.



(a) Network



(b) Flow Network G

flow to be admitted. There also exists an edge (v, u), in relation to (u, v) such that the capacity $c_f(v, u) = f(u, v)$. This edge admits flow in the opposite direction to (u, v) at most cancelling out the flow $f(u, v)$. The result of this in the residual graph G_f is that an edge (u, v) with residual capacity c_f can become "blocked" in one direction (u, v) if flow in that direction equals capacity $f(u, v) = c(u, v)$. This does however allow flow to be admitted in the opposite direction f(v,u), thus "reliving" the net flow of a vertex. These reverse edges are essential with respect to some algorithms' maximisation of flow in a network, as will be shown below.

The flow value of an edge may very well be 0. An example flow network (Figure 6.11a) and corresponding residual network (Figure 6.11b) are shown in Figure 6.11, the edges with capacity $c_f = 0$ are shown for completeness. Consider edge (s, v_1) of G in Figure 6.11a, it has $c(s, v_1) = 16$ and $f = (s, v_1) = 4$. The corresponding edges in G_f are:

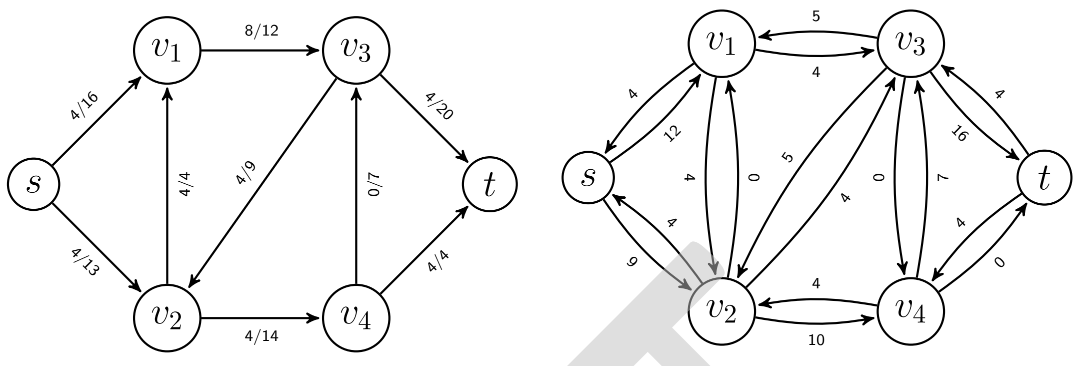
$$\begin{aligned} &(s, v_1) \\ &c_f(s, v_1) &&= c(s, v_1) - f(s, v_1) \\ & &&= 16 - 4 \\ & &&= 12 \end{aligned} \qquad \begin{aligned} &(v_1, s) \\ &c_f(v_1, s) &&= f(s, v_1) \\ & &&= 4 \end{aligned}$$

The process of calculating corresponding values in G_f given G with flow f is repeated for all edges in G to build the residual graph G_f, shown in Figure 6.11b.

6.3.1.2 Augmenting paths

An augmenting path is a path from vertex source s to sink t in a residual network G_f. The (augmenting) path from s to t must intuitively have a residual capacity along its edges on which an amount of flow can be admitted, other wise the path would not

Figure 6.11: The image in (a) shows a flow network $G = (V, E)$ with flow f, the edges are labelled as $f(u, v)/c(u, v)$. The image in (b) is the corresponding residual network $G_f = (V, E_f)$ of G induced by flow f, the edges are labelled with their residual capacity $c_f(u, v)$.



(a) Flow Network G (b) Residual Network G_f

exist in G_f. An example of an augmenting path can be seen in Figure 6.12b. If the residual network G_f is also treated as a flow network, then the permitted flows along an augmenting path p can be increased (as previously shown in Figure 6.10 where path p has a flow of 1 unit). In a similar manner to the way constraints are handled in a standard flow network, the maximum value of flow that can be admitted along an augmenting path is determined by the lowest residual capacity c_f on path p, formally; $c_f(p) = min\{c_f(u, v) : (u, v)\ is\ on\ p\}$. An example of the permitted flow along an augmenting path is shown in Figure 6.12.

6.3.1.3 Network Cuts

This sub-section describes the notion of a network cut, it will later be explained how cutting the network representing the web graph can be used to produce a solution to the WBD problem. A $cut(S, T)$ of a flow network $G = (V, E)$ is a partition of the set of vertices V into two sets S and T, such that $s \in S$ and $t \in T$. The capacity of a $cut(S, T)$ is defined as:

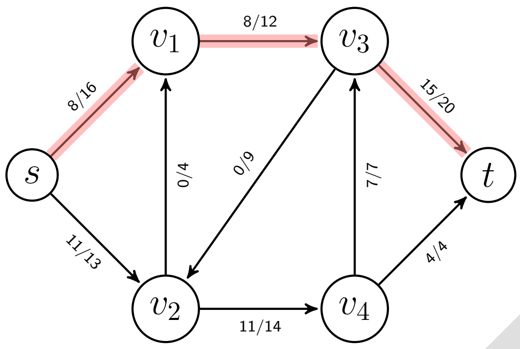
$$c(S, T) = \sum_{u \in S} \sum_{v \in T} c(u, v) \tag{6.7}$$

This formula sums the capacities of edges cut that point in the direction of S to T. The net flow of a cut is the sum of flows from S to T minus the flows from T to S. The net flow is defined as:

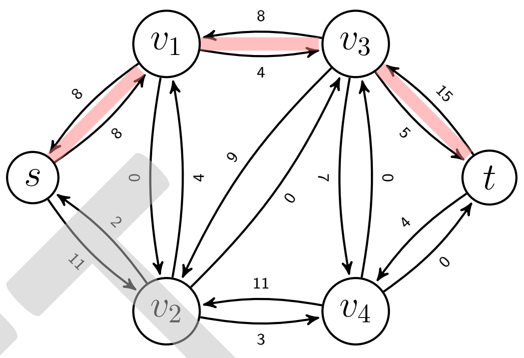
$$f(S, T) = \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{u \in S} \sum_{v \in T} f(v, u) \tag{6.8}$$

Figure 6.13 shows a $cut(S, T)$ of the flow network G such that set $S = \{s, v_1, v_2\}$ and set $T = \{t, v_3, v_4\}$. The net flow of sets S and T is $f(S, T) = 19$ and the capacity

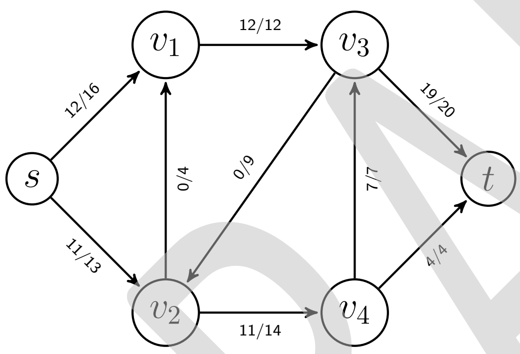
Figure 6.12: Augmenting path example. (a) and (b) show a flow network G and its corresponding residual network G_f respectively, each are highlighted with an augmenting path p ($s \to v_1 \to v_3 \to t$). (c) and (d) show the resulting flow network G' and residual network G'_f respectively after a max flow has been admitted along augmenting path p. The max flow is determined by $c_f(p) = 4$ as a consequence of edge (v_1, v_3)).



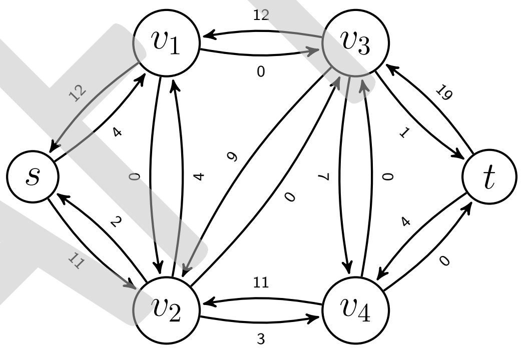
(a) Flow Network G



(b) Residual Network G_f

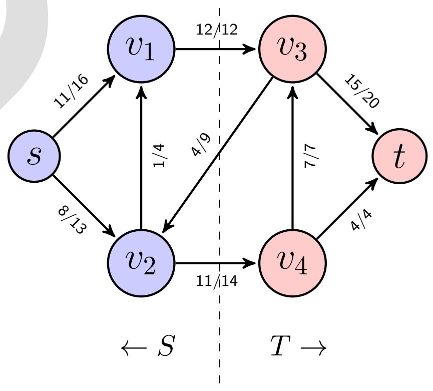


(c) Resulting Flow Network G'



(d) Resulting Residual Network G'_f

Figure 6.13: A network with an example cut(S,T).



$c(S,T) = 26$. A minimum $cut(S,T)$ of a flow network G is a cut such that the capacity of sets S and T resulting from the cut is the minimum over all possible cuts of the network.

6.3.1.4 Max-Flow / Min-Cut theorem

Given a flow network $G = (V, E)$ with vertex source s and sink t with a flow f in G, the max-flow/min-cut theorem states that *the maximum flow f in a flow network G between s and t is equal to the capacity of the minimum $cut(S,T)$ of G.* Given a flow f in a flow network G the max-flow/min-cut is found when the following (equivalent) conditions are met:

1. A flow f is a maximum flow in G
2. The residual network G_f contains no augmenting paths
3. The flow in the network $f(G) = c(S,T)$ for some $cut(S,T)$ of G.

(1) states that a flow f needs to be a maximum flow in G. This can be proven using (2). If f is said to be a max flow, while there is an augmenting path in G_f, then there will be a possible increase in flow such that f can be increased and is not a max flow. (3) states that the total flow out of the source minus the total flow into the source equals the capacity of some $cut(S,T)$.

An example of a max-flow/min-cut is shown in Figure 6.14. The residual network G_f in Figure 6.14b has no augmenting paths from s to t. The flow network G in Figure 6.14a has a maximum flow f. By traversing non zero edges in G_f from s the minimum $cut(S,T)$ can be found, this traversal is highlighted in Figure 6.14b.

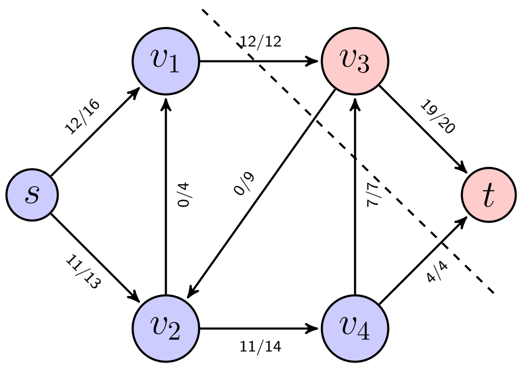
The following section 6.3.2 gives a description of the Ford Fulkerson Algorithm [83] which can be used to find a solution to the max-flow / min-cut problem.

6.3.2 Ford Fulkerson Algorithm

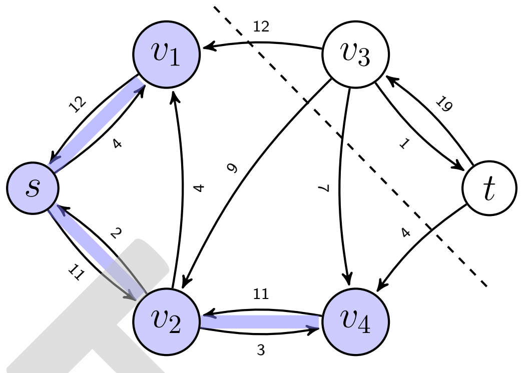
This section gives a description of the Ford Fulkerson Algorithm [83] which is used to compute the max-flow in a given flow network $G = (V, E)$. The algorithm works very simply. For a flow network G, if there exists an augmenting path in the corresponding residual network G_f from s to t, with available residual capacity c_f, flow is admitted along this path. This process is performed iteratively until there are no more augmenting paths in G_f, thus f is a max-flow in G. The resulting residual network G_f can then be used to find the min-cut (as shown in 6.14).

The Ford Fulkerson algorithm is shown in Table 6.2. The parameters are the flow network $G = (V, E)$ which includes the vertices source s and sink t. It is assumed that edges have associated capacities $c(u, v)$ and flows $f(u, v)$. These values are used by the algorithm to find the maximum flow $f(u, v)$. On commencement of the algorithm

Figure 6.14: (a) is a flow network G with max-flow f. (b) is the corresponding residual network G_f with no augmenting paths. The min-cut (S, T) is shown such that $S = \{s, v_1, v_2, v_4\}$ and $T = \{v_3, t\}$. The cut has a capacity of $c(S, T) = 23$, a net flow of $f(S, T) = 23$.



(a) Flow Network



(b) Residual Network

the edges have flow of zero ($f(u, v) = 0$). The algorithm then proceeds to augment flow along available augmenting paths until no more augmenting paths exist in G_f. An illustration of the operation of the Ford Fulkerson algorithm, using an example network, is presented in Appendix B.

Table 6.2: Ford Fulkerson Algorithm [83], based on the description presented in [57].

```
Algorithm Ford Fulkerson ($G$, $s$, $t$)
 1: while { exists path $p$ from $s$ to $t$ in $G_f$ } do
 2:   $c_f(p) = min\{c_f(u, v) : (u, v)\ is\ in\ p\}$
 3:   for { each edge $(u, v)$ in $p$ } do
 4:     if { $(u, v) \in E$ } then
 5:       $f(u, v) = f(u, v) + c_f(p)$
 6:     else
 7:       $f(v, u) = f(v, u) - c_f(p)$
 8:     end if
 9:   end for
10: end while
```

6.3.2.1 Ford Fulkerson Implementation characteristics.

The Ford Fulkerson algorithm (Table 6.2) is a way of finding the maximum flow in a network. It does this by finding augmenting paths in a residual network, notice that the algorithm does not explicitly state how augmenting paths should be found (as indicated by the "while" condition in Table 6.2, line 1). Finding an augmenting path could be done by arbitrarily searching paths, however if a poor method is chosen it could lead to a failure to terminate the algorithm [190].

In this work a breadth first search is used, commencing by finding shortest paths from the source in the residual network. This particular method of finding augmenting paths is also known as the Edmonds-Karp Algorithm [72]. It has a polynomial run time because of its greedy approach to finding an augmenting path [92]. In the remainder of this work, the algorithm used to find the max flow in a network is simply referred to as the Ford Fulkerson algorithm.

The implementation used with respect to the work described in this thesis is founded on the code presented in [166]. This implementation uses a Breadth First (BF) search to discover augmenting paths from source s to sink t. The method crawls the graph, in a BF manner, using only paths with available residual flow. When the sink node is reached an augmenting path has been found, and flow can be augmented along this path. The process is repeated until no more paths can be found.

Some particular characteristics of the implementation should be noted. Firstly, due to the nature of the BF crawl, vertices that are visited once, are not visited again. Thus once an edge has been used to travel from (u, v):

1. Any additional edges (u, v) will be disregarded (where several edges linking vertices exist).

2. If an edge (u', v) exists, this will also be disregarded as v has already been visited (using edge (u, v)).

These characteristics produce a method that can reduce the complexity of a web graph with many edges by discounting certain edges; because, when the method is used to model a web graph as a flow graph, duplicate edges will not be used, and self loops and reverse edges will be excluded. This is done with a bias of edges in a BF ordering from the source vertex to the sink. It will be shown in the evaluation section that this particular characteristic, when used in conjunction with a standard flow graph representation, can produce high quality WBD solutions.

6.3.3 Partitioning web graphs using flow network theory

The k-cut problem requires finding the set of edges which, when removed, partition a graph into k connected components (clusters). The problem of partitioning a graph into $k = 2$ partitions has been extensively studied [148]. In this work a 2-cut graph partitioning approach using the *min-cut max-flow* theorem is applied to the WBD problem. In this section two problems related to the 2-cut problem are considered: (1) the 2-cut $s - t$ problem given a graph G with source s and sink t vertices and (2) the general 2-cut problem given G only.

The 2-cut $s - t$ problem seeks to address the question *how can a graph be segmented by cutting the minimum number of edges so that $s \in S$, and $t \in T$?* This question can be answered by modelling the graph as a flow network, and subsequently using a single instance of the Ford Fulkerson algorithm given G, s and t to find the maximum flow, and subsequently the minimum cut (as explained earlier in section 6.3.2).

The general 2-cut problem seeks to address the question *how can a graph be divided into two sets by cutting the minimum amount of edges?* Notice that in the general 2-cut question there is no mention of which vertices of the graph should be designated as source s and sink t. A simple (yet expensive) approach to the general 2-cut question is to iterate the choice of s and t over all possible combinations of vertices. In practice this can be done by using an instance of the Ford Fulkerson algorithm for each combination of s and t. The combination that has the minimal cut of all the instances is then chosen as the mincut of the given graph, subsequently producing two partitions.

This simple approach highlights that choosing the nodes s and t is a non-trivial problem to solve. It should be noted that the work described in this thesis is not concerned with improving the run time of mincut algorithms as in the case of the work of (say) Karger [103], who used a randomised method in contrast to considering all combinations, to produce fast solutions to the general mincut problem. Instead, with respect to the work described in this thesis, the interest is focused on the quality of the WBD solutions produced.

The approach of applying a 2-cut method to the WBD problem consists of a slight variation, to the methods considered above, related to a choice of s and t. The variation is due to the fact that there is a specific seed page (w_s) in the given web graph. This extra information can be used when choosing the source and sink vertices. When the graph is modelled as a flow graph, the vertex that represents the seed page can be designated as the source s (or indeed the sink t).

The work presented in this section broadly addresses the 2-cut problem using two techniques, the first is to fix the source s at the given seed w_s while iterating t and the second is to iterate all combinations of s and t. The evaluations of these two techniques are presented in section 6.4.1 and 6.4.2 respectively. It will be shown that representing the seed page node as the source s does not yield the best results in terms of a WBD solution. In fact choosing nodes s and t from within the same cluster yields the "best" cut of the graph in terms of WBD solution.

6.3.3.1 Flow Graph Models

In this work, the web graph is modelled in two ways:

Standard. The standard model simply represents the flow graph in exactly the same manner as the underlying structure of the given web graph. Note, this is not strictly a valid flow graph model, given the criteria in section 6.3. This is because web graphs typically include: self loops, reverse edges and duplications. When used in conjunction with the implementation shown in section 6.3.2.1 these violations are not an issue. It will be shown that this model gives the most appropriate WBD solutions.

BackLinks. The backlinks model is founded on the standard model described above with the addition of *backlinks*. These edges are added to increase the connectivity of the graph, where otherwise flow could be restricted or minimised. A backlink is added for every vertex v, such that for every edge pointing to v (u, v) a backlink is added (v, u).

6.3.3.2 WBD solution

To produce a WBD solution from a graph partitioned into two clusters using the min cut of the graph, a similar method is used as in previous techniques. The cluster containing the seed node w_s is chosen as the target cluster K_T, and the remaining cluster is identified as the noise cluster K_N. The task is to produce a target cluster that contains a maximum number of target pages from the website, and a minimum number of noise pages.

6.4 Evaluation of Minimum Cut Approach

The evaluation of the minimum cut approach to generating WBD solutions presented in this section is divided into two main sub-sections. The first (section 6.4.1) considers the situation when the source s is fixed as the seed page and the remaining vertices are considered as potential sinks. The second (section 6.4.2) considers the situation where neither s or t are fixed and the complete set of vertices is considered as potential sources and sinks. The reported evaluation was conducted using both flow graph models introduced in section 6.3.3.1, the standard and backlinks models.

6.4.1 Fixed source s, iterating sink t

The evaluation in this sub-section reports on an approach to finding the mincut of a graph when sink t is unknown, it uses the technique of fixing s and iterating t. Fortunately, in the definition of the WBD problem, a known vertex is supplied in the form of a seed web page (w_s). In this particular case the single seed is fixed as the source node, the sink is then iterated via all possible remaining nodes.

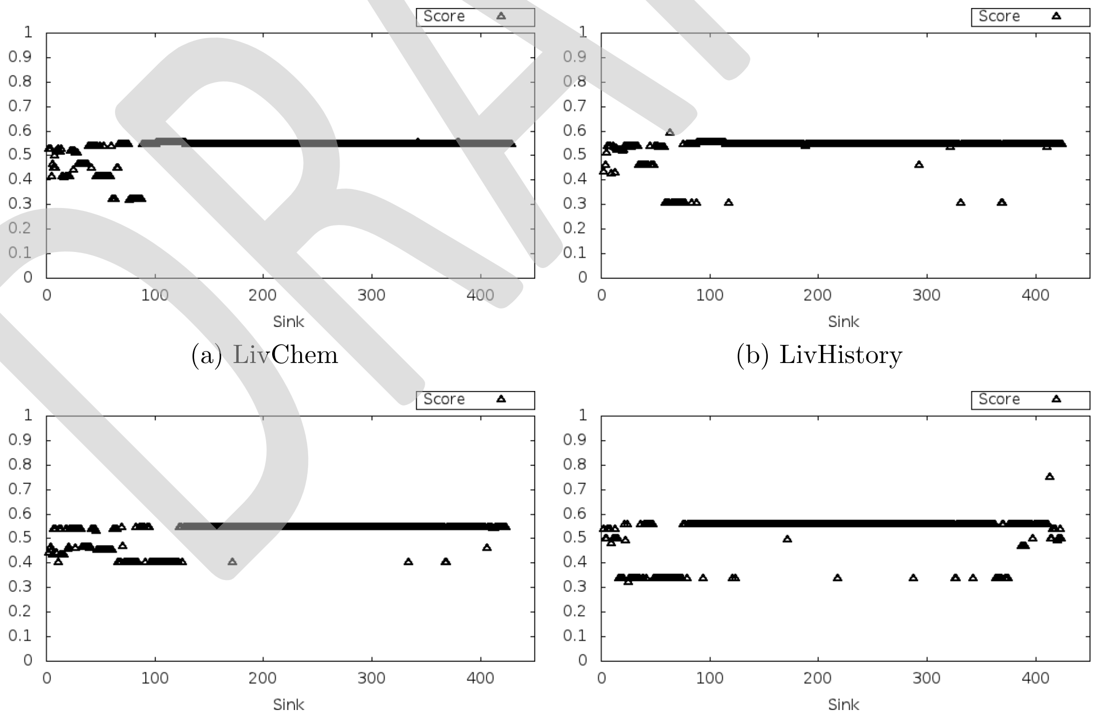
For a graph of n vertices labelled $\{0 \dots n\}$ starting from the seed page w_s, the source is fixed at s equals vertex 0. The sink t iterates from vertex 1 to n. The results were first analysed in order of sink t selection (1 to n).

The min cut partition produced is such that $s \in S$, and $t \in T$. In this evaluation scenario, because the source is the seed page ($s = w_s$), it can be inferred that the partition produced will be such that the vertices in set S are part of the website, thus forming the website boundary. Therefore the nature of the generated WBD solutions was measured with respect to the set S.

6.4.1.1 Standard Flow Graph Model

The WBD performance scores (shown in Figures 6.15a, 6.15b, 6.15c and 6.15d) illustrate the poor and consistent performance (scores of between 0.3 and 0.6) for all iterations of t in the standard flow graph representation of the web graph. In some iterations the cardinality of the sets S and T can reflect an ideal WBD solution, relative to the number of target vertices. However, the composition of these cuts do not reflect the correct target and noise ratios (Figure 6.16). The identified website boundary cluster (K_T) either contained: a high ratio of target to noise pages, but with the target pages in K_N; or a low ratio of target to noise pages are contained (low precision), but with a high proportion of target pages in K_T (high recall). Over the data sets tested, no emerging pattern could be identified as to what source vertices to choose in terms of cutting the graph to give a high performing WBD solution.

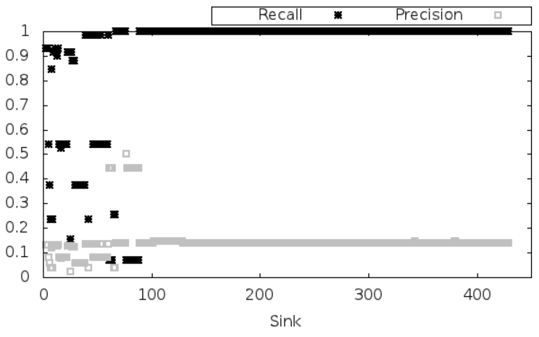
Figure 6.15: WBD performance score using the standard flow graph model, with a fixed source $s = w_s$ (the seed page) and iterating sink t. For a graph of n vertices, $s = 0$ is fixed, t is then iterated from $t = 1$ to $t = n$. The plots show the performance score value (y-axis), plotted for each sink t (x- axis).



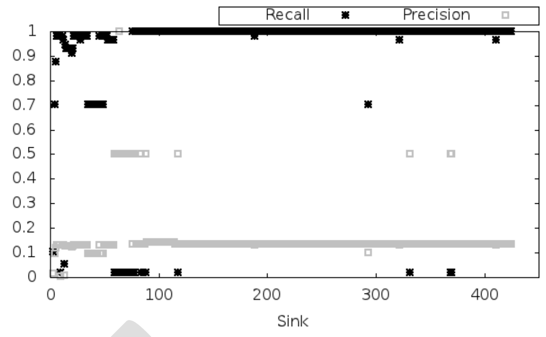
(a) LivChem (b) LivHistory

(c) LivMath (d) LivSace

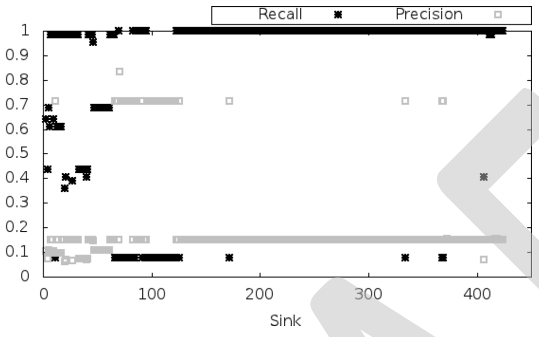
Figure 6.16: WBD performance recall and precision values using the standard flow graph model, with a fixed source $s = w_s$ (the seed page) and iterating sink t.



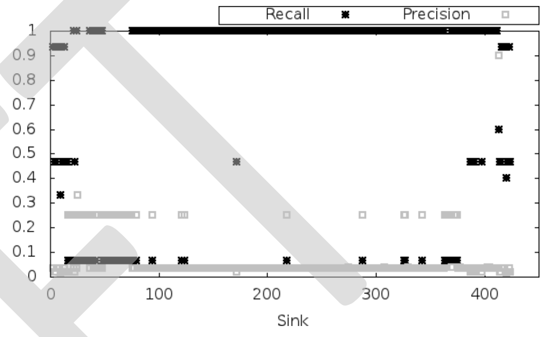
(a) LivChem



(b) LivHistory



(c) LivMath



(d) LivSace

6.4.1.2 BackLinks Flow Graph Model

The performance using the backlinks flow graph model show a slightly different story to the poor performing standard model results presented above. The performance results generated from the backlinks tests show a clear two tiered performance (Figure 6.17). The performance is either similar to the standard test with scores of between 0.4 and 0.5, or a performance score of around 0.7. This "two level" trend is consistent across all data sets tested (as shown in Figures 6.17a, 6.17b, 6.17c and 6.17d).

Closer observation of the compositions of clusters S and T reveals that there were only two types of cut being made in the flow graph. The cut was either:

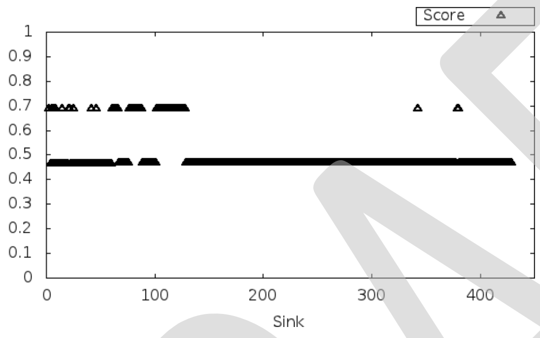
1. vertex $s \in S$ and all remaining vertices in T.

2. vertex $t \in T$ and all remaining vertices in S.

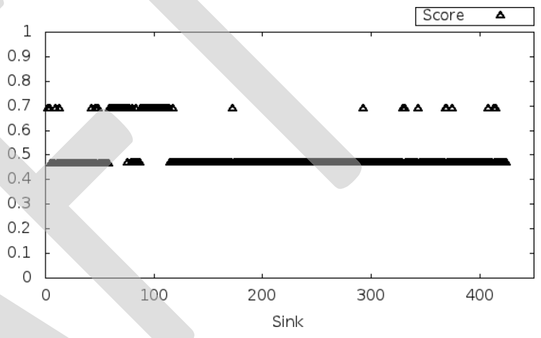
Thus the minimum cut is such that a single node (vertex) is being cut from the graph. Given the fixed nature of s, and the iteration of t the cut is either: set S containing only s, and set T containing the remaining nodes; or set T containing only t, and set S containing the remaining nodes. Interpretation of these two types of cut in the context of a WBD solution produces either: (i) a collection containing a single isolated target vertex, the seed node (1 above); or (ii) a collection containing the seed node and a large number of noise pages (2 above).

The dense nature of the backlinks representation of the web graph intuitively makes the lowest capacity cut of the network in order to separate s from t, this is simply a cut isolating s or t from the network. Which essentially makes the capacity of the mincut in the network equal to the degree of the isolated vertices s or t. Cutting s or t from the main network is much cheaper than performing a cut of potentially very high capacity across the dense connections of the network in order to correctly separate all target nodes from remaining noise nodes.

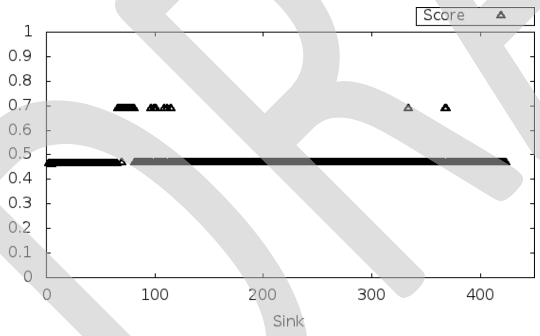
Figure 6.17: WBD score using the backlinks model, with a fixed source $s = w_s$ (the seed page) and iterating sink t. For a graph of n vertices, $s = 0$ is fixed, t is then iterated from $t = 1$ to $t = n - 1$. The plots show the score value (y-axis) plotted for each possible sink t (x- axis).



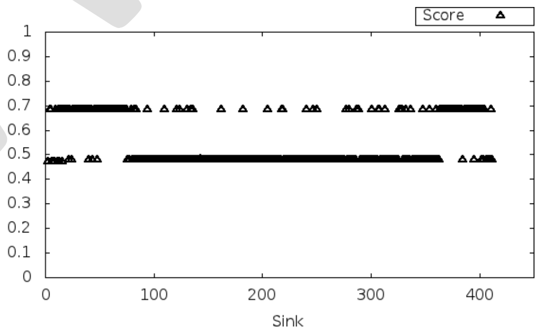
(a) LivChem



(b) LivHistory



(c) LivMath



(d) LivSace

The potential WBD solutions were subsequently analysed in order of: (1) lowest capacity mincut of all iterations of t and (2) highest density of set S. The configuration that featured the lowest capacity mincut was then selected as the most desirable WBD solution in each case. Inspection of the ordered results did not provide any greater insight into how the sink should be selected so as to produce the best WBD solution.

The lowest capacity mincut did not yield the highest score value for the WBD solution (in some cases quite the opposite infact). A low capacity min cut implies that a low number of edges were cut to separate s from t. As in the case of the reported experiments using both the backlinks and standard flow models, the lowest capacity cuts often resulted in low performing WBD solutions, this is because a low cut can segment a small portion of the graph irrelevant to WBD purpose.

Consideration of the density value of set S and T gives slightly better insight into the cluster composition. If a cluster contains a single vertex, then it will have a minimum density value associated with it. Analysing each potential WBD solution in terms of which solutions produced the highest density cuts of the graph in terms of set S, effectively filters out cuts of the graph that contain single vertices. Although analysing potential WBD solutions in terms of the highest density of set S removed low performing solutions, it was found not to be an effective method to yield high performing WBD solutions.

By fixing the source s (as the given seed node w_s), and iterating sink t, for a graph of n vertices, the minimum cut approach using the backlinks flow graph model using $n-1$ iterations, performs a mincut of the graph on each iteration. As shown above, this did not yield a good performance in terms of the WBD solutions produced. The cuts of the graph did not reflect good performing partitions using both backlinks and standard representations of the web graph.

6.4.2 Iterating source s and sink t

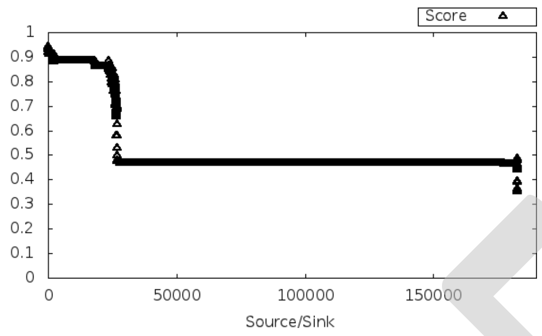
An alternative to fixing the source s and iterating t is to iterate both the source s and the sink t for all combinations of vertices in the input graph. For a graph containing n vertices, this produces $n * n$ combinations of s and t. This is therefore an exhaustive approach to partitioning the graph when compared to the previous method which considered only $n-1$ combinations. The same method for finding the minimum cut of the graph for each s and t combination is again used. For each iteration a mincut is performed, the lowest cut across all combinations is then the mincut of the graph.

6.4.2.1 Standard Flow Graph Model

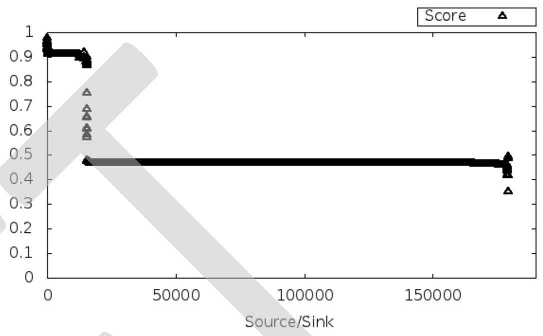
For each of the datasets there exists at least one WBD solution that has a score of > 0.9. (shown in Figures 6.18a, 6.18b, 6.18c and 6.18d). Sorting the results by highest performing WBD solution shows an interesting trend. For two of the datasets (LivChem Figure 6.18a and LivHistory Figure 6.18b) the performance tails off quite dramatically after around 25,000 combinations. This trend can also be observed in the remaining two datasets but when a much lower number of combinations is reached (LivMath Figure6.18c and LivSace Figure 6.18d). The performance trend for all datasets eventually tails off to the same low scoring amount as reported in the fixed source testing results presented above (a score value of approximately 0.5).

The frequency plots of the top performing results reveal that the sink vertex should be selected from target website vertices, while the source should be selected from either the target or the noise vertices (Figures 6.19a, 6.19b, 6.19c, 6.19d). In fact the results shown in the figures indicate that it is usually better to identify the sink node with the seed page rather than the source, which is contrary to the intuitive thinking adopted for the first set of recorded tests where the source was identified as the given seed page of a website.

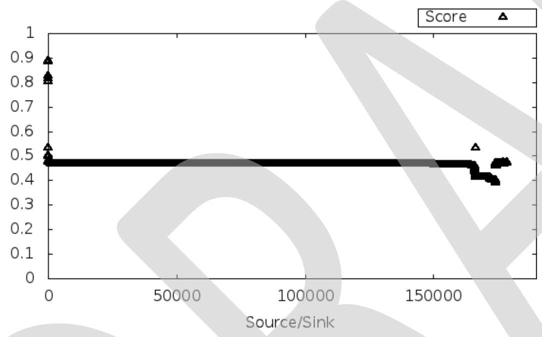
Figure 6.18: WBD performance score using the standard flow model, with iterating source s and sink t. The plots show the score value (y-axis), plotted for each sink s and t combination (x- axis). All cominbations of s and t are shown.



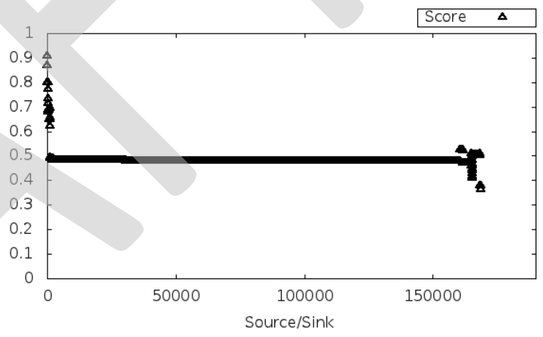
(a) LivChem



(b) LivHistory



(c) LivMath



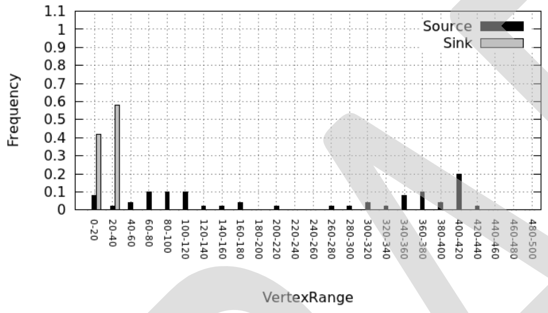
(d) LivSace

6.4.2.2 Backlinks Flow graph model

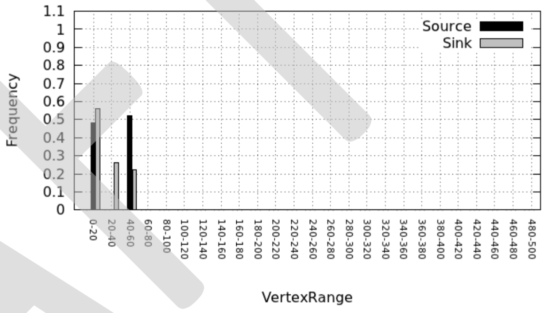
The backlinks model does not show any improvement when iterating over all combinations of s and t in the graph. The results exhibit the same two level score pattern as for the previous experiments (Figures 6.20a, 6.20b, 6.20c and 6.20d), which is consistent with the results from the the fixed source iterating sink tests shown previously (Figure 6.17).

The iterating source and sink WBD solutions presented here show a vast improvement in terms of the quality of the website boundary discovered compared to the previous fixed source iterating sink solutions. When using the standard model the results proved that there does exist a s and t combination where the mincut (S, T) of the graph produces a more representative WBD solution.

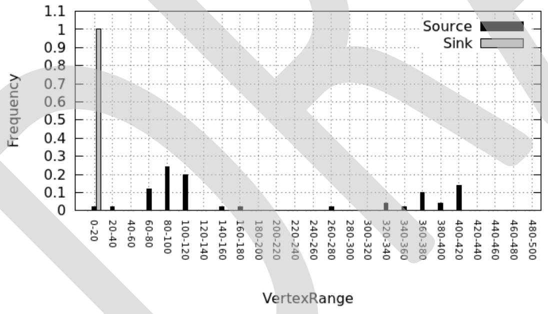
Figure 6.19: Frequency distribution of source and sink for the top 25 scoring WBD solutions using the standard flow model, with iterating source s and sink t. The plots show the frequency value (y-axis) plotted against the vertex range. Note the x-axis depicts vertices in breadth first order from seed node, the lower range represent target nodes, the higher range represent noise nodes, in accordence with the amount of target and noise in a data set (see section 4.3).



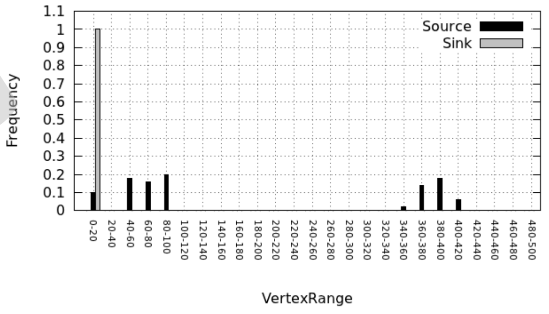
(a) LivChem



(b) LivHistory

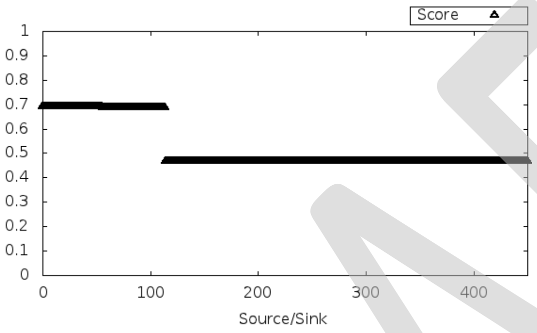


(c) LivMath

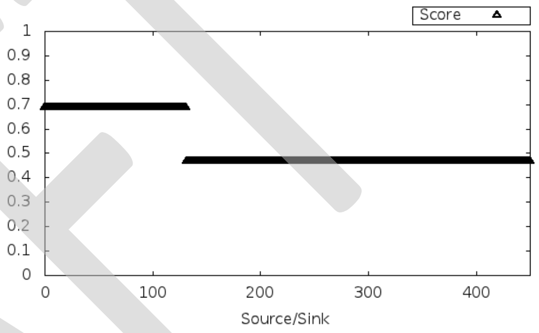


(d) LivSace

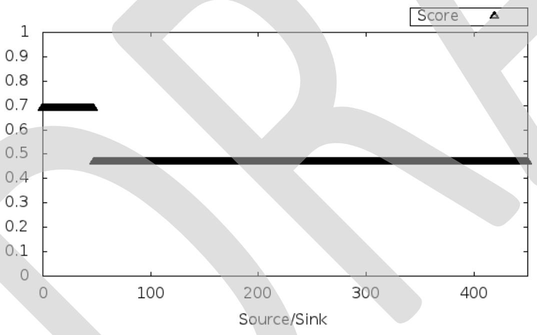
Figure 6.20: WBD score using the backlinks model, with iterating source s and sink t. The plots show the score value (y-axis), plotted for each sink s and t combination (x-axis). The top 450 results are shown only, this is to highlight the two level trend.



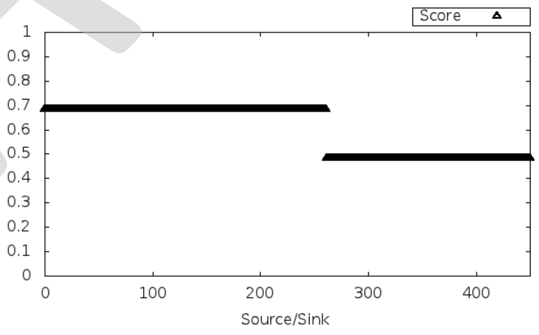
(a) LivChem



(b) LivHistory



(c) LivMath



(d) LivSace

6.4.3 Evaluation Summary

The evaluation of results using the minimum cut approach to graph partitioning are summarised in this sub section. This summary compares the four possible combinations for allocating the source s and sink t using either target vertices or the noise vertices. Given a graph G which contains two clusters: C_T containing target pages from the website and C_N containing noise pages (notation recalled from section 3.4.1). The selection of a source s and sink t from vertices in G falls into one of four scenarios:

1. $s \in C_T$ and $t \in C_T$ - Good performance.
2. $s \in C_T$ and $t \in C_N$ - Bad performance.
3. $s \in C_N$ and $t \in C_N$ - Bad performance.
4. $s \in C_N$ and $t \in C_T$ - Good performance.

The scenarios listed above illustrate the main finding which is that if the sink vertex is not chosen from the target cluster, the performance of the WBD solution produced is poor.

Choosing the sink from the target cluster means that flow is augmented to an end point that is in the more highly connected (target) reign of the graph. If flow is augmented from inside ($s \in C_T$) the target cluster a bottle neck is created (and subsequently cut) on the boundary of the target (website) cluster. This is because flow is augmenting inside the more highly connected cluster, with low amount of flow augmenting to the noise cluster due to fewer augmenting paths. If flow is augmented from outside ($s \in C_N$) the target cluster, a bottle neck is again created on the boundary of the target (website) cluster due to flow being augmented along the few available paths into the target cluster. These paths cause a bottle neck of flow on the edge of the more highly connected cluster.

The backlinks model provided a much more stable graph partitioning compared to the standard model. This was exhibited as a clear two tier trend in WBD performance. The stability refers to the range of source and sink selections that can be made which produces a graph partition that has the same (stable) WBD performance. Consequently the WBD solutions produced were found to be less “volatile” than those produced using the standard model. This implies that there is some slight leniency with respect to the selection of source and sink vertices and the quality of WBD solutions generated. However, the major drawback of the backlinks model was that it did not perform as well as the standard model in terms of WBD solution performance. This was due to the model creating uniformity in the graph structure by adding extra (back) links which would otherwise be used to advantage with respect to WBD performance using a flow analysis method.

6.5 Conclusion

This chapter presented two graph partitioning approaches to generate solutions to the WBD problem. The first approach presented was a hierarchical graph partitioning method based on Newman's approach. This method partitions an input graph into clusters based on a modularity value that produces clusters of high interconnectivity. The second approach was based on the min-cut max-flow theorem and finds the minimum cut within a web graph when the graph is represented as a flow network. This method segments the graph into two dense clusters while minimising the edges cut between the clusters.

For each approach two sets of experiments were conducted. In the first experiments both methods were applied in an intuitive context. The Newmans method was simply applied to complete data set snapshots, while the minimum cut method used the given seed web page w_s as the source of flow, and iterated the sink from the remaining vertices. The reported results generated from these experiments indicated that they were unable to produce good quality WBD solutions.

In the second set of experiments variations of both approaches were considered. The Newman method was applied to varying sized snapshots of the data sets of the form that may be produced using different prior web crawl strategies. The minimum cut method was iterated over all combinations of source s and sink t. The results produced from the further experiments with respect to both the mincut and Newman approaches indicated an improvement in the WBD solutions produced in comparison with those produced by the first sets of experiments. These results thus indicated that both approaches have potential in terms of producing high quality WBD solutions. The results obtained also demonstrated that the connectivity between vertices of a single website are in fact encoded in the underlying web graph structure. It was also shown that this structural characteristic has the potential to be exploited using the methods presented in this chapter. Finally it should be noted that the WBD approaches presented in this chapter focused on exploiting only the hyper link structure when producing WBD solutions. The previous chapter, Chapter 5, exploited only content structural elements with respect to the WBD problem. The following chapter, Chapter 7, concentrates on both the content and hyper link structure of web pages when producing WBD solutions.

Chapter 7

Dynamic Techniques

This chapter presents the investigation of the WBD problem in the dynamic context. In the dynamic context the web data is not fully available prior to the start of analysis (as previously explained in section 3.4.2). The approaches presented in this chapter used various graph traversal techniques to gather portions of data, which were then clustered incrementally in order to produce a WBD solution using only partial data.

The approaches in chapters 5 and 6 described solutions to the WBD problem in a static context. The static approach operated using a three phase process: (1) collect data, (2) pre-process data and then (3) produce WBD solution. In the static context these phases are performed in sequence, there is no repeating of previous phases once a phase has been completed. In the dynamic approaches described in this chapter the same three phases are used, however they are applied in such a way that the phases are repeated. The repetition allows a portion of web data to be gathered, pre-processed and then a WBD solution produced from this portion. In the dynamic approach the web page data is gathered by traversing the web graph using the hyperlink structure. The web pages are then pre-processed and feature representations created for each page. The pages are then incrementally clustered as the pages are traversed, a website boundary is then identified based on the clusters produced. A main focus of this chapter is an investigation of the "power" of the random walk based method of graph traversal with respect to WBD performance in particular settings where the amount of data is large and not immediately available.

The evaluation of the dynamic approaches presented in this chapter was performed using the three categories of data sets which were previously introduced in chapter 4: (1) Binomial Random Graphs (BRG), (2) Artificial Data Graphs (ADG), and (3) Real Data Graphs (RDG). Recall that the BRGs are a very simplistic model of the web graph with respect to the WBD problem, which was used to test preliminary dynamic approaches to the WBD problem. The ADGs modelled the web using a more sophisticated method based on the preferential attachment model. The evaluation using ADGs allowed for the dynamic approaches to produce WBD solutions in a controlled environment. The final data category used for evaluation of the dynamic approaches described in this chapter consisted of RDGs which were used previously as reported in Chapters 5 and 6.

The remainder of this chapter is organised as follows. In section 7.1 a formal description of the WBD problem in the dynamic context is given. Details of the dynamic approach with respect to the WBD problem is presented in section 7.2. Two main methodologies are used in the dynamic approach, a graph traversal method and a incremental clustering method; which are presented in sections 7.3 and 7.4 respectfully. Section 7.5 details methods of graph representation which can be used to weight or add edges to a graph. Section 7.6 presents some issues and characteristics of the dynamic approach. Section 7.7 presents an evaluation of the dynamic approaches with respect to the WBD performance, followed by an evaluation summary in section 7.8. Finally this chapter presents a conclusion to the work undertaken in this chapter in section 7.9.

7.1 Formal Description

In this section a formal description of the WBD problem in the dynamic context is presented. Recall the general WBD problem's formal description presented in section 3.4. Given a collection of web pages W, comprising n individual pages w, such that $W = \{w_1, w_2, \cdots, w_n\}$, where the seed page is w_s, the website boundary (ω) is said to be the bounded subset of pages in W that form the website given by w_s. Each of the individual web pages can be described using a dimensional numerical vector of length m, $V = \{v_1, v_2, \ldots, v_m\}$. This set of pages can be modelled using a graph $G = (W, E)$, where W is a collection of web pages (as noted above), and the set E keeps track of all directed (hyper) links between pairs of elements of W. The key characteristic of the dynamic context is that the graph $G = (W, E)$, and intuitively the set W and E, is not known apriori. Thus at commencement of the dynamic approach to identify a solution to the WBD problem all that is known is the seed page w_s, and its associated directed edges. The following section 7.2 presents the dynamic approach to the WBD problem, with respect to the formal description presented in this section.

7.2 The Dynamic Approach

This section presents the dynamic approach used in this chapter to provide solutions to the WBD problem in a dynamic context. Each of the dynamic approaches presented in this chapter is founded on an incremental methodology which was used to segment a graph in order to provide a WBD solution. The dynamic approaches are based on the incremental methodology as defined in the template presented in Table 7.1. This template exhibits two important aspects that underpin the dynamic approaches presented in this work:

1. Graph Traversal.
2. Incremental Clustering.

The graph traversal methods detail how the nodes of the graph are visited starting at the initial point (node w_s). Thus, referring to the algorithm presented in Table 7.1 selecting Q from G (line 4). In this work both deterministic and probabilistic methods are considered for the graph traversal of G (see section 7.3 below). The graph representation of G used in conjunction with the probabilistic methods of graph traversal, employed methods to effect the traversal of the graph (see section 7.5). The incremental clustering methods (see section 7.4) determines what cluster the nodes belong to, based on their similarity as they are visited. This is shown as adding Q to K_T or to K_N (line 5).

Due to the differing operation of graph traversal methods when coupled with an incremental clustering method, the clusters produced can vary greatly. Hence the notion of a 'step' is incorporated into the experimental analysis of the techniques. Referring to the template presented in Table 7.1 a step is considered as a single iteration of the repeated sub routine (lines 3-7).

In the forthcoming sections details are provided about the proposed graph traversal methods (section 7.3) and graph representation methods (section 7.5). The incremental clustering processes that will be considered in the rest of this chapter are detailed in section 7.4.

```
Algorithm clustering_template ($w_s$)
1: $K_T = \{w_s\}$; $K_N = \{\}$;
2: set up the process internal state;
3: repeat
4:     select a web page $Q$ from web graph $G$;
5:     add $Q$ to $K_T$ or $K_N$;
6:     update the process state;
7: until convergence;
8: return $K_T$;
```

Table 7.1: Template detailing the dynamic approach to WBD.

7.3 Graph Traversal

This section reports on the methods of graph traversal with respect to the dynamic approaches presented in this chapter. Given a graph G, the sequence of nodes visited by starting at a node and then repeatedly moving from one node to the next by selecting a neighbour of the current node at random is termed a *Random Walk* (RW) on the graph G (see section 2.5.2). An example is shown in Figure 7.1.

The graph G is assumed to model a portion of the web. For the purpose of the evaluation presented it is assumed that graph G resides on some secondary memory storage device and that its nodes/edges are retrieved as needed. If the dynamic graph traversals were to be used on a real web graph, the graph would be distributed at different sites across the internet and its content would have to be downloaded through http requests. An internal representation of the graph G is incrementally built as nodes/edges are retrieved. This means that the structure of the internal graph representation is subject to the graph traversal method used.

A Random Walk (RW) on a directed graph structure can be hindered by the implicit one directional links in a (web) graph, therefore it is necessary to add additional links to the internal graph representation to aid navigation, as shown in Figure 7.2. If a directed link to a leaf node is traversed, a RW will stop at that node. This is illustrated in Figure 7.2a, if edges (D, C) or (B, C) are followed, the walk remains at node C. Figure 7.2b illustrates the graph representation after an s step traversal, this graph essentially is bi-directional.

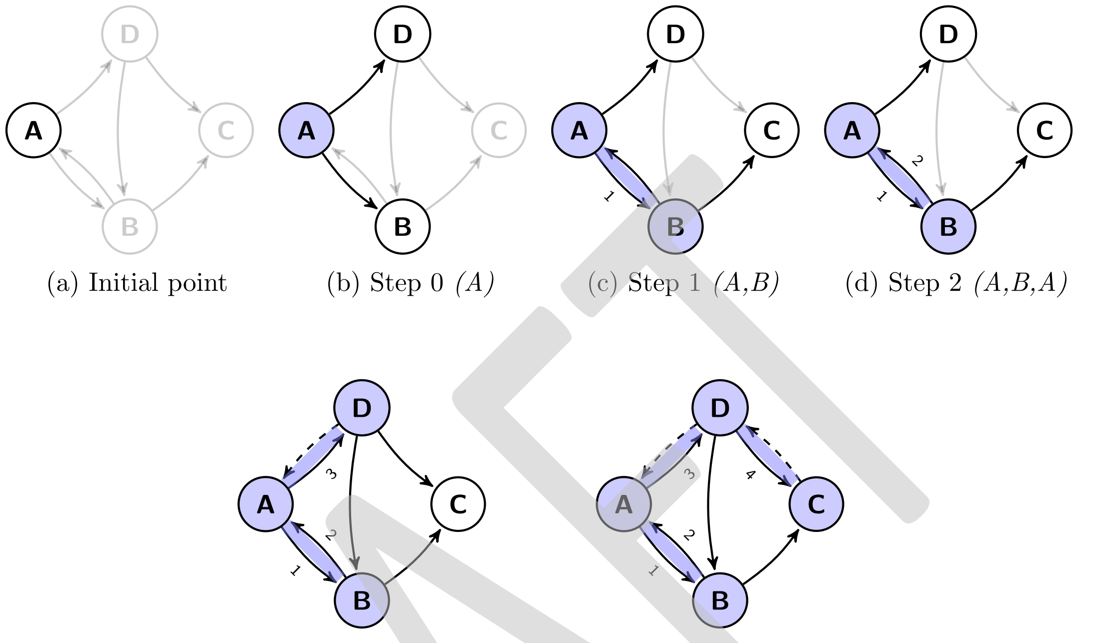
As previously mentioned, this chapter reports on dynamic techniques for the purpose of WBD. A main characteristic of the dynamic context is that the web graph is not available at the start. This is illustrated in Figure 7.1, which shows each step of a random walk and how the graph is retrieved and represented internally. Notice that nodes of the graph can be "known about", but not yet traversed, as directed edges from visited pages indicate potential nodes to be visited next.

In the remainder of this section two categories of graph traversal are considered: (1) Deterministic and (2) Probabilistic. The deterministic traversals proposed were BF and DF (see sub-section 7.3.1). The traversals use a heuristic process to visit nodes of the graph, which is influenced by the graph structure and previous states. The ordering produced by the deterministic methods remained fixed for every traversal of the same graph. The probabilistic traversals proposed were based on a random walk, which means there is an element of choice and unpredictability involved. The first traversal is a simple RW presented in sub-section 7.3.2, it is appealing because of its simplicity. The second is a Self Avoid Random walk (SAR) presented in sub-section 7.3.3; sub-section 7.3.4 presents the third traversal, a Metropolis Hastings Random Walk (MHRW). The final sub-section 7.3.5 presents the Random Ordering (RO) method of selecting pages from a graph. Each graph traversal method is given in detail in the following sub-sections, with respect to the dynamic approach template that was presented in Table 7.1 in the previous section (section 7.2).

7.3.1 Breadth First (BF) and Depth First (DF)

This sub-section presents the Breadth First (BF) and Depth First (DF) graph traversal methods. Both BF and DF are based on a deterministic traversal of the graph G from

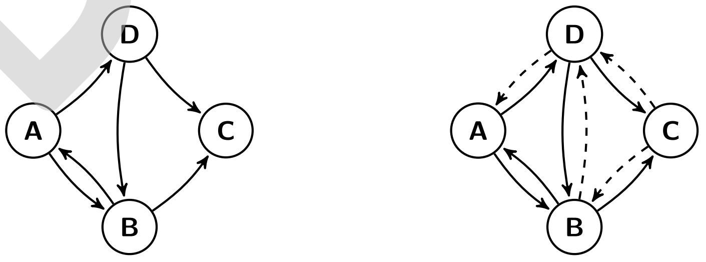
Figure 7.1: Illustration of a random walk discovering of nodes of the graph as it explores the structure. The ordering of nodes visited are *A,B,A,D,C*. The greyed out nodes and edges are undiscovered or unknown, the visible nodes and edges are known via traversal of the structure, while the darker nodes and edges are marked as visited.



(a) Initial point (b) Step 0 *(A)* (c) Step 1 *(A,B)* (d) Step 2 *(A,B,A)*

(e) Step 3 *(A,B,A,D)* (f) Step 4 *(A,B,A,D,C)*

Figure 7.2: A directed graph (a) and corresponding directed graph (b) after an s step traversal whereby additional edges are added to aid a random walk traversal (shown as dashed lines).



(a) Directed Graph (b) Graph after s steps the root node (w_s). The pseudo code for the BF method is shown in table 7.2. Note that the DF process is the same as the BF, except that the queue data structure is replaced with a stack.

The decision on whether to add Q to K_T the target cluster, or to K_N, the noise cluster, is based on the nature of the clustering algorithm used. In the case of IKM, the decision is made based on the Euclidean distance between Q's feature vector and the centroids of the two clusters $K_T \backslash \{Q\}$ and $K_N \backslash \{Q\}$, in the case of ICA the CASA value is used to add Q to the most similar cluster. The sequence of pages to be clustered, is initially given by the traversal of the graph in breadth first order. Once the complete graph has been traversed, this sequence is repeated until convergence is reached.

```
Algorithm BF ($w_s$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  set $Q$ to $w_s$; set a counter to one;
  set up the process internal state;
  set Queue empty; set Ordering empty; e=0;
  repeat
    if Queue not empty then
      Q = Queue.pop; Ordering,push(Q);
      for all neighbours q of Q do
        if not marked then
        mark q; Queue.push(q);
      end for
    else
      Q = Ordering.element(e);
      if e < Ordering.size then e++; else e=0;
    end if
    add $Q$ to $K_T$ or $K_N$;
    increase the counter;
    update the process state;
  until convergence;
  return $K_T$;
```

Table 7.2: Pseudo code for Breadth First (BF)

7.3.2 Random Walk (RW)

The Random Walk (RW) method of graph traversal is described by the pseudo code presented in Table 7.3. In general, given the current page Q, the page to be visited next is selected at random among those pointed to by the links from Q and the set of pages seen so far that point to Q. Note that the walk can revisit nodes multiple times, even before having completed a full sweep of G. It is therefore convenient to re-compute the process state after each step of the walk, rather than at the end of a sweep. The process (internal) state for example, in the case of kmeans, is the calculation of the centroids for each cluster. It could be any process that the algorithm uses to keep track of clusters, for example nearest neighbours etc. It is well-known that any RW on a finite connected graph eventually visits all vertices in it (section 2.5.2). Thus, in principle, the process could run until convergence as in the standard kmeans algorithm. It will turn out, however, that stopping the process after a given maximum number of steps (MaxIterations) is more effective and still results in good quality clusters.

```
Algorithm RW ($w_s$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  set $Q$ to $w_s$; set a counter to one;
  set up the process internal state;
  repeat
    redefine $Q$ to be a random neighbour of $Q$ in $G$;
    add $Q$ to $K_T$ or $K_N$;
    increase the counter;
    update the process state;
  until counter goes past MaxIterations;
  return $K_T$;
```

Table 7.3: Pseudo code for Random Walk (RW)

7.3.3 Self Avoiding Random (SAR) Walk

The Self Avoiding Random (SAR) walk with reset crawls the graph in a random order without returning to previously visited nodes. The pseudo code is shown in table 7.4. When the walk is reset, the previous nodes are cleared, and the process is re-started (R is the list of nodes seen in a particular "run" of SAR.). The reset probability is based on how many noise nodes (pages) are visited, as this number increases the chance of resetting is also increased. The reset parameter r can be adjusted to control the sensitivity of the walk. The number of noise nodes is calculated from the result of adding Q to either K_T or K_N.

7.3.4 Metropolis Hastings Random Walk (MHRW)

Intuitively the SAR and RW methods are based on a random traversal of the graph which can be effected dramatically by the underlying graph structure. Given a graph G, if a node has a high degree, then it is more likely to be chosen randomly, as there is an increased number of connections to this node in G. The Metropolis Hastings Random Walk (MHRW) aims to reduce this behaviour. The MHRW does not favour high degree nodes, in contrast the approach taken is to use a calculation which effectively produces an inverse probability of choosing a neighbour which has a high degree.

The pseudo code for MHRW is shown in Table 7.6. The function $Deg()$ simply re-

```
Algorithm SAR ($w_s$, $r$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  $reset = r$;
  set $Q$ to $w_s$; set a counter to one;
  set up the process internal state;
  loop
    increase counter;
    define $v_0$ to be an element of $K_T$;
    $i = 1$; $j = 0$;
    $R = \{v_0\}$;
    repeat
      $v_i \leftarrow$ random neighbour of $v_{i-1}$ NOT in $R$;
      $Q = v_i$;
      add $Q$ to $R$;
      add $Q$ to $K_T$ or $K_N$;
      update the process state;
      if $Q$ added to $K_T$ then
        decrease $j$ (if positive);
      else
        increase $j$;
      end if
      increase $i$;
    until random number $> (reset)^j$;
  end loopcounter goes past MaxIterations
  return $K_T$;
```

Table 7.4: Pseudo code for Self Avoiding Random (SAR) with reset

turns the degree of the given node. It is shown that as the value of $Deg(Q)/Deg(Q_{new})$ increases, the likely hood of *random number* $> Deg(Q)/Deg(Q_{new})$ decreases, thus there is a decreasing chance of not moving, or staying at the current node (Table 7.5). The value of $Deg(Q)/Deg(Q_{new})$ decreases due to the value of the denominator ($Deg(Q_{new})$) increasing, essentially being a larger value in proportion to $Deg(Q)$, in this case, of a higher degree.

Table 7.5: The probability of traversing neighbour Q_{new} based on increasing degree.

$Deg(Q)$	$Deg(Q_{new})$	$\frac{Deg(Q)}{Deg(Q_{new})}$
10	20	0.5
10	30	0.3
10	40	0.25
10	50	0.2

```
Algorithm MHRW ($w_s$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  set $Q$ to $w_s$; set a counter to one;
  set up the process internal state;
  repeat
    redefine $Q_{new}$ to be a random neighbour of $Q$ in $G$;
    if  $random\ number > Deg(Q)/Deg(Q_{new})$ then
      {dont move} $Q = Q$;
    else
      {move} $Q = Q_{new}$;
    end if
    add $Q$ to $K_T$ or $K_N$;
    increase the counter;
    update the process state;
  until counter goes past MAXITERATIONS;
  return $K_T$;
```

Table 7.6: Pseudo code for Metropolis Hastings Random Walk (MHRW)

7.3.5 Random Ordering (RO) Clustering

The Random Ordering (RO) method of traversing a graph G is not strictly a dynamic graph traversal method. This method cannot actually be used practically in a dynamic context, as the graph is needed apriori. This method is presented and used as a comparison against the previous described methods only. The RO method operates in a similar manner to the RW method, but selects a random node from the entire graph G, rather than a neighbour of the current node. The RO is therefore not biased by graph structure or previous states in any way when making a choice of node to select.

7.4 Incremental Clustering

This section presents the incremental clustering methods used with respect to the dynamic approaches in this chapter. The methods presented in this section fit the template of the dynamic approach presented previously in Table 7.1 The clustering algorithms considered are an incremental version of the kmeans algorithm (IKM), presented in sub-section 7.4.1 and the incremental clustering algorithm (ICA) presented in sub-section 7.4.2.

7.4.1 Incremental Kmeans (IKM)

The classic kmeans clustering algorithm (section 2.6.2), which is traditionally considered in a static context, was derived to fit the template given in Table 7.1 by:

1. Assuming that the state of the process contains information about the (two) clusters centroids (K_T and K_N) based on all items.
2. The pages in G are inspected one at a time in some (fixed) given order.
3. The state update is performed after a complete sweep of all the pages.

The loop (lines 3-7) is repeated until the system state does not change from one sweep of the graph to the next (convergence). As already noted, in the dynamic context it is assumed that the graph G is not initially available, but is retrieved from the web incrementally as different pages get requested inside the process main loop. The characteristics of using an incremental version of the kmeans algorithm is that the ordering of items are subject to the traversal of G, which may vary over subsequent iterations, and include repetitions (see graph traversal techniques section 7.3). A new page of the web graph is added to a cluster based on the closest similarity with the cluster centroid. If a previously visited page is traversed, it can be reassigned to a different cluster. The loop is repeated until the system state does not change, or a termination criteria of a maximal number of steps is imposed.

7.4.2 Incremental Clustering Algorithm (ICA)

In addition to IKM a second algorithm is used with respect to the dynamic approaches presented in this thesis, the Incremental Clustering Algorithm (ICA). This algorithm measures the degree of coherency of a cluster based on the Clusters Average Similarity Area (CASA), previously defined in section 2.6.2. A new node is added to a cluster based on lowest CASA calculated by simulating the node being added to each cluster. Upon traversal of a previously clustered node, the node's cluster is reconsidered based on the new calculation of the CASA.

7.5 Graph Representation

This section describes methods that can be used to change the link structure of the internal representation of the visited graph. The methods described can be used dynamically, as the graph is traversed. Table 7.7 presents an example of an edge weight method of graph representation with respect to the dynamic approach (presented above in section 7.2). It will be shown that by utilising a graph representation method the traversal of the graph can be effected or "controlled" to a certain degree by creating extra links or weighting links to change the pathways of the graph.

Each of the methods described in this section can be used with the probabilistic graph traversal methods RW, SAR, MHRW. The underlying link structure is modified as the graph is retrieved, thus creating an interpretation of the graph compatible with any traversal method. The deterministic methods of traversal (BF, DF) can in infact be used with a modified graph, but it will not effect these traversals, as the graph is modified once retrieved (as it is not available apriori), therefore these methods cannot effect a deterministic walk process in advance.

There are two main methods by which the link structure can be modified. The first are edge weighting methods. The edges weights can be assigned using arbitrary values, and can be modified at various points in the traversal of the graph. The second are so called "artificial" edges. Essentially increasing the weight of an edge which was originally equal to 0 to a value > 0. These are edges that can be added between nodes based on a certain criteria, they are not strictly limited to a physical hyper link existing in the original web graph. Each is described in further detail below.

7.5.1 Edge Weighting

An edge weight is assigned to each edge in the structure as it is traversed. The weight can then be increased or decreased as the walk progresses (see Table 7.7). The edge weighting methods presented in this work include:

Similarity Weighting (SW) Edges are reduced in weight depending on the similarity of the connecting nodes being above a threshold. This process is conducted upon each step. Each time an edge is used that is considered to lead from a target node, to a noise node, the edge weight is decreased. This will then reduce the likelihood of a random walk using this edge in the future. The default weighting of edges is 1. This effectively means that all edges have equal probability of traversal. The magnitude of the value whereby edge weights may be decreased was experimentally found to be proportional to the number of connections between clusters (section 7.7.1.2). The weighting of an edge is updated upon traversal of the edge.

```
Algorithm Edge Weighting_template ($w_s$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  set up the process internal state;
  set $Q$ to $w_s$; set $q_i$ to null;
  repeat
    { Edge Weighting }
    if $Q$ and $q_i$ adhere to Edge Weighting conditions then
      set edge $(Q, q_i) = value$;
      set edge $(q_i, Q) = value$;
    end if
    $q_i = Q$;
    select a page $Q$ from $G$;
    add $Q$ to $K_T$ or $K_N$;
    update the process state;
  until convergence;
  return $K_T$;
```

Table 7.7: Pseudo code for generic edge weighting method.

Euclidean Weighting (EW) Edges are weighted inversely proportional to the similarity between nodes, using the actual Euclidean distance between features of the nodes. A random walk has a greater probability of visiting similar nodes, than nodes of increasing dissimilarity.

Cluster Weighting (CW) This method draws on the clusters of the clustering algorithm to reduce the edge weight of edges across the clusters K_T and K_N. At each iteration, the nodes traversed are assigned to either cluster K_T or K_N, depending on the clustering algorithm used. The edges which reside between the nodes of the clusters are extracted, and are subsequently reduced in weight.

Note that all of the above methods work using dynamic graphs, which means that at any two points in the traversal, the graph may have a different number of nodes, and edges subsequent to change.

7.5.2 Similarity Edges

The process of adding an "artificial" edge between two nodes is subject to the similarity between these nodes. These are known as Similarity Edges (SE). As the traversal progresses, upon visiting a previously unseen node, edges are added between this and any visited node that has a similarity above a certain threshold. This creates dense clusters of similar nodes that can subsequently be traversed more often (see Table 7.8). The similarity can be based on a multitude of characteristics, in this work the similarity is based on attributes associated with nodes.

```
Algorithm Similarity Edge_template ($w_s$)
  $K_T = \{w_s\}$; $K_N = \{\}$;
  set up the process internal state;
  set $Q$ to $w_s$; set $q_i$ to null;
  repeat
    { SE Additions }
    if first visit to $Q$ then
      for $q_i = 0$ to number of nodes visited so far do
        if Similarity($q_i$, $Q$) > threshold then
          add a "similarity edge" $(q_i, Q)$ to $G$;
        end if
      end for
    end if
    select a page $Q$ from $G$;
    add $Q$ to $K_T$ or $K_N$;
    update the process state;
  until convergence;
  return $K_T$;
```

Table 7.8: Pseudo code for artificial edges based on similarity.

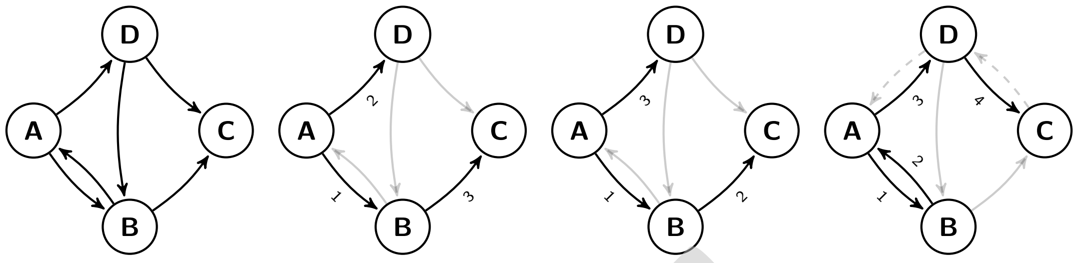
7.6 Issues of the Dynamic Approach to WBD

This section presents some issues and characteristics of the dynamic approaches to resolving the WBD problem. The issues and characteristics are given in the list below. Each point in the list is referred back to in the evaluation presented in the following section.

- Web graph coverage
- Dynamic WBD performance
- Web graph visitation ordering
- Variability of Probabilistic Graph Traversals

Web Graph Coverage The coverage of a web graph is an important issue with respect to dynamic WBD (see Figure 7.3). The amount of a graph that is covered directly effects how much of a graph needs to be potentially retrieved from the web. The coverage issue becomes more apparent when comparing the WBD performances relative to the amount of noise and target pages that have been visited, as each page visit comes with an associated retrieval and pre-processing cost. A simple example is shown in Figure 7.3. Notice the number of steps that each graph traversal takes to cover (or visit) the nodes of the graph.

Figure 7.3: Traversal of the graph in (a) are shown in the corresponding Sub-Figures (b, c and d). The traversals are; Breadth First (BF) in (b), Depth First (DF) in (c) and Random Walk (RW) in (d). The steps of the traversal are shown as edge labels and the corresponding nodes visited are noted in each example.



(a) Graph (b) BF *(A,B,D,C)* (c) DF *(A,B,C,D)* (d) RW *(A,B,A,D,C)*

Dynamic WBD Performance As outlined in section 3.4.3 the performance of a WBD solution is measured relative to a given seed node (w_s). The approaches produced in this chapter are dynamic, which means that the cluster compositions (K_T and K_N), which reflect the WBD solution produced, can change at any step of the graph traversal. This can cause variability in the WBD performance over time.

Web Graph Visitation Ordering The ordering in which pages are considered by the incremental clustering method is completely influenced by the type of graph traversal method used. Figure 7.3 shows some examples of graph traversals and the ordering of nodes produced. The deterministic methods of traversal (BF and DF) follow edges in a fixed ordering until visitation of all pages of the graph is complete (Figures7.3b and 7.3c). This fixed ordering of pages is then iterated over repeatedly until the clustering algorithm converges. An example is shown in Figure 7.4. Notice the end converged state can be continued for any number of steps. Iteration until a maximal number of steps will be used for the comparison of both the probabilistic and deterministic methods presented in the following evaluation.

Variability of Probabilistic Graph Traversals The approaches that are based on probabilistic traversals (RW, SAR, MHRW and RO) of the graph can produce WBD solutions with varying degrees of performance. On average, there is some consistency in performance of the probabilistic based dynamic approaches. An example is shown in Figure 7.5 where the grey lines depict the variability in coverage of each RW. In the evaluation presented in this chapter an average performance is reported for each of the approaches based on probabilistic graph traversals.

Figure 7.4: Accuracy with respect to the traversal, converging and converged phases of a BF walk on ADG Set1 G1 (see section 4.2).

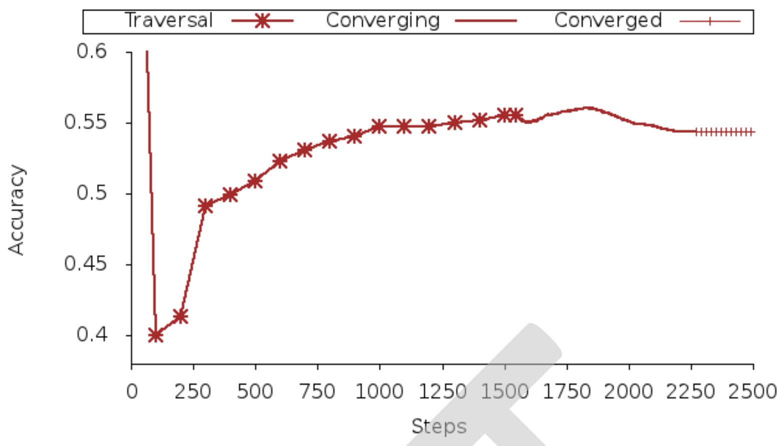
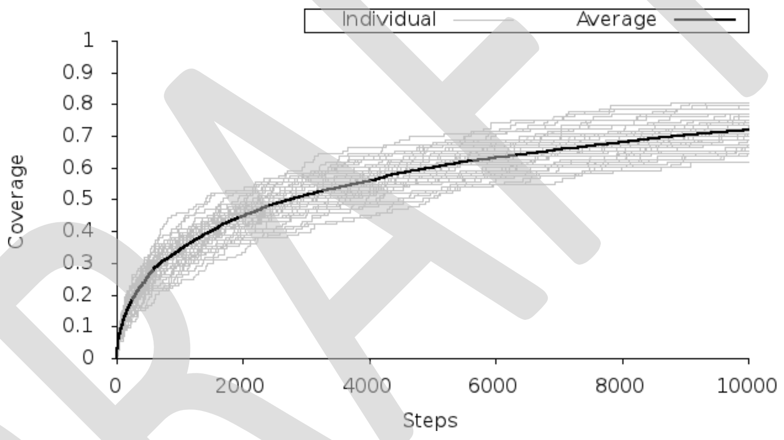


Figure 7.5: Coverage variability of 50 random walks on ADG Set1 G1 (see section 4.2). The grey lines indicate the coverage of individual RWs, and the black line the average coverage.



7.7 Evaluation

This section reports on the evaluation of the dynamic approaches to the WBD problem presented in this chapter. The evaluation of the approaches is presented in the following three sections 7.7.1, 7.7.2 and 7.7.3. Each of the three sections reports on a conducted evaluation using different models of the WBD problem but of increasing complexity. The WBD problem was modelled in three ways using: Binomial Random Graphs (BRGs), Artificial Data Graphs (ADGs) and Real Data Graphs (RDGs) which were previously introduced in sections 4.1, 4.2 and 4.3. The complexity of the dynamic approaches were devised to provide solutions to the WBD problem so as to reflect the complexity of the WBD model used in each of the sections. The findings from each of the evaluations presented in each of the three sections were used to re-enforced the approaches described in subsequent sections of evaluation. The evaluation presented in this chapter does not provide an exhaustive list of every graph traversal technique coupled with every graph representation used with all methods of producing clusters as described earlier in this chapter. Rather the evaluation in this chapter aims to highlight the most interesting and appropriate approaches with respect to WBD in the dynamic context. The strategy used to evaluate the dynamic approaches with respect to each model of the WBD problem is given below. The evaluation questions that are considered in this section are as follows:

- The performance of graph traversal techniques at detecting clusters using simplified graph scenarios.
- The graph coverage ratios of target and noise nodes and the influence graph features have on these ratios.
- The WBD performance of dynamic techniques using RDG data sets.

Evaluation Strategy The evaluation strategy used in this chapter was based on devising dynamic approaches to the WBD problem as modelled using increasingly complex data sets. The evaluation strategy is presented in three sections, corresponding to each of the three data sets used. Each method of evaluation is explained in further detail as follows:

- Binomial Random Graph (BRG) Evaluation

  The first section (section 7.7.1) presents an initial evaluation of the characteristics of a RW graph traversal method using a simplified model of the WBD problem using BRGs. The evaluated dynamic methods were RW and RW-SW used in combination with a buffer to store the ordering of pages produced by the graph traversals. The buffer is used to calculate a ratio value which can be used to detect cluster transitions in the BRG data sets. The evaluation presented in this section demonstrated that RW can be a powerful method for traversing the graph. The RW method exhibited the characteristic of frequently visiting highly connected pages detecting cluster transitions with respect to the WBD problem. In the following evaluation explained below, the characteristics of the RW were re-enforced and it was shown that when the graph traversal technique is combined with an incremental clustering algorithm (in contrast to a buffer) a high performing WBD solution can be produced.

- Artificial Data Graph (ADG) Evaluation

  The second section 7.7.2 presents an evaluation using the ADGs to model the WBD problem, which is based on the preferential attachment model. The ADGs exhibit an increased complexity compared with the BRG evaluation as explained above. The dynamic approaches to the WBD problem, RW and BF, were initially compared in terms of WBD performance using both the ICA and IKM algorithms. The higher performing IKM algorithm was used to compare RW graph traversal using graph representations SE, SE-SW, EW, CW with BF, DF, SAR and RO. The graph representation methods evaluated show that it is possible to control the graph traversal of a dynamic approach to WBD such that the graph coverage of target and noise pages can be influenced to increase the performance of the WBD solution while decreasing the cost of processing unwanted noise. The RO method produced a high performing WBD solution if the number of target and noise pages in the graph are not proportionally large. The RO cannot strictly be deployed in a dynamic context, as the entire graph is needed apriori. In the evaluation presented below, dynamic approaches were evaluated using RDGs. In a bid to recreate the RO method that can be applied in a dynamic context, the MHRW method of graph traversal was evaluated.

- Real Data Graph (RDG) Evaluation

  The final section 7.7.3 presents the evaluation using the RDGs. The evaluated dynamic approaches to the WBD problem were BF, DF, RW, RO, MHRW and SAR in combination with the standard hyperlink graph representation using the IKM algorithm.

Each of the dynamic approaches and corresponding data sets used for the evaluation listed above is presented in Table 7.9. The category of data set used is given on the left and in further detail in the data set column. The graph traversal and corresponding graph representation refer to the combination of methods used for the particular dynamic approach (as previously explained in section 7.3 and 7.5 respectively). In the table n/a indicates that a graph representation was not applicable, as in the case of BF and DF which are deterministic traversals unaffected by the graph representations; std indicates that the standard hyperlink structure is used to represent the graph. The incremental clustering column indicates the method that was used to detect clusters, and subsequently website boundaries.

7.7.1 Binomial Random Graphs Evaluation

The evaluation in this section presents results using a simplified model of the WBD problem implemented using Binomial Random Graphs (BRGs). The BRGs model the WBD problem as a binary cluster problem, producing two clusters; one cluster represented the collection of pages in a website boundary (target), and the other represented unwanted (noise) pages. There are two specific characteristics relevant to WBD when generating a binomial random graph, namely:

- The density of connections inside the two clusters

Table 7.9: The evaluation strategy for the dynamic approaches to the WBD problem. The three categories of data set are given on the left hand side.

	Graph		**Incremental**	
	Traversal	**Representation**	**Clustering**	**Data set**
Binomial Evaluation	RW	std	Buffer Ratio Calculation	Complete, partial BRGs
	RW	SW	Buffer Ratio Calculation	Feature BRGs
Artificial Evaluation	RW	std	ICA	ADGs set 1 and set 2
	RW	std	IKM	
	BF	n/a	ICA	
	BF	n/a	IKM	
	DF	n/a	IKM	
	RW	std	IKM	
	RW	SE	IKM	
	RW	SE-SW	IKM	
	RW	EW	IKM	
	RW	CW	IKM	
	SAR	std	IKM	
	RO	std	IKM	
Real Data Evaluation	BF	n/a	IKM	RDGs LivChem, LivHistory, LivMath and LivSace
	DF	n/a	IKM	
	RW	std	IKM	
	MHRW	std	IKM	
	SAR	std	IKM	

- The number of connections across clusters

The density of connections inside the two clusters used in this evaluation was either: (1) complete, or (2) partial. The complete clusters used connections between every page to create dense clusters. The partial clusters have connections between pages according to a probability (0.5). The connections across clusters used in this evaluation are connected at random, the number used was either: 5, 50 or 500 connections. Varying the density of clusters and the connections across the two clusters changed the graph from exhibiting strong or weak relationships between clusters, in effect blurring boundaries between the target website pages, and unwanted noise pages.

A cluster transition is when a walk traverses edges of a graph that lead from one cluster into a different cluster. In terms of WBD, this is when a walk moves out of the cluster of target website pages to a cluster of noise pages. This essentially means the walk is leaving or "crossing" the website boundary.

As the pages of the graph are walked (or traversed), a buffer of size b is used to record the visited pages. The buffer initially contains no pages, as the walk progresses the buffer is filled. Once its capacity b is reached, the first page that was added is removed to make room for a new page (the buffer is essentially a first in first out queue).

A ratio calculation was computed in real time using the nodes contained in the buffer. The ratio is calculated by first counting the unique pages in the buffer and dividing this amount by the size of buffer b.

The ratio value relies on the fact that the RW will traverse nodes of a single cluster more often. This in turn populates the buffer with a continual stream of pages from a single cluster. The ratio value at this point will be steady. When the RW makes a transition to a different cluster, a stream of new pages start to enter the buffer, this increases the ratio value until it again reaches a steady point when the RW remains in the new cluster for a period of time.

It will be shown that this calculation can be used effectively to detect a cluster transition, which is indicated by a spike in the calculated ratio value. The buffer size used was $b = 500$, this was found experimentally to be proportional to the size of nodes in the input graph.
The two types of BRG that were used in this evaluation:

1. Simple BRGs using a varying number of connections inside and between clusters simulating density.

2. Feature BRGs using varying connections inside and across clusters, with the pages of a cluster assigned a feature representation.

The evaluation in sub-section 7.7.1.1 used Simple BRGs. The evaluation presents the RWs characteristics to detect cluster transitions, based on the structural properties of the graph, using a buffer to store previously visited pages. A ratio value calculated using the buffer indicated the transition of the RW moving from one cluster to another cluster. The transition between clusters simulated the crossing of the boundaries of a website.

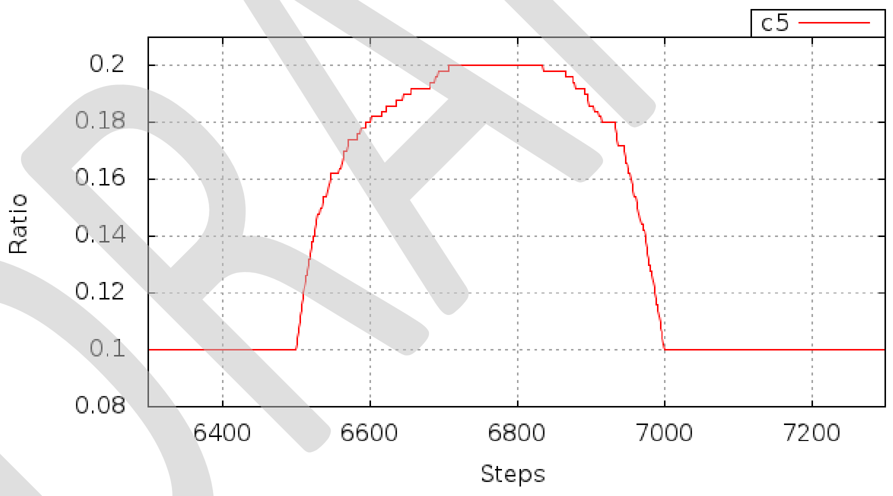
The evaluation in sub-section 7.7.1.2 used Feature BRGs. The evaluation presents the characteristic of the RW method when combined with SW (Similarity Weighting) to detect cluster transitions when additional complexity is introduced to blur the distinction of clusters in the BRGs. This was done by decreasing connections inside clusters and increasing connections between clusters of the BRGs.

7.7.1.1 Simple Binomial Random Graphs Evaluation

The output ratio value calculated using a RW on the Simple BRG C5 data sets is shown in Figure 7.6. The Figure clearly shows a spike in the ratio value that starts around 6500 steps, peaks at 6800 and then returns to normal around 7000 steps. The ratio value increases when pages are encountered that have not been seen before, then decreases once the pages in the buffer are no longer unique. The "resting" ratio value of 0.1 is output while the walk remains inside a single cluster for a period of time. The spike in the value is caused by an influx of unique pages into the buffer, thus moving from a constant traversal of pages in a single cluster, to visiting "new" pages in a different cluster. The maximal value of the spike (0.2) indicates the walk being halfway between clusters.

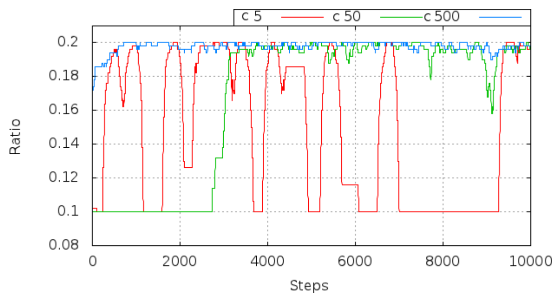
Complete clusters Figure 7.7 shows the ratio output for a RW on graphs BRG C5, C50 and C500, which has an increasing number of connections between clusters (5, 50 and 500 connections respectively). The Figure shows that the clear spikes in ratio value to indicate cluster transitions for graph C50 and C500 are not clear as in C5 (as highlighted in Figure 7.6). The behaviour of the RW on graph C50 and C500 is not consistent because of the increased number of connections between each of the clusters. The random walk transitions erratically between clusters, this makes it difficult for the ratio output to exhibit clear cluster transitions. This shows the variability and nature of the RW, which is effected directly by the structure of the graph.

Figure 7.6: The ratio value output of a RW using data set BRG C5. This plot is a segment of the plot in Figure 7.7a (between steps 6400-7200).

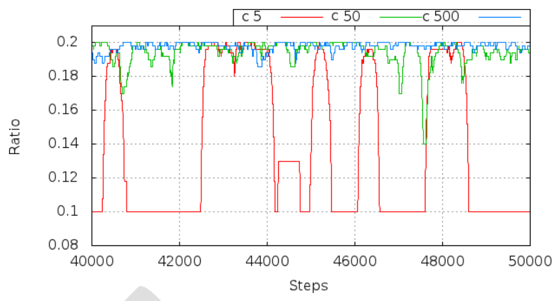


Partial clusters The partial BRGs used in this evaluation have less structural cohesion inside clusters, while the number of connections between clusters remains the same as the complete BRGs used above. It is evident that when the number of connections between clusters is low (P5 in Figure 7.8), it is possible to detect cluster transitions indicated by spikes in the ratio values, this is more prominent in the long run (shown in Figure 7.8b). As the number of connections increases from 5 to 50 to 500 (P5, P50 and P500 respectfully in Figure 7.8) the number of cluster transitions made by the RW also increases. It can also be seen that there is a "settling down" characteristic of the

Figure 7.7: The ratio values for RWs using data set Simple BRG C5, C50 and C500 with complete clusters. Plot (a) shows the earlier stages of the walk (0-10k steps) and plot (b) shows the later stages (40k-50k steps).



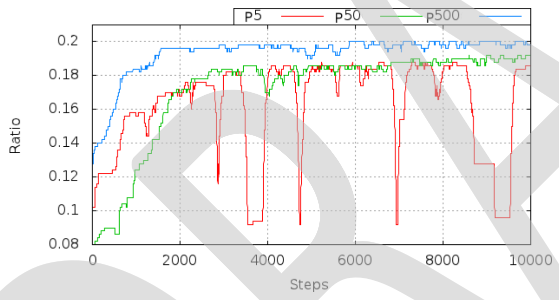
(a) 0-10k Steps



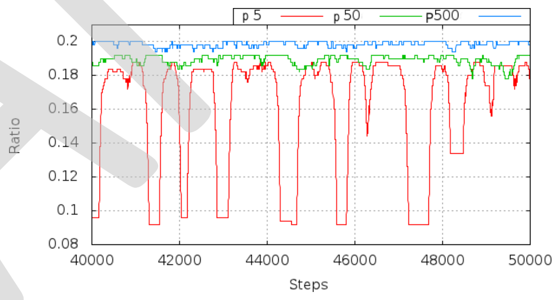
(b) 40k - 50k Steps

RW, when the ratio values recorded in the earlier stages of the walk are compared to the later stages, the walk is a lot less erratic, as indicated by the more stable ratio output (see Figures 7.8a to 7.8b).

Figure 7.8: The ratio values for RWs using data set Simple BRG P5, P50 and P500 with partial clusters. Plot (a) shows the earlier stages of the walk (0-10k steps) and plot (b) shows the later stages (40k-50k steps).



(a) 0-10k Steps



(b) 40k - 50k Steps

The evaluation in this sub-section presented the characteristics of the RW, which showed that using only the structural properties of (complete and partial) BRGs, RW can be used to detect cluster transitions when the connections between clusters remain low. In the evaluation in the following subsection, the RW method is further extended to include Similarity Weighting (SW) graph representation method. It is shown that biasing the edges with similarity weighting means that the RW can be "controlled" to the extent to detect cluster transitions even in the worst case scenario of 500 connections between clusters.

7.7.1.2 Feature Binomial Random Graphs Results

In the evaluation presented in this sub-section, Feature BRGs are considered. In contrast to the simple BRGs used in the evaluation above, the Feature BRGs had a set of (similar) attributes associated with pages of each cluster, the structural cohesiveness between pages within the same cluster remained low (see section 4.1.2 for further details). The Feature BRGs had the following characteristics associated with them with respect to the detection of cluster transitions:

1. Low cohesiveness inside clusters
2. Increasing amount of connections across clusters

Both decrease the probability of a RWs next step being a related page in the same cluster as the previous page. This reduces the amount of time spent traversing pages of a single cluster, and increases the number of cluster transitions. The RW graph traversal method with Similarity Weighting (SW) graph representation is evaluated on the BRGs F5, F50 and F500 with respect to the two issues above. The SW method calculates and adjusts the weighting of edges dynamically as the RW traverses and explores the graph structure. In the first instance of the RWs traversal, each edge used is given a standard weight value. The weighting of all available edges from a page determines the probability of taking that particular edge. Intuitively an edge with lower weight is less likely to be chosen. Upon traversal of an edge from page a to b, if page a and b are deemed dissimilar using a threshold value, then the weighting of the edge is reduced. The factor to reduce the edge weights by was found experimentally to be proportional to the connections across clusters (0.5). The SW method has the effect of "controlling" the RW, by steering the direction it takes by reducing the weight of "bad" edges leading to pages of low similarity to the current page.

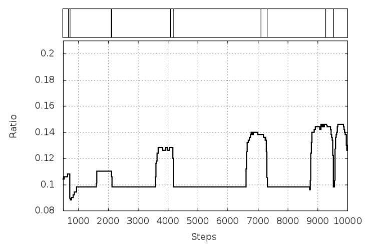
Figures 7.9, 7.10 and 7.11 show the ratio value for the RW-SW approach using the feature BRG F5, F50 and F500 respectively. The Figures show that the cluster transitions (lines above ratio plot) were greatly reduced using the SW method, this is illustrated when comparing the cluster transitions in the initial stages of the RW (Figures 7.9a, 7.10a and 7.11a) to the later stages of the walk where the weightings of edges between unrelated nodes have been reduced (Figures 7.9c, 7.10c and 7.11c).

The RW-SW approach applied to all three feature BRGs (F5, F50 and F500) shows that after a period of time the edge weights of unrelated nodes are reduced to the point where the RW is highly biased towards edges to related nodes, these edges are chosen by the RW with a much higher probability.

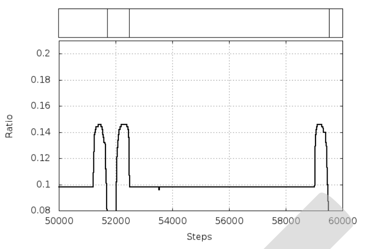
7.7.1.3 Summary

The evaluation presented in this section concentrated on the RW and RW-SW approaches to WBD using a simplified model of the WBD problem implemented using Binomial Random Graphs (BRGs). The evaluation presented in this section illustrated that:

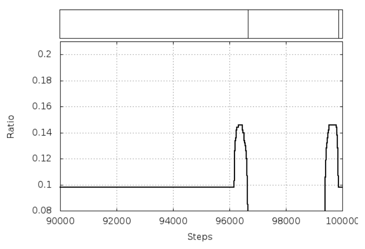
Figure 7.9: Ratio values calculated for RW-SW using data set Feature BRG F5. The plot shows both the ratio values and the cluster transitions (spikes) in the line (above). Each plot (a,b,c) shows the progress of the random walk at various step segments as indicated.



(a) 0-10k Steps

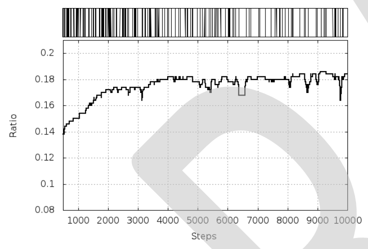


(b) 50k - 60k Steps

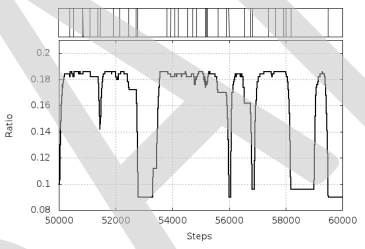


(c) 90k - 100k Steps

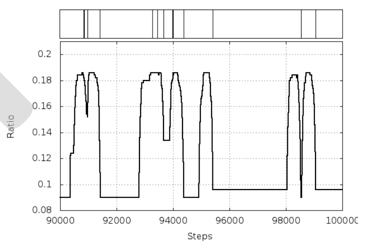
Figure 7.10: Ratio values calculated for RW-SW using data set Feature BRG F50. The plot shows both the ratio values and the cluster transitions (spikes) in the line (above). Each plot (a,b,c) shows the progress of the random walk at various step segments as indicated.



(a) 0-10k Steps

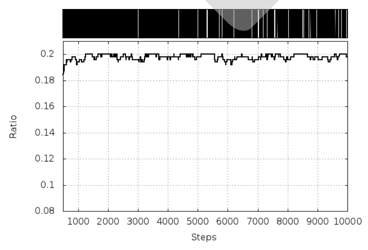


(b) 50k - 60k Steps

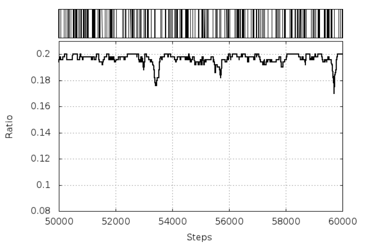


(c) 90k - 100k Steps

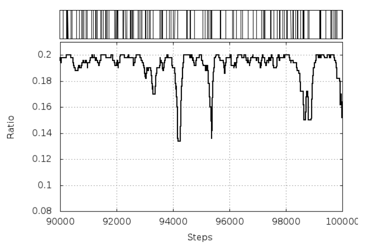
Figure 7.11: Ratio values calculated for RW-SW using data set Feature BRG F500. The plot shows both the ratio values and the cluster transitions (spikes) in the line (above). Each plot (a,b,c) shows the progress of the random walk at various step segments as indicated.



(a) 0-10k Steps



(b) 50k - 60k Steps



(c) 90k - 100k Steps

1. The ordering of pages selected by the RW exhibits frequent traversal of highly connected or dense sub-regions of pages in a graph.

2. The ordering of pages produced by the RW can be used to detect cluster transitions, by using a buffer and calculated ratio value.

3. The graph representation method SW can be used to improve the ability of RW to remain inside a cluster when a cluster is adversely connected, for example when connections across cluster are proportionally high, or connections inside a cluster are proportionally low.

The evaluation in this section reported that the RW can be a powerful tool for traversing a graph in such a way so as to exploit the cohesion of connected pages. The findings in this evaluation are further re-enforced by the evaluation in the following section 7.7.2, which reports on dynamic approaches to WBD with respect to a more complex model using Artificial Data Graphs (ADGs) of the WBD problem. In the following section it is shown that when the graph traversal technique is combined with an incremental clustering algorithm (in contrast to a buffer) a high performing WBD solution can be produced. Various graph representation methods are also considered to "control" the random traversal of the graph when traversing the more complex Artificial Data Graphs (ADGs).

7.7.2 Artificial Data Graphs

The evaluation in this section reported on results using Artificial Data Graph (ADGs) that were synthetically generated based on the preferential attachment model (see section 4.2 for further details). The ADGs data sets were a more sophisticated technique to model the WBD problem compared with the simplistic BRGs used in the previous section 7.7.1 evaluations. The ADGs increase the complexity of the WBD problem by modelling a more "web like" graph of pages. The evaluation reported in this section used data sets ADGs set1 and set2. There are two main differences between the graphs in set1 and those in set2. First, typically graphs in set2 contain fewer edges and, individually, their pages contain fewer out-links. Second, for graphs in set1 the target class coincides with the set of all pages at a diameter [1] at of most three. For the graphs in set2 the target website pages have a diameter of at most five, with only half of the neighbours of a page in the target cluster. The target cluster of set2 is said to be more "stringy" than that of set1. The ADG set1 and set2 reflect the problem of discovering varying sized website boundaries with respect to varying noise conditions.

Re-enforcing the approaches used in the previous evaluation, a graph traversal was used to produce an "ordering" of pages as each page is visited. In contrast to the

[1]Diameter is the greatest distance between any pair of pages in the graph.

evaluation of the previous approaches, the pages visited are not stored in a buffer (to perform ratio calculations), but are instead used by an incremental clustering algorithm. The incremental clustering algorithm performs clustering in real time based on the similarity of attributes of the pages, as each of the pages are visited. An output clustering is produced by the algorithm providing a solution to the WBD problem for every "step" of the graph traversal. The WBD performance of various graph traversal and representation methods were evaluated with respect to three main outcomes:

1. The comparison of two incremental clustering algorithms, ICA and IKM , in terms of WBD performance.

2. The WBD performance of graph representations (SW, EW, CW and SE) with respect to the RW graph traversal method.

3. The comparative performance of the RO method in terms of WBD solution produced.

The evaluation presented in sub-section 7.7.2.1 provided a comparison of the ICA and IKM clustering algorithms with respect to the BF and RW graph traversal methods in terms of WBD solution performance. The IKM clustering algorithm exhibited an increased performance over the ICA in terms of the WBD solutions produced. The IKM algorithm was subsequently used for the remaining evaluation reported in this chapter.

The evaluation presented in sub-section 7.7.2.2 provides a comparison of the various graph representations using the RW graph traversal (RW-SE, RW-SW, RW-SE-SW, RW-EW and RW-CW) with respect to the BF, DF and SAR methods. The characteristic behaviour of the graph representations using RW were evaluated with respect to graph coverage, and performance of the WBD solution produced. The evaluation of the graph representation methods demonstrated that it is possible to control the graph traversal when adopting a dynamic approach to WBD so that the graph coverage of target and noise pages can be influenced in such a way so as to increase the quality of the WBD solution, while decreasing the cost of processing unwanted noise.

The evaluation presented in sub-section 7.7.2.3 reports on the WBD performance of the RO method of selecting pages from the graph. The RO method selects pages at random from all pages in the graph independent of the link structure. It is shown that the RO method proves to provide the best WBD solution when the number of pages is relatively small.

7.7.2.1 ICA and IKM Clustering comparison

The evaluation presented in this sub-section provided a comparison of the ICA and IKM clustering algorithms with respect to the BF and RW graph traversal methods in terms of WBD solution performance. The performance comparisons were made relative to each of the clustering algorithms WBD performance with respect to a particular graph traversal method. This ensured that the comparison was made based according to how the clustering algorithm performed using the exact same ordering from a graph traversal method in terms of the WBD solutions produced. The evaluation shows that the IKM method produced comparative or improved WBD performance over the ICA method relative to the the BF and RW graph traversal methods.

The average WBD performance of the BF and RW approaches, using ICA and IKM for data sets ADG set1 and set2, are shown in Figures 7.12 and 7.13 respectfully. Table 7.10 summarises the WBD solutions in terms of score performance. The Table shows the IKM method produced either a higher or comparable WBD performance score when compared to the ICA method.

Figures 7.12 and 7.13 show the "history" of the graph traversal using both clustering algorithms as the pages of the graph were visited step by step. The history of the walks show how the performance of the WBD solutions changes as the walk proceeds. The BF method of traversing the graph produced an ordering of pages such that both the ICA and IKM algorithms effectively converge in under 2k steps. The RW method of traversal, however, produces an ordering of pages such that the pages were constantly being randomised therefore the algorithms do not converge within 10k steps.

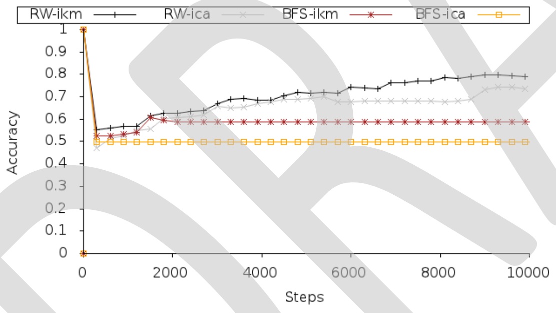
The accuracy of the WBD solution produced by the BF traversal using IKM is shown to outperform that of ICA (Figure 7.12 and 7.13), this indicated that the IKM algorithm groups pages in a more beneficial way compared to ICA in terms of WBD using the BF ordering of pages. Due to the randomised ordering of pages produced by the RW graph traversal method, the accuracy of the WBD solution constantly increases relative to the number of steps (Figures 7.12a and 7.13a). It is shown that IKM produced either: (1) a similar measure of accuracy as ICA (Figure 7.13a), or (2) IKM outperforms ICA over a prolonged number of steps (Figure 7.12a).

The precision measurement indicated whether the target cluster (K_T) contains a high number of target pages compared to noise pages, as shown in Figures 7.12c and 7.13c. The high starting value of precision indicated that the initial condition where the target cluster (K_T) contained only the seed page (w_s). The sharp decline represents the grouping of visited noise and target pages influenced by the exploration of the graph traversal method. The measurements of precision was consistent for ICA and IKM using the RW traversal method (Figures 7.12c and 7.13c), this indicated that the WBD solution produced using the RWs ordering of pages is similar irrespective of using ICA or IKM. The BF method of traversal shows a lower value of precision using ICA over IKM for ADG set1 (Figure 7.12c), this indicated that ICA could not group pages as well as IKM using the BF ordering of pages in terms of WBD solution.

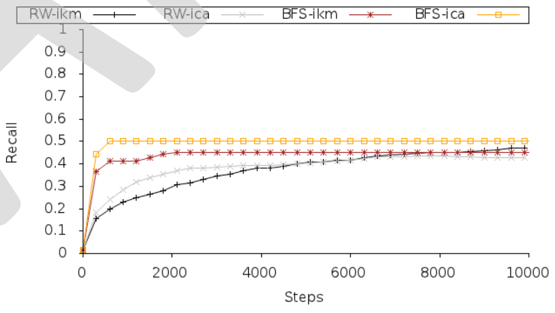
The recall measurement indicated whether the target cluster (K_T) included all target pages contained within a websites boundaries (C_T) or not, as shown in Figures 7.12b and 7.13b. The low initial value of recall indicated that the target cluster (K_T) included only the seed page (w_s). The rise in recall value represents the grouping of target pages in the target cluster as the traversal method visits pages of the graph. The recall value for both ICA and IKM favours the BF method over RW in the initial stages of the graph traversal, however it is shown that in the later stages fairly constant values of recall are produced (Figures 7.12b and 7.13b).

This sub-section has reported on the evaluation of the IKM and ICA methods to cluster a graph for the purpose of WBD. The incremental clustering methods were evaluated using both BF and RW graph traversal methods, the effectiveness in terms of WBD was determined using a number of performance measures. The evaluation showed that the IKM method produced a comparative or improved performance over the ICA method using the BF and RW graph traversal methods. The IKM algorithm was subsequently used for the remaining evaluation in this chapter. The following sub-section 7.7.2.2 presents an evaluation using various graph representation techniques in terms of WBD performance using the IKM algorithm.

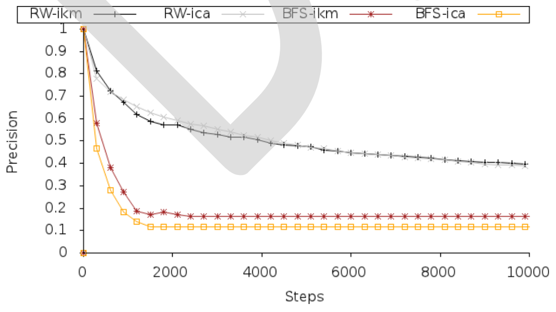
Figure 7.12: The average WBD performance of BF and RW graph traversal methods using both incremental clustering algorithms IKM and ICA on the data set ADG Set1 (see section 4.2).



(a) Accuracy

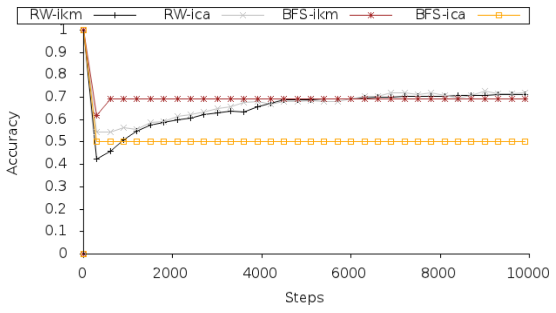


(b) Recall

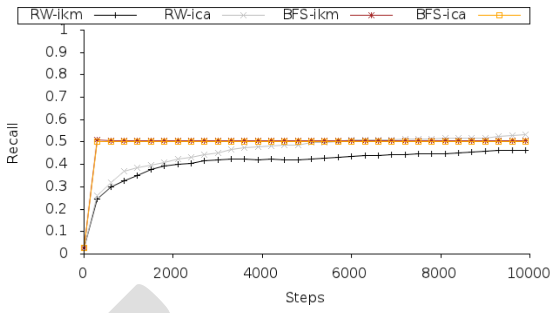


(c) Precision

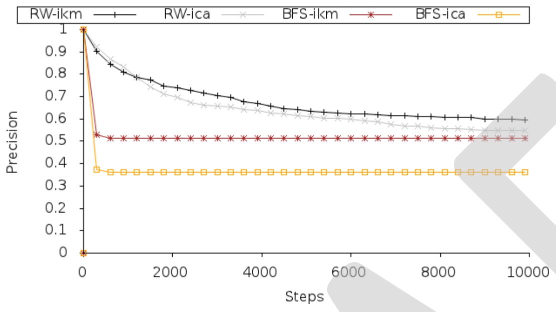
Figure 7.13: The average WBD performance of BF and RW graph traversal methods using both incremental clustering algorithms IKM and ICA on the data set ADG Set2 (see section 4.2).



(a) Accuracy



(b) Recall



(c) Precision

7.7.2.2 Graph Representation comparison

The evaluation in this sub-section compared the various graph representations using the RW graph traversal (RW-SE, RW-SW, RW-SE-SW, RW-EW and RW-CW) with respect to the BF, DF and SAR graph traversal methods. The characteristic behaviour of the graph representations using RW were evaluated with respect to graph coverage, and performance of the WBD solution produced. The graph representation methods evaluated show that it is possible to control the graph traversal of a dynamic approach to WBD such that the graph coverage of target and noise pages can be influenced to increase the performance of the WBD solution while decreasing the cost of processing unwanted noise. The graph representation has an increased cost attached which is associated with the particular mechanism used to control the traversal. The cost of the graph representation, in some cases, does not provide a justifiably significant improvement in performance with respect to the WBD solution produced when compared with the simplistic implementation and low cost of the alternative dynamic approaches evaluated.

The dynamic approaches presented in this sub-section use the IKM algorithm to produce a WBD solution as it was proven more effective in the previous sub-section 7.7.2.1. A substantial number of experiments were conducted using numerous combinations of each edge representation using various adjustments and parameters. In

Table 7.10: A comparison of ICA and IKM algorithms using the BF and RW graph traversal in terms of WBD score performance. Score values are shown for 2k and 10k steps for BF and RW respectively.

	Graph Traversal	**Clustering Algorithm**	**Score**
ADG Set1	BF	IKM	0.585
		ICA	0.497
	RW	IKM	0.791
		ICA	0.726
ADG Set2	BF	IKM	0.691
		ICA	0.501
	RW	IKM	0.729
		ICA	0.712

some cases the conducted experiments generated results which either; (1) did not provide significant variation from a base line, or (2) did not provide significant insight in to the WBD problem. As a consequence, these tests have been omitted from the narrative presented in this section. The evaluated approaches that are reported in this sub-section are listed as follows:

- Breadth First (BF)
- Depth First (DF)
- Random Walk (RW)
- Random Walk using Similarity Weighting (RW-SW)
- Random Walk using Similarity Edges (RW-SE)
- Random Walk using Euclidean Weighting (RW-EW)
- Random Walk using Cluster Weighting (RW-CW)
- Random Walk using Similarity Edges and Similarity Weighting (RW-SE-SW)
- Self Avoiding Random (SAR) Walk

The best performing dynamic approaches presented in this evaluation were shown to be RW-SE-SW and RW. The dynamic approach that covered the least amount of noise was shown to be RW-CW, but this method has a considerable cost associated with the Cluster Weighting (CW) graph representation and does not produce a high performing WBD solution. The RW dynamic approach produced the overall best WBD solution when considering the amount of noise covered with respect to the lowest cost and consistent performance for the data sets used for evaluation. The RW method was deemed the most appropriate method due to its simply operation, which is fast and does not require a high resource cost in terms of graph representation or selection of edges to traverse.

Table 7.11: The graph traversal and graph representation approaches (as indicated) ordered according to WBD performance score. The average graph coverage, WBD performance score and average time per step is shown for each approach using data set ADG Set1.

	Coverage				**Time(ms)/**	**Total**
	Target	**Noise**	**Total**	**Score**	**Steps**	**Steps**
RO	1.000	1.000	1.000	0.779	0.191	50000
RW-SE-SW	0.997	0.703	0.845	0.703	0.212	50000
RW	0.997	0.887	0.940	0.657	0.174	50000
RW-SE	0.995	0.692	0.838	0.653	0.167	50000
SAR	0.999	0.649	0.818	0.643	0.189	50000
RW-EW	0.997	0.869	0.931	0.637	0.234	50000
RW-SW	0.987	0.879	0.921	0.627	0.224	50000
DF	1.000	1.000	1.000	0.615	3.047	287
RW-CW	0.894	0.395	0.636	0.601	12.148	50000
BF	1.000	1.000	1.000	0.556	3.215	287

Table 7.12: The graph traversal and graph representation approaches (as indicated) ordered according to WBD performance score. The average graph coverage, WBD performance score and average time per step is shown for each approach using data set ADG Set2.

	Coverage				**Time(ms)/**	**Total**
	Target	**Noise**	**Total**	**Score**	**Steps**	**Steps**
DF	1.000	1.000	1.000	0.859	5.642	466
RO	1.000	1.000	1.000	0.766	0.259	50000
RW-SE-SW	0.959	0.459	0.731	0.745	0.262	50000
RW	0.979	0.633	0.821	0.676	0.184	50000
RW-EW	0.980	0.643	0.827	0.669	0.235	50000
RW-SW	0.970	0.663	0.827	0.659	0.225	50000
RW-SE	0.959	0.479	0.740	0.658	0.210	50000
BF	1.000	1.000	1.000	0.625	4.869	466
RW-CW	0.659	0.270	0.482	0.614	27.250	50000
SAR	0.984	0.638	0.826	0.557	0.175	50000

Tables 7.11 and 7.12 show the experimental results of each of the dynamic approaches listed above with respect to WBD performance for the data sets ADG set1 and set2 respectfully. The Tables show the graph coverage in terms of target, noise and total, the WBD performance score, and average time taken per step for each of the approaches. The Random Ordering (RO) method is include in the Tables, see the following section 7.7.2.3 for the evaluation of the RO approach.

The top performing graph representation in terms of WBD performance was the RW-SE-SW approach. The evaluation shows consistent performance on ADG set1 and set2 (Tables 7.11 and 7.12). The top performance of graph representation RW-SE-SW using both Similarity Edges (SE) and Similarity Weighting (SW) compared to when used individually (RW-SE and RW-SW), showed an increase in performance. The RW-SE-SW method exhibited an increased average time to complete a step as a consequence of the graph representation used (SE and SW). The RW method produced a WBD performance score just below that of the RW-SE-SW method. The RW method showed a much smaller average time to complete a step, this is due to its comparatively simplistic operation.

The RO method has a good comparative performance on both ADG set1 and set2. The high performance of the RO method is due to its randomised clustering approach which is independent of graph structure. This allows the clustering algorithm to randomise the ordering of nodes it clusters on a constant bases which increases cluster performance. The DF method performed better with respect to set2 than set1 because of the underlying graph structure of set2. The connections of target and noise clusters have fuzzy edges (clusters are more stringy) in set2. This allows the DF method to gain an improved initial clustering compared to other methods, as a deep crawl is initially performed.

Figures 7.15 and 7.17 illustrate the graph coverage history of the RWs and SAR approaches for data sets ADG set1 and set2 respectfully. Figures 7.14 and 7.16 show the history of the RWs and SAR methods in terms of WBD performance.

RW-CW is consistently the worst performer in terms of recall for both ADG set1 (Figure 7.14b) and set2 (Figure 7.16b). In contrast it is the best performer in terms of precision (Figures 7.14c and 7.16c), which can be explained using the graph coverage measures. The overall coverage of the graph using the RW-CW approach is much lower than the other approaches evaluated. The ratio of items covered includes a much larger amount of target than noise web pages. This will increase the precision because the ratio of target to noise pages in the website is higher, while the amount of target pages included in the website boundary is low compared to the target pages contained in the website.

The RW-SE and RW-SE-SW approaches perform consistently across set1 and set2 in terms of precision (Figures 7.14c and 7.16c), recall (Figures 7.14b and 7.16b) and coverage (Figures 7.15a and 7.15a). The RW-SE-SW approach produced a slightly higher accuracy for the graphs in set2 (Figure 7.16a) than in set1. This implies that adding Secondary Edges (SE) improves the performance in terms of the WBD solutions produced when the graphs have less connected target pages. Whereas the SE method provides no observable benefit when the amount of target pages in the graph are densely connected, although the Similarity Weighting (SW) method improved the overall WBD performance, shown in set1 (Figure 7.14a),

The DF method outperforms other methods in terms of WBD performance score when evaluated using ADG set2. The high performance of the DF method can be attributed to the fact that the graphs structure of set2 lends its self to a depth first crawl. The amount of noise and target web pages are crawled with respect to an overall view of the data, which is due to the deep search of the DF method. This ordering is much more beneficial for a clustering algorithm with respect to WBD.

The RW-EW method covers the most target and noise web pages compared to the other RW methods with respect to both set1 and set2 (Figures 7.15c and 7.17c); however, there is no significant increase in the WBD solutions produced (Figures 7.14a and 7.16a). The RW-EW method has the effect of increasing coverage of the graph compared to the RW, which was achieved by Euclidean Weighted (EW) edges. This has the adverse effect of covering more noise web pages, and is due to increasing the Euclidean weighting of edges that are not part of the website, but exhibit some similarity; hence the increased weighting.

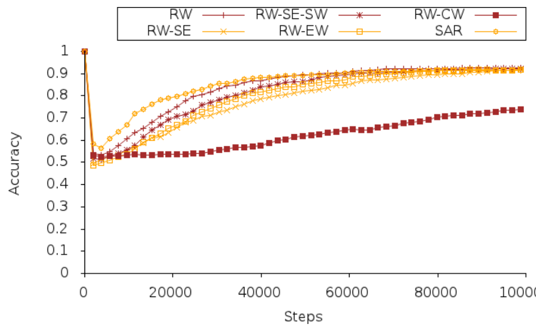
The SAR method performs much better on set1 than set2 in terms of the WBD performance accuracy (Figures 7.14a and 7.16a). The coverage of the SAR method produced a desirable result in terms of reducing noise coverage with respect to set1 as the amount of noise covered is less than other RW methods. In the less dense web graph of set2, the performance of the SAR method decreased in terms of WBD performance score (Table 7.12), and graph coverage (Figures 7.17b and 7.17c).

7.7.2.3 Random Ordering

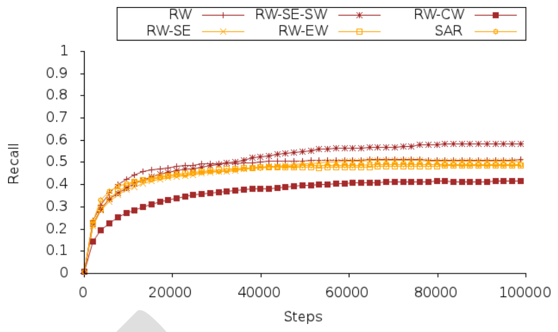
The evaluation presented in this sub-section reports on the WBD performance of the Random Ordering (RO) method of selecting pages from the graph. The RO method is not strictly a dynamic approach to the WBD problem, because the entire graph data is needed apriori (see section 7.7.2.3). The RO method selects pages uniformly at random from all pages in the graph independent of link structure. This traversal can be considered possible if all web pages in a graph are known at the initial point, and a complete set of edges is imposed between all pages. The evaluation of the RO method allowed for the comparative performance of a method that produced a randomised ordering of pages, but is independent of the link structure of the graph.

Tables 7.11 and 7.12 show the experimental results obtained for each of the dynamic

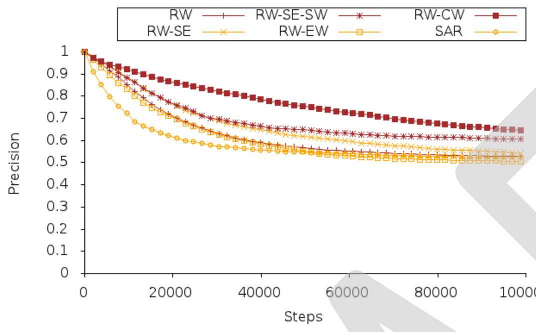
Figure 7.14: The average WBD performance of the dynamic approaches indicated on data set ADG set1.



(a) Accuracy

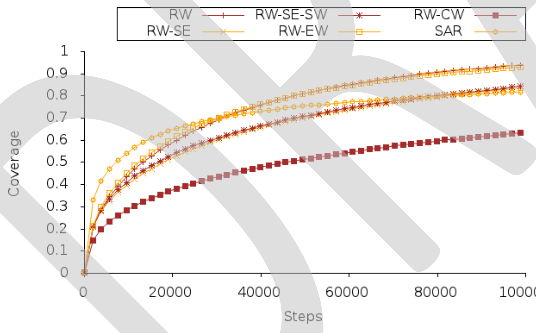


(b) Recall

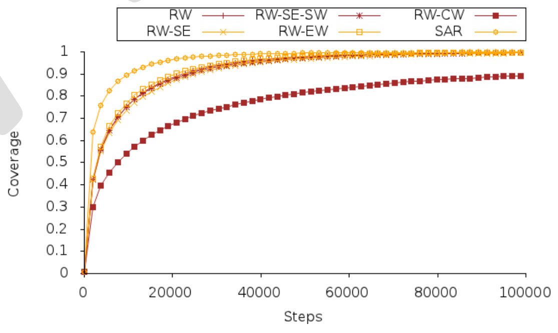


(c) Precision

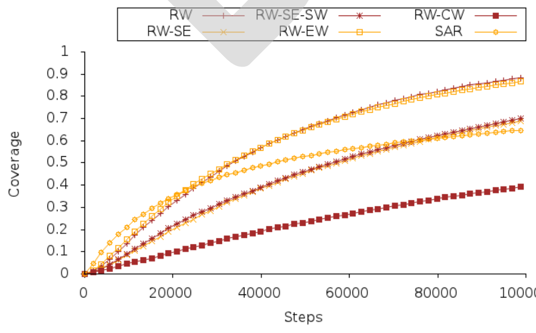
Figure 7.15: The graph coverage of the dynamic approaches indicated on data set ADG set1.



(a) Total Coverage

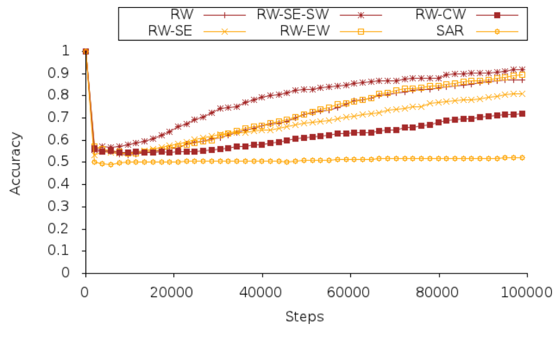


(b) Target Coverage

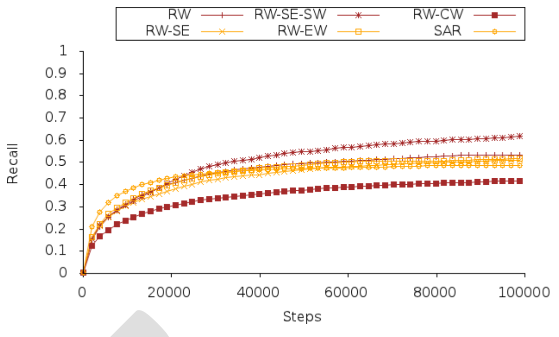


(c) Noise Coverage

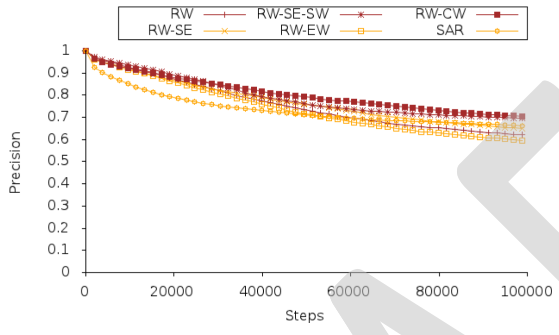
Figure 7.16: The average WBD performance of the dynamic approaches indicated on data set ADG set2.



(a) Accuracy

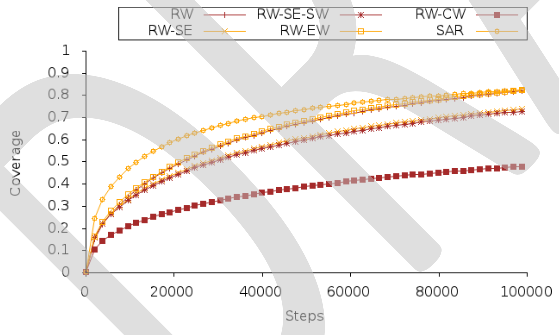


(b) Recall

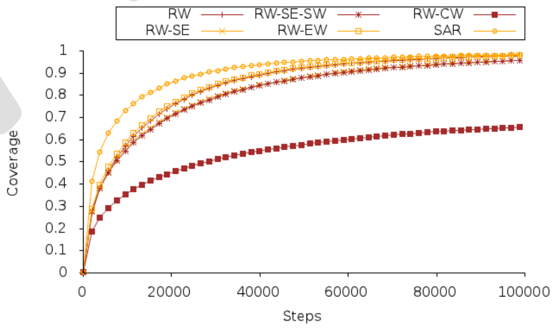


(c) Precision

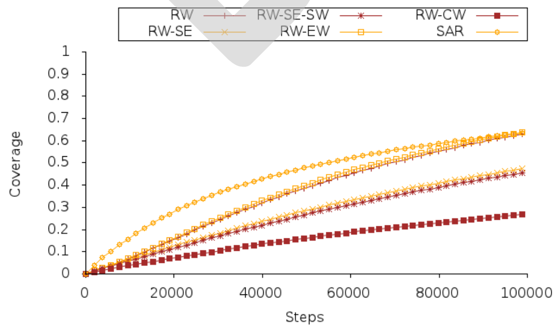
Figure 7.17: The graph coverage of the dynamic approaches indicated on data set ADG set2.



(a) Total Coverage



(b) Target Coverage



(c) Noise Coverage

approaches with respect to WBD performance for the data sets ADG set1 and set2 respectfully. The Tables show the graph coverage in terms of: (1) target pages, noise pages and total pages, (2) the WBD performance score, and (3) the average time taken per step for each of the approaches. It is shown that the RO method provides the highest performing WBD solution when compared to the other approaches considered if the number of pages in a graph is not an adversely large amount. This is shown in Table 7.11 were RO produces the highest performing WBD solution for ADG set1, while subsequently fails to do the same for the larger ADG set2 as shown in Table 7.12.

Figures 7.18 and 7.19 show the history in terms of WBD performance and graph coverage for the RO, BF, DF, RW and SAR approaches on ADG set1 and set2 respectfully. The RO method has improved accuracy recall and precision for data set1 within a small number of steps (Figures 7.18a, 7.18b and 7.18c). Where as RW and SAR need a larger number of steps to effectively cover web pages and subsequently randomise the selection of pages to produced an increasing accuracy (Figure 7.18a).

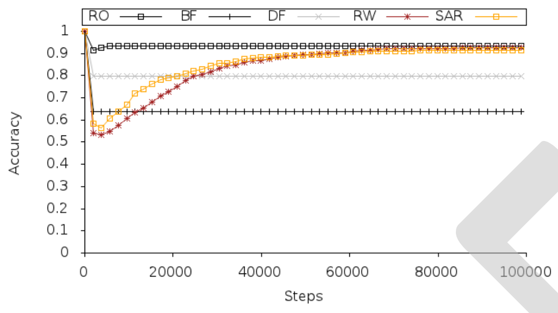
The evaluation of the RO method on set2 illustrated consistency with that of set1 in terms of the accuracy of the WBD solution produced, the accuracy was higher than that of the BF,DF, RW and SAR methods (Figure 7.19a). A different result was shown for recall and precision were the DF method is the best performing solution (Figures 7.19b and 7.19c). The lower recall and precision of the RO method on ADG set2 is due to the fact that the RO method cannot use the link structure of the graph to explore regions of connected and related pages, thus when applied to graphs of increasing size, in terms of both target and noise, it cannot make any adjustment to visit more target pages, and less noise pages. The ordering of target and noise pages produce by RO from a large graph makes it difficult to make the correct decisions to cluster target pages in a website cluster. The incremental clustering algorithm will not get to reconsider, and thus re-cluster the target pages, thus re-clustering pages will become increasingly infrequent as the graph increases in size. This means that initial clustering will have adverse effects when using the RO method with respect to graphs of increasing size.

The graph coverage of the RO method covers the target at a much faster rate than that of the RW and SAR methods which is consistent for both set1 and set2 (Tables 7.11 and 7.12). Due to the nature of the uniform selection of the RO method, revisits of nodes are possible, therefore the graph coverage cannot be as fast as the linear time coverage of the BF and DF methods. Another characteristic of the RO method is that it covers the noise page of the graph at a much faster rate than that of the RW and SAR methods, this has an obviously negative effect on the resource cost associated with the RO method.

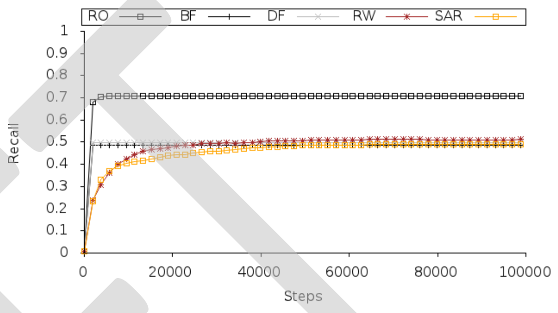
The run time of RO is also faster than that of the linear time BF and DF methods, but not as fast as the randomised methods of RW and SAR (Tables 7.11 and 7.12). The reason for the lower run time per step in comparison with BF and DF is that the

RO method has the advantage of not having the overhead costs of processing edges contained in the web graph by extracting hyperlinks. The RO method still has to extract features from the pages, which means that due to its random selection of noise and target pages, is not faster than RW and SAR methods. The RW and SAR methods have a higher probability of re-visiting a web page, re-visiting a page has a low cost when compared to visiting a new page; because features and edges are already extracted and cached.

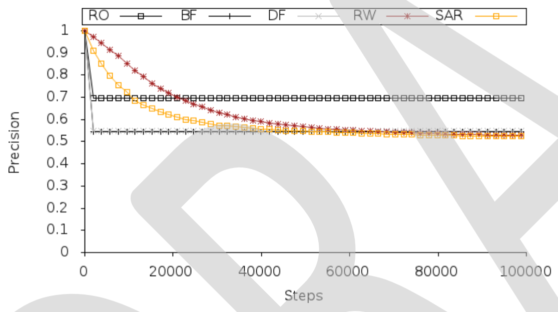
Figure 7.18: The WBD performance of the dynamic approaches indicated on data set ADG set1.



(a) Accuracy



(b) Recall



(c) Precision

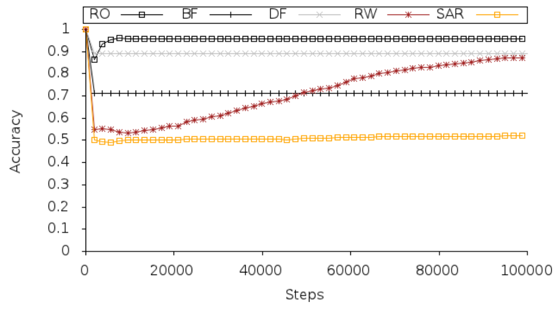
7.7.2.4 Summary

The evaluation that was presented in this section reported on results using ADGs that were synthetically generated based on the preferential attachment model. The evaluation presented in this section shows that:

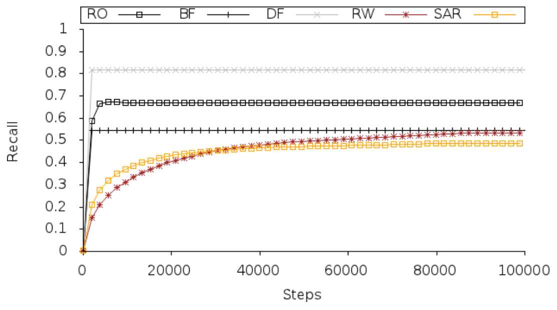
1. The IKM algorithm outperformed the ICA in the evaluation presented using the BF and RW graph traversal methods.

2. It is possible to control the graph traversal of a dynamic approach to WBD such that the graph coverage of target and noise pages can be influenced to increase the effectiveness of the WBD solution while decreasing the cost of processing unwanted noise.

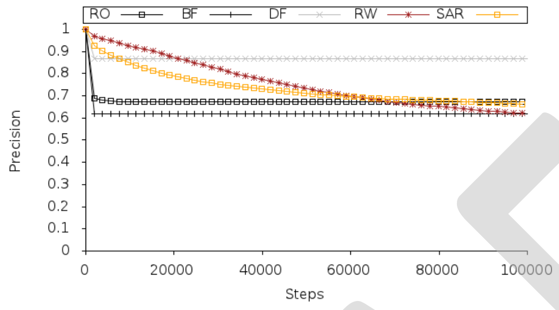
Figure 7.19: The WBD performance of the dynamic approaches indicated on data set ADG set2.



(a) Accuracy



(b) Recall



(c) Precision

3. The RO method produces the highest performing WBD solutions with respect to ADG data sets that are not proportionally large compared with the number of noise and target pages.

The evaluation reported in the following section presents the WBD performance of the MHRW, BF, DF, RW and SAR methods using the real data graphs. Utilising the real data sets the complexity of the graphs used for evaluation was increased compared to the evaluation presented in this section. In a bid to recreate the RO method that can be applied in a dynamic context, the MHRW method of graph traversal was evaluated. The following evaluation shows the MHRW traversal is most like the RO method, but traverses a local area which means it is not as adversely effected by noise. It is shown that the MHRW method produces the best WBD performance with respect to the dynamic approaches presented in this chapter using real data graphs.

7.7.3 Real Data Graphs

The evaluation presented in this section reports on results using the Real Data Graphs (RDG). The WBD performance was evaluated in terms of:

1. The comparison of five feature representations (Body text, Title text, Script links, Resource links, Image links) in terms of WBD performance in a dynamic context.

2. The WBD performance of graph traversal methods (BF, DF, RW, MHRW and

SAR) using the highest performing feature representation (Title text) in terms of WBD solution produced.

The evaluation presented in sub-section 7.7.3.1 provides a comparative analysis using the five feature representations that performed the best in the static context. The evaluation of the five features proved that the title text feature was the most robust, and performed the best in terms of WBD in the dynamic context. The evaluation presented in sub-section 7.7.3.2 provides a comparison of the graph traversal methods (BF, DF, RW, MHRW and SAR) representing web pages using the title feature which was the highest performing in terms of WBD in the dynamic context.

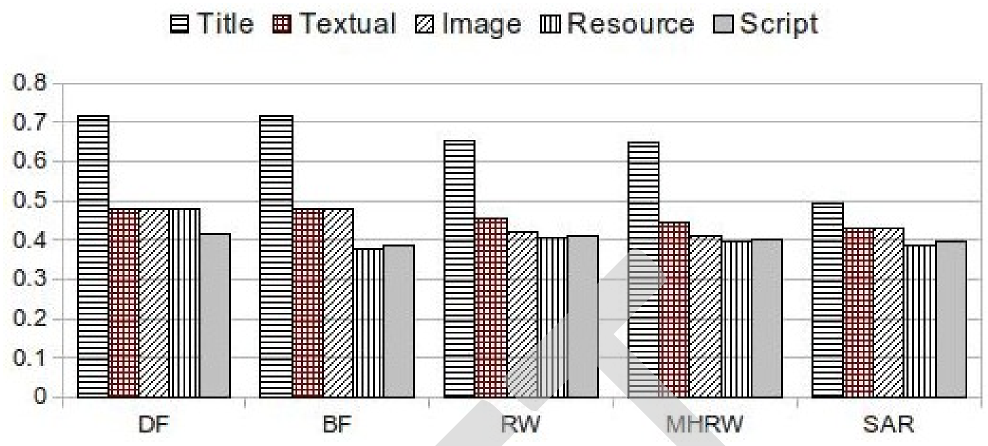
7.7.3.1 Feature Representation Comparison

Figure 7.20 presents the WBD performance using the five feature representations and the five graph traversal methods in the dynamic context. The evaluation of the five feature representations (Body text, Title text, Script links, Resource links, Image links) shows that the title feature exhibited a higher performance in terms of the WBD solutions produced. The other four features (textual, image, resource and script) evaluated using real data fall behind in terms of WBD score performance. The performance of the features in this chapter concentrating on dynamic techniques contrast to that of the features used in a static context (Chapter 5). In comparison, the evaluation presented in this chapter suggests that the title feature performs better when used in a dynamic context. Which contradicts the evaluation of the static chapter (section 5.5.1). It is suggested this is due to the common elements of the feature producing a good similarity between related pages when only portions of the data are available.

The title of a web page is written by the author/publisher to convey an accurate summary of the page that is quickly digestible by users. The title contains much less noise and is typically focused on a particular subject. The feature space created by using the title attribute of web pages is smaller in comparison to the other feature spaces. Typically few words are used in titles, when compared to the free text contained in the main body of a web page. The title typically rounds up the subject or context of a web page in a concise manner. In comparison, the main body text usually describes the subject/topic in greater detail, and may divert from the main point of the website. The body text feature space can be quite large, due to the diverse nature and unrestricted content that can be contained in a web page.

The image, script and resource links performed the worst in the dynamic context, it is suggested that this is due to the nature of the features with respect to the incremental clustering procedure used in a dynamic context. The image, script and resource feature representations do not provide a good enough similarity to distinguish pages contained in a website, when only partial data is available, which is the case in the dynamic context.

Figure 7.20: The WBD score performance for BF, DF, RW, MHRW and SAR dynamic approaches using Body text, Title text, Script links, Resource links, Image links feature representation.



7.7.3.2 Graph Traversal Method Comparison

The average performance of dynamic approaches BF, DF, RW, MHRW and SAR using the title feature and the ADG data sets is shown in Table 7.13. Figure 7.22 illustrates the WBD performance history of the approaches. In the initial stages of the approaches the average WBD performance score is higher for the BF and DF methods in comparison to RW, MHRW and SAR (Figure 7.22g). This behaviour is also consistent for the Fmeasure and recall values (Figure 7.22b and 7.22e). The higher values of recall indicated that the target pages were visited at a faster rate using the BF and DF methods when compared to RW, MHRW and SAR. The higher performance of the deterministic dynamic approaches BF and DF indicated that the identified web pages from the target website were subsequently being grouped together successfully into a representative website boundary using the title feature representation.

The values of precision and purity produced were around the same value for BF, DF, RW and MHRW (Figures 7.22f and 7.22d). This indicated that the number of the target web pages in the target cluster (K_T) was high compared to noise web pages.

In the long term, both BF and DF produced consistently similar WBD performances to the RW and MHRW methods (Table 7.13). The advantages of BF and DF methods are that they can traverse the graph structure in linear time, indicated by the number of end steps. Due to complexities of the RDGs, coupled with the task of clustering incrementally and the adverse effects of noise, the DF traversal does not show an improved WBD performance. The randomised traversal of RW, MHRW and SAR are subject to the underlying structure of the graph, in RDGs the structure can be complex, and unpredictable. The graph coverage of the RW dynamic approach remained high in comparison to MHRW and SAR (Figure 7.21). A closer analysis reveals that MHRW is much slower at covering the total graph than RW and SAR (Figure 7.21a). However, the MHRW traversal covered noise pages at a lower rate (Figure 7.21c), while it also maintained a high coverage of target pages (Figure 7.21b).

Table 7.13: The WBD performance the dynamic approaches as indicated, ordered by performance score. The graph coverage, WBD performance score, average time per step and total steps is shown for each approach using the average performance on the RDG (LivChem, LivHistory, LivMath and LivSace) data sets.

	Coverage				**Time(ms)/**	**Total**
	Target	**Noise**	**Total**	**Score**	**Steps**	**Steps**
BF	1.000	1.000	1.000	0.717	314.801	421
DF	1.000	1.000	1.000	0.714	320.561	421
RW	0.960	0.992	0.988	0.654	5.957	25000
MHRW	0.941	0.768	0.788	0.646	4.740	25000
SAR	0.890	0.869	0.871	0.495	5.611	25000

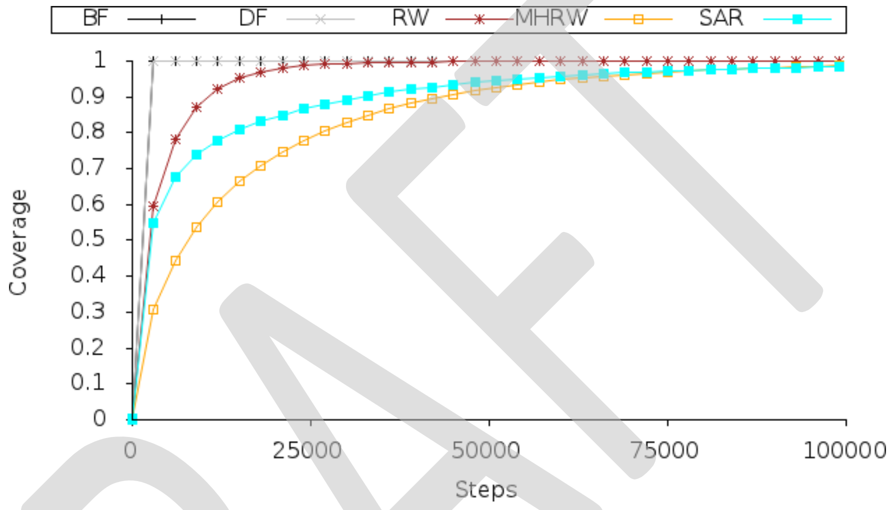
7.7.3.3 Summary

To summarise, the MHRW method proved to be the best performing graph traversal method when compared to the other methods tested in terms of the WBD solutions generated in a dynamic context using the RDGs. The MHRW method traverses the graph selecting pages in a random order depending on the hyperlink structure. In contrast to RW, the MHRW method avoids high degree pages with a certain probability. The MHRW method of selecting pages from the graph offered the advantages of:

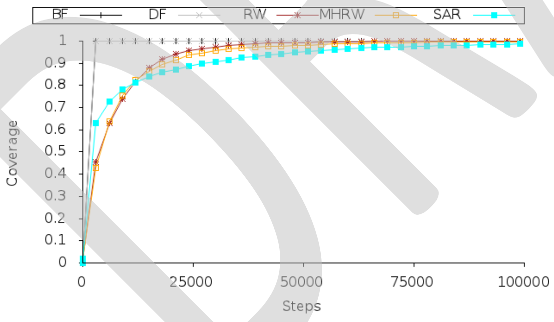
1. Not visiting a high number of noise web pages.
2. Producing a high WBD performance in comparison to the other methods evaluated.
3. Applicability in a dynamic context by using the hyperlink structure for its traversal of an unknown graph.
4. Randomising the ordering of pages so as to provide an effective order for the IKM algorithm.

The characteristics of the MHRW method translate into a solution that requires fewer resources to execute, as it minimises the downloading and parsing of unnecessary noise pages from the web, while producing a comparatively high quality WBD solution. The cost associated with downloading and processing unwanted noise pages from the web can rise if the WBD problem is scaled up to address larger domains.

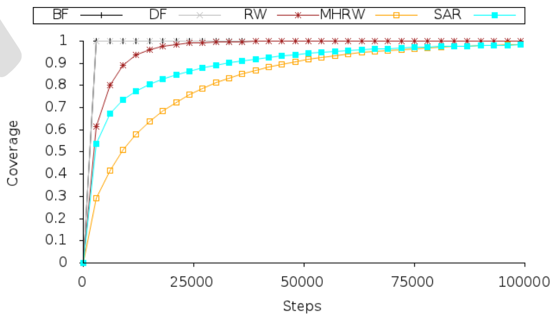
Figure 7.21: The average graph coverage for five graph traversals (as indicated) on ADG data sets (LivChem, LivHistory, LivMaths and LivSace) using the title feature.



(a) Total Coverage

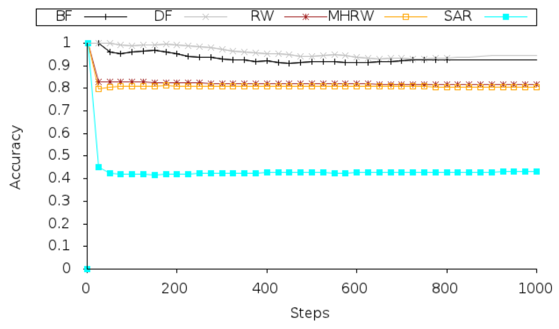


(b) Class Coverage

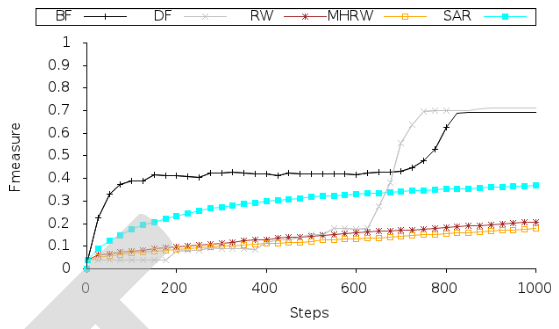


(c) Noise Coverage

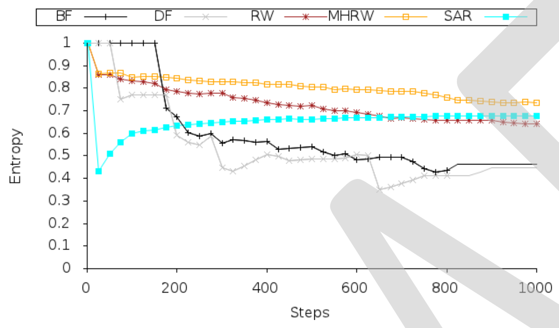
Figure 7.22: The WBD performance for five graph traversals (as indicated) on RDG data sets (LivChem, LivHistory, LivMaths and LivSace) using the title feature over 1000 steps.



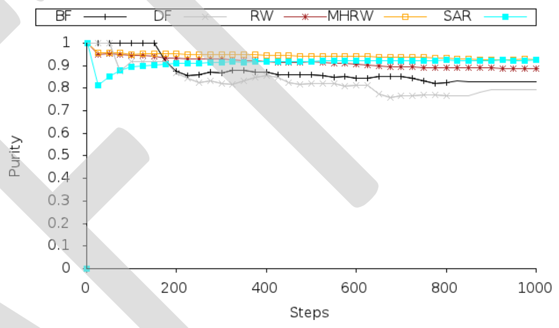
(a) Accuracy



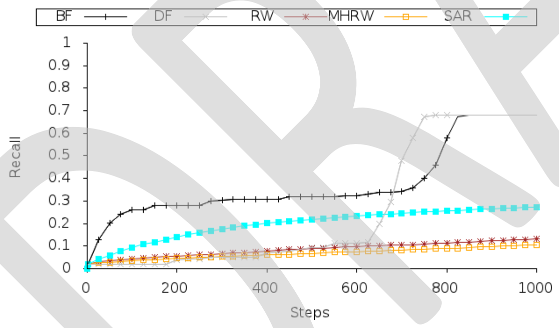
(b) Fmeasure



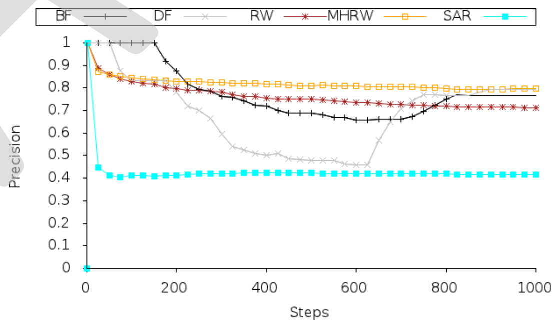
(c) Entropy



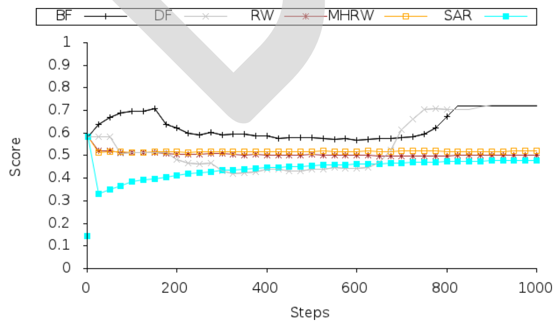
(d) Purity



(e) Recall



(f) Precision



(g) Score

7.8 Evaluation Summary

The evaluation strategy that was used in this chapter, directed at dynamic approaches to the WBD problem, considered an increasingly complex sequence of data sets. The test data was modelled in three particular ways using Binomial Random Graphs (BRGs), Artificial Data Graphs (ADGs) and Real Data Graphs (RDGs). The evaluation strategy was presented in three sections corresponding to the three kinds of data set used.

Binomial Random Graphs Evaluation The evaluated dynamic methods were RW and RW-SW used in combination with a buffer to store the ordering of pages produced by the graph traversals. The buffer is used to calculate a ratio value which can be used to detect cluster transitions in the BRG data sets.

The evaluation illustrated that: (1) the ordering of pages selected by RW exhibits frequent traversal of highly connected or dense sub-regions of pages in a graph, (2) the ordering of pages produced by the RW can be used to detect cluster transitions, by using a buffer and calculated ratio value, and (3) the graph representation method SW can be used to improve the ability of RW to remain inside a cluster when a cluster is adversely connected (for example when connections across clusters are proportionally high, or connections inside a cluster are proportionally low).

Artificial Data Graph Evaluation The dynamic approaches to the WBD problem, RW and BF, were initially compared in terms of WBD performance using both the ICA and IKM algorithms. The higher performing IKM algorithm was used to compare RW graph traversal using graph representations SE, SW, EW, CW, SE-SW with BF, DF, SAR and RO.

The evaluation demonstrated that: (1) the IKM algorithm outperformed the ICA using the BF and RW graph traversal methods, (2) the graph representation methods evaluated show that it is possible to control the graph traversal using a dynamic approach to WBD such that the graph coverage of target and noise pages can be influenced to increase quality of the WBD solution produced while decreasing the cost of processing unwanted noise, and (3) the RO method produces the highest quality WBD solution using ADGs that are not proportionally large compared with the number of noise and target pages.

Real Data Graph Evaluation The evaluated dynamic approaches to the WBD problem were BF, DF, RW, RO, MHRW and SAR in combination with the standard hyperlink graph representation using the IKM algorithm.

The characteristics of the MHRW method translate into a solution that cost fewer resources to execute, as it minimises the downloading and parsing of unnecessary noise pages from the web, while producing a comparatively high WBD solution. The MHRW method of selecting pages from the graph offered the following advantages: (1) randomising the ordering of pages which proves to provide an effective order for the IKM algorithm, (2) applicable in a dynamic context, using hyperlink structure for its traversal of an unknown graph, (3) avoid visiting high amount of noise web pages, and (4) producing high WBD performance in comparison to the other method evaluated in this section.

7.9 Conclusions

This chapter presented an investigation of the WBD problem in the dynamic context. In the dynamic context the web data is not fully available prior to the start of the analysis. The approaches presented in this chapter used various graph traversal techniques to gather portions of web data, which were then clustered incrementally in order to produce a WBD solution using only partial data.

In the dynamic approach the web page data is gathered by traversing the web graph using the hyperlink structure. The web pages are then pre-processed and feature representations created for each page. The pages are then incrementally clustered as the pages are traversed, a website boundary is then identified based on the clusters produced.

The evaluation of the dynamic approaches presented in this chapter was performed using the three categories of web graph data set: (1) Binomial Random Graphs (BRG), (2) Artificial Data Graphs (ADG), and (3) Real Data Graphs (RDG). The evaluation using each category of data set illustrated the advantages of the random walk based method of graph traversal with respect to WBD performance. In particular settings where the amount of data is large and not immediately available, and thus an adverse cost is associated with gathering data.

The Metropolis Hastings Random Walk (MHRW) method of graph traversal coupled with the title feature representation using the incremental kmeans algorithm (IKM) proved to be the best performing dynamic method in terms of the evaluation conducted using the Real Data Graph (RDG). The evaluation specifically concentrated on techniques that can be applied in a dynamic context, produce acceptable WBD performance while at the same time reducing the amount of noise pages visited. The website boundary representation was “as good” as that produced by other methods; however, MHRW visited fewer noise pages, hence efficiency gains were realised. Thus the MHRW method was found to provide the most effective WBD problem solution, providing the best comprise between WBD performance, while maintaining to visit a lower amount of noise pages of the graph, which proves to cost much less in terms of resources.

Chapter 8

Conclusion

This chapter presents a summary of the proposed approaches to the Website Boundary Detection (WBD) problem, a comparison of the approaches, the contribution and main findings of the research, and possible directions for future work. The summary of the research is presented in section 8.1. A comparison of the proposed approaches is presented in section 8.2. The contributions and main findings are given in section 8.3 and the suggested future research directions in section 8.4.

8.1 Summary

This thesis has described research undertaken in the field of web data mining, which is a sub field of Knowledge Discovery in Databases (KDD). The specific area this thesis contributes to is the area of website mining. The specific problem that the research described was directed at was an investigation of solutions to the WBD problem. The WBD problem is described as the task of identifying the collection of all web pages that are part of a single website. Potential solutions to the WBD problem can be beneficial with respect to the archiving of web content and the automated construction of web directories, amongst many others (see section 1.1).

The findings in this work can be validated by the application of the techniques to WBD problems other that what have been presented in this thesis. In particular opinion mining is a problem that can benefit from the techniques presented in this work. The website boundary in question with respect to the problem of opinions mining is one that is focused on a certain topic. The topic containing opinions on a product or a brand. This boundary can span multiple physical domains and can have a large magnitude given the prolific use of social media. The boundary detection problem in such a case benefits from techniques that can discover website boundaries in a dynamic setting were the magnitude of the problem can cause traditional techniques to become non-scalable.

A pre-requisite to any practical WBD approach is that of a definition of a website. This thesis has presented a discussion of previous definitions of websites, which concluded that the current definitions provide ambiguity with respect to any practical approach to the WBD problem. A proposed definition of a website, which was used with respect to the approaches presented in this thesis, was thus presented.

The approaches to the WBD problem investigated in this thesis were directed at both the static and dynamic contexts. In the static context the web data to be considered is required to be available prior to the start of any WBD solution process. In the dynamic context the web data is collected as the WBD solution generation process proceeds. Three approaches to the solution of the WBD problem were presented in this thesis, two static approaches and one dynamic approach:

1. Feature analysis based static WBD
2. Graph structure partitioning based static WBD
3. Incremental clustering using graph traversal based dynamic WBD

The latter built upon the initial research findings generated through the evaluation of the two static approaches.

The first static approach presented in this thesis concentrated on the types of features that could be used to represent web pages. This approach presented a practical solution to the WBD problem by applying clustering algorithms to various combinations of features. Further analysis investigated the best combination of features to be used in terms of WBD performance.

The second static approach investigated graph partitioning techniques based on the structural properties of the web graph in order to produce WBD solutions. Two approaches were considered, a hierarchical graph partitioning technique, and a method based on the minimum cuts of flow networks.

The final proposed approach to the WBD problem presented in this research considered the dynamic context. The dynamic approach was founded on the findings from the application of the two static approaches. The dynamic approach subsequently produced a solution that incrementally built a website boundary by traversing the web graph structure, while clustering web pages using various feature representations.

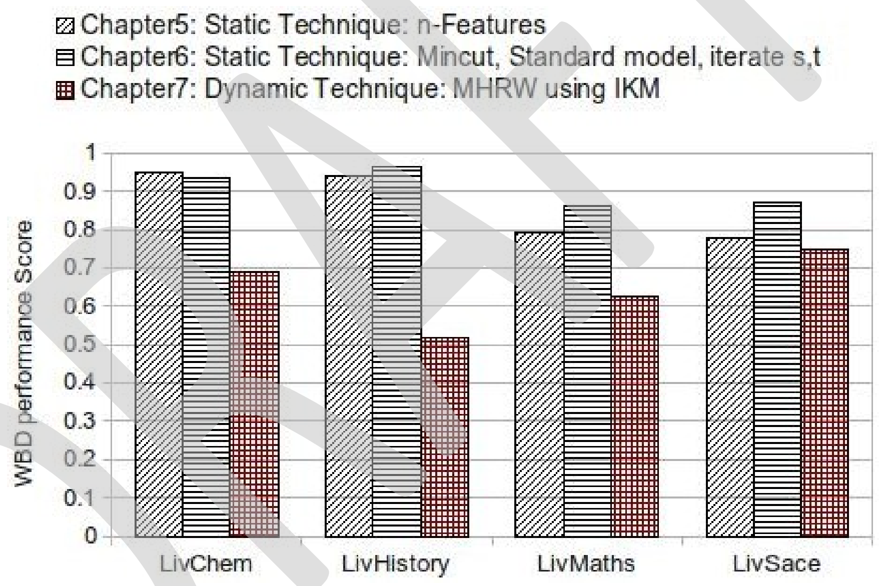
In an initial attempt to evaluate the approaches presented in the dynamic context, synthetically generated Artificial Data Graphs (ADGs) were created to provide a controlled environment for analysis. A final evaluation of both the static and dynamic approaches presented in this thesis was conducted using Real Data Graphs (RDGs) gathered from four academic departments hosted by the University of Liverpool (LivChem, LivHistory, LivMaths and LivSace).

8.2 Comparison of proposed approaches

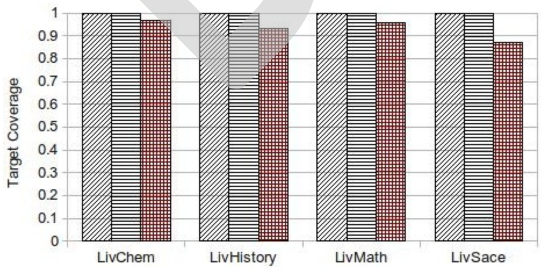
This section reports on an overall comparison of the approaches to provide solutions to the WBD problem. The comparison was undertaken in terms of WBD performance, and graph coverage. The WBD performance score was used to measure how representative the website boundary solution was, with respect to each approach. The graph coverage measured the amount of data gathered to produce the WBD solution, which effectively illustrated the potential resource cost associated with requesting, downloading and pre-processing data from the web.

The evaluation of the afore mentioned approaches considered the effects of using various parameters or variations in conditions. For the overall comparison reported in this section the best performing variations of each of the approaches were compared. Each of the approaches was compared using the Real Data Graphs (RDGs) data sets (LivChem, LivHistory, LivMaths and LivSace) which provided a comparison using real data gathered from the WWW.

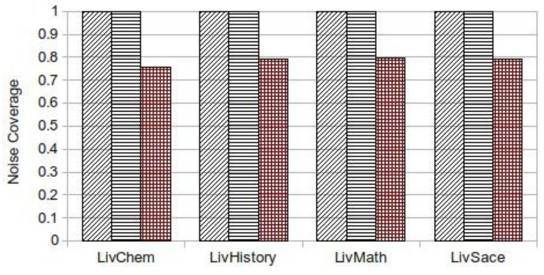
Figure 8.1: Comparison of the highest performing approaches for the production of WBD solutions presented in this thesis. Sub-Figure (a) shows WBD performance score, (b) Target and (c) Noise coverage.



(a) Score



(b) Target Coverage



(c) Noise Coverage

Figure 8.1 presents a comparison of the highest performing approaches to WBD from chapters 5, 6 and 7. The Figure indicates that the highest performing WBD solutions were produced from the static n-features approach and the Mincut approach. The static approaches perform much better than the dynamic approach. This outcome would be expected as the static approaches required all the data aprior, subsequently the approaches can use all the data to make relative decisions with respect to the WBD problem. In the dynamic context it is not the case that all data is known, in advance only partial data is used to produce a WBD solution. Therefore the decisions made using the dynamic approach are made relative to what is known about the partial data.

The comparative evaluation of the dynamic approaches proposed in this work provide an adequate WBD solution performance in most cases (Figure 8.1a), while at the same time maximising the number of target pages gathered (Figure 8.1b) and reducing the number of unwanted noise pages (Figure 8.1c). If a standard cost is to be associated with each web page, which is consistent with the requesting, downloading and pre-processing, then it can be seen from the comparative evaluation that the dynamic approach provides a much lower cost WBD solution, while still maintaining an adequate WBD performance.

8.3 Contributions and main findings

In the introduction to this thesis (chapter 1) a number of research issues were identified. Each of these issues are listed below together with how the work described in this thesis addressed each issue.

1. **Definition of a website**

   The work in chapter 3 provides a discussion with respect to the term "website" and the nature in which is it widely used, but poorly defined. The chapter described how the term means many things to many different observers, which ultimately leads to the conclusion that existing website definitions are ambiguous in their nature, and thus not suitable for a practical approach to provide a solution to the WBD problem. Chapter 3 describes the derived definition of a website proposed in this thesis, which was subsequently used effectively for the practical implementation of approaches to provide solutions to the WBD problem.

2. **Web page model**

   The work described in chapters 5, 6 and 7 addresses the issue of how to model web pages with respect to the WBD problem. A web page comprises many possible features that could potentially be used to model the web page, and by extension websites, with respect to the WBD problem. The issue of how to model web pages was address as follows:

   (a) **Web page model based on page content**

The work described in chapter 5 presented an investigation of the best feature representations to model web pages based on web page content with respect to the WBD problem in a static context. This work presented the best features, and combinations of these features, to effectively model web pages to produce WBD solutions. The most signicant features to represent web pages in a static WBD context are shown to be script, image and resources link feature representations.

(b) **Web page model based on hyperlink structural properties**

The work described in chapter 6 presented an investigation founded on the structural properties of web pages induced by the hyperlinks of the web graph. This work presented two specific static approaches that effectively modelled the web pages using the hyperlink structure to produce WBD solutions.

(c) **Web page model based on page content and hyperlink structural properties**

The work described in chapter 7 presented an investigation which modelled web pages and exploited the hyperlink and context features in order to provide effective WBD solutions in a dynamic context.

3. **Magnitude of the web**

   The work described in chapter 7 provided an approach in a dynamic context, which provided solutions to the WBD problem with respect to the increasing magnitude of the web and the associated computational overheads.

   The work described incremental clustering approaches using graph traversal techniques that were shown to provide effective WBD solutions while: (1) reducing the number, and associated cost, of requesting, downloading and preprocessing pages from the web, (2) providing a reduction in knowledge encompassing the website to be identified requiring only a seed page as a starting point.

The overall objective of the research described in this thesis was to provide a solution to the research question *is it possible to discover website boundaries using unsupervised learning techniques?* The research presented in this thesis described approaches to the WBD problem providing approaches based on unsupervised learning techniques with respect to a static and a dynamic context, which clearly indicate effectiveness by the WBD solution performance produced by each approach.
The main contributions of this research may be summarised as follows:

1. A website definition was proposed and used effectively in the implementation of WBD approaches.

2. The use of content features to represent web pages in terms of WBD solutions was proven to be effective.

3. A static approach to provide WBD solutions concentrating on content of web pages to represent features was proven to be effective.

4. Two static approaches to provide WBD solutions concentrating on the hyperlink structure of the web graph was proven to be effective.

5. WBD in a dynamic context, using both structural properties and content features to represent web pages was proven to be effective.

In conclusion this thesis presented static and dynamic approaches to practically produce solutions to the WBD problem.

8.4 Future Work

The research described in this thesis has sparked a number of promising directions for future research:

- **Dynamic Web Graphs.** In the research presented in this thesis, the evaluation was conducted on web graphs that were assumed to be fixed, and do not change over time. The web graph is gathered, either at once in the case of the static approach, or in incremental portions in the case of the dynamic approach. However, the web is an example of an evolving graph, were by the structure changes over time [62]. This means that nodes and edges may be added/removed or updated at any given time. One avenue for further work would be to extend the approaches in this research to web graphs that exhibit changes over time. Thus:
  - The dynamic approaches could be extended to a situation where a local database needs to keep a "fresh" copy of the ever evolving web graph with respect to the website. This open problem of web persistence [121] could be approached using the proposed dynamic graph traversal methods so as to keep a local database upto date.
  - The static and dynamic context of producing website boundaries could be further evaluated on evolving web graphs so as to investigate the advantages of each approach in this setting.

- **Recommendation Systems.** The research presented in this thesis is strictly directed at providing solutions to the WBD problem. In the dynamic context the approach taken is to traverse the web graph, and produce a clustering incrementally. There is previous work, using random walk based methods, with respect to recommendation systems where such systems produce lists of related academic papers from a co-citation network [93]. One avenue to extend the dynamic work presented in this thesis would be to apply the dynamic approach to identify related items in a graph. The dynamic work could be extended from identifying related web pages in the web graph, to identifying related items in a graph representation of item similarity.

- **Multiple Crawlers.** In the dynamic approach to the WBD problem a single graph traversal method was used to incrementally build website boundaries. There has been previous work that uses multiple random walk crawlers in the context of reducing the "cover time" [56]. Another avenue for future work is to explore web graphs using multiple random walks acting in sequence in order to incrementally cluster web pages with respect to the WBD problem.

- **Experiment with alternative data sets.** The evaluation described in this thesis concentrated on synthetically generated web graph data, and four data sets collected from the University of Liverpool. Evaluating the approaches presented in this thesis on further data sets would provide further insight into how generic the WBD solutions are at producing representative website boundaries.

Bibliography

[1] *The American Heritage Dictionary of the English Language.* Houghton Mifflin Company, 4th edition, 2006.

[2] *Collins English Dictionary Complete And Unabridged 10th Edition.* HarperCollins Publishers, 10th edition, 2009.

[3] Wikipedia: Article 'website'. `http://en.wikipedia.org/w/index.php?title=Website\&oldid=293893527`, June 2009. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63OOWHqfU`.

[4] wordnetweb.princeton.edu (wordnet): Princeton university wordnet 3.0 definition: 'web site'. `http://wordnetweb.princeton.edu/perl/webwn?s=website`, June 2009. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63OOWHqfU`.

[5] dictionary.reference.com (dictionary.com): Unabridged dictionary (based on the random house dictionary, random house, inc. 2011.): 'website'. `http://dictionary.reference.com/browse/website`, Nov 2011. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63OQGqFF6`.

[6] webarchive.jira.com (jira studio): Glossary of web archiving terms. `https://webarchive.jira.com/wiki/display/ARIH/Glossary%20of%20Web%20Archiving%20Terms`, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/64e1Np5qt`.

[7] Wikipedia: Article 'entity'. `http://en.wikipedia.org/wiki/Entity`, August 2012. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/6AJtzEerK`.

[8] www.about-the-web.com: Glossary of internet terms. `http://www.about-the-web.com/shtml/glossary.shtml`, Jan 2012. Accessed: 2012-08-30. Archived by WebCite: `http://www.webcitation.org/64exAh9pG`.

[9] www.cardinalbusinessgroup.com: Seo glossary of terms. `http://www.cardinalbusinessgroup.com/resources/seo-glossary/`, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/64e0m74b4`.

[10] www.cpr.ca.gov (california performance review): Glossary of terms. http://cpr.ca.gov/CPR_Report/Issues_and_Recommendations/Appendix/Glossary_of_Terms.html, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e17ORrN.

[11] www.d.umn.edu (university of minnesota): Web design glossary. http://www.d.umn.edu/itss/support/Training/Online/webdesign/glossary/w.html, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e1BscQJ.

[12] www.find-info.ru (redhat): Glossary. http://www.find-info.ru/doc/unix/009/glossary.html, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e0g4rH8.

[13] www.itconnections.unc.edu (it connections): Glossary. http://itconnections.unc.edu/glossary.html, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64ewUqsyN.

[14] www.lib.tudelft.nl (delft university): Glossary. http://www.lib.tudelft.nl/tulib/glossary/index.htm, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e1Wrcvw.

[15] www.naa.gov.au (national archives of australia): Glossary. http://www.naa.gov.au/records-management/publications/glossary.aspx, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e1J6sqw.

[16] www.socialbysocial.com: A to z of key terms, jargon buster. http://www.socialbysocial.com/book/a-to-z, Jan 2012. Accessed:2012-08-29. Archived by WebCite: http://www.webcitation.org/64e0JFBeP.

[17] www.soundmarketingconcepts.com: Services. http://www.soundmarketingconcepts.com/services.htm#, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e0b07th.

[18] www.upedu.org (unified process for education): Glossary. http://www.upedu.org/process/glossary.htm, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64ex1DLG1.

[19] www.websiteslug.com: Glossary of terms for making and publishing a website. http://www.websiteslug.com/pages/glossary-of-terms.htm, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: http://www.webcitation.org/64e0saIDp.

[20] www.wolfwebdesign.co.uk: Jargon buster. `http://www.wolfwebdesign.co.uk/features/jargon-buster/`, Jan 2012. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/64e0zAj2Z`.

[21] S. Abiteboul, G. Cobena, Julien Masanès, and G. Sedrati. A First Experience in Archiving the French Web. In *Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, volume 2458 of *Lecture Notes in Computer Science*, pages 1–15, London, UK, 2002. Springer-Verlag.

[22] S Abou-Zahra. W3c mailing lists: proposed definition for “website”. `http://lists.w3.org/Archives/Public/public-wai-evaltf/2012Feb/0021.html`, Feburary 2012. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/6AGnxkoBh`.

[23] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[24] D. Aldous and J. Fill. Reversible Markov chains and random walks on graphs. *Monograph in preparation*, 2002.

[25] R. Aleliunas, R.M. Karp, R.J. Lipton, L. Lovasz, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. *20th Annual Symp. on Foundations of Computer Science*, pages 218–223, 1979.

[26] J. Alpert and H. Hajaj. We knew the web was big. `http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html`, July 2008. Accessed: 2012-08-28. Archived by WebCite: `http://www.webcitation.org/6AFzp5W8z`.

[27] E. Amitay, D. Carmel, A. Darlow, R. Lempel, and A. Soffer. The Connectivity Sonar: Detecting Site Functionality by Structural Patterns. In *Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, pages 38–47, New York City, USA, 2003. ACM.

[28] Y Asano, H. Imai, M. Toyoda, and M. Kitsuregawa. Applying the Site Information to the Information Retrieval from the Web. In *Proceedings of the Third International Conference on Web Information Systems Engineering, 2002. WISE 2002.*, WISE 2002, pages 83–92. IEEE Computer Society, 2002.

[29] A. P. Asirvatham and K. K. Ravi. Web Page Classification based on Document Structure. Technical report, International Institute of Information Technology, Hyderabad, India., 2001.

[30] R Baeza-Yates and B Ribeiro-Neto. *Modern information retrieval.* Addison-Wesley Longman, 1999.

[31] W. Barbakh and C. Fyfe. Online clustering algorithms. *International journal of neural systems*, 18(3):185–94, June 2008.

[32] K. Barber, editor. *The Canadian Oxford Dictionary.* Oxford University Press, 2nd edition, 2004.

[33] J. A. Bargh and K. Y. A. McKenna. The Internet and social life. *Annual Review of Psychology*, 55:573–590, 2004.

[34] E. Baykan, M. Henzinger, L. Marian, and I. Weber. A Comprehensive Study of Features and Algorithms for URL-Based Topic Classification. *ACM Transactions on the Web*, 5(3):1–29, July 2011.

[35] L Becchetti, C Castillo, D Donato, S Leonardi, and R Baeza-Yates. Link-based characterization and detection of web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web*, pages 1–8, 2006.

[36] A. Benczúr, K. Csalogány, and T. Sarlós. Link-based similarity search to fight Web spam. In *Adversarial Information Retrieval on the Web*, pages 1–8, Seattle, Washington, USA, 2006.

[37] T. Berners-Lee. Information management: A proposal. `http://www.w3.org/History/1989/proposal.html`, March 1989. Accessed: 2012-08-28. Archived by WebCite: `http://www.webcitation.org/6AFy0VV3Q`.

[38] T Berners-Lee. Information Management: A Proposal. *CERN. Geneva, Switzerland*, March 1989.

[39] T. Berners-Lee, J Hendler, W. Hall, N. Shadbolt, and D. J. Weitzner. Creating a science of the Web. *Science*, 313(5788):769–71, 2006.

[40] T Berners-Lee, Daniel J. Weitzner, Wendy Hall, Kieron O'Hara, J Hendler, and Nigel Shadbolt. A Framework for Web Science. *Foundations and Trends in Web Science*, 1(1):1–130, January 2006.

[41] K. Bharat, B-W. Chang, M. Henzinger, and M. Ruhl. Who links to whom: mining linkage between Web sites. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 51–58, Washington, DC, USA, 2001. IEEE Computer Society.

[42] R Brachman and T Anand. The Process of Knowledge Discovery in Databases: A First Sketch. In *Proceedings Workshop on Knowledge Discovery in Databases*, pages 1–12, Washington, Seattle, 1994.

[43] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998.

[44] A. Z Broder. Graph structure in the Web. *Computer Networks*, 33(1-6):309–320, June 2000.

[45] A. Z Broder, M Najork, and J. L Wiener. Efficient URL caching for world wide web crawling. In *WWW'03 Proceedings of the 12th international conference on World Wide Web*, pages 679–689, Budapest, Hungary., 2003. ACM.

[46] A. Brown. *Archiving Websites: a practical guide for information management professionals*. Facet Publishing, London, England, 2006.

[47] S Carberry. *Plan recognition in natural language dialogue*. MIT press, 1990.

[48] V. Carchiolo, A. Longheu, and M. Malgeri. Information categorization in web pages and sites. *Web Intelligence and Agent Systems*, 3(3):183–198, July 2005.

[49] S Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, 2003.

[50] G. Chang, M. J. Healey, J. A. M. McHugh, and J. T. L. Wang. *Mining the World Wide Web: An Information Search Approach*. Kluwer Academic Publishers, London, second edition, 2001.

[51] P Chapman, J Clinton, R Kerber, T Reinartz, C Shearer, and R Wirth. CRISP-DM 1.0 Step-by-step data mining guide, 2000.

[52] K. Chellapilla and D. M Chickering. Improving Cloaking Detection using Search Query Popularity and Monetizability. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web*, pages 17–24, Seattle, WA, August 2006.

[53] M S Chen, J Han, and P Yu. Data mining: an overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, 1996.

[54] K-W. Cheung and Y. Sun. Mining Web Site's Clusters from Link Topology and Site Hierarchy. In *Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence*, page 271, Washington, DC, USA, October 2003. IEEE Computer Society.

[55] M Clark and O Thyen. *The Concise Oxford-Duden German Dictionary: German-English, English-German*. Oxford University Press, USA, 1998.

[56] C Cooper. Multiple random walks in random regular graphs. *SIAM Journal on Discrete Mathematics*, 23(4):1738–1761, 2009.

[57] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, third edition, September 2009.

[58] M.G. da Costa. Web structure mining: an introduction. In *IEEE International Conference on Information Acquisition*, pages 590–595. IEEE, 2005.

[59] J Daintith and E Wright, editors. *A Dictionary of Computing*. Oxford University Press, 6th edition, 2008.

[60] M. Deegan and S. Tanner. *Digital Preservation*. Digital futures series, 2006.

[61] H. Denis. Free online dictionary of computing (foldoc): 'website'. `http://foldoc.org/website`, July 2005. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63OR89nT6`.

[62] O. Denysyuk and L. Rodrigues. Random Walk on Directed Dynamic Graphs. *Arxiv preprint arXiv:1101.5944*, 2011.

[63] P. Dmitriev. As we may perceive: finding the boundaries of compound documents on the web. In *Proceeding of the 17th international conference on World Wide Web*, pages 1029–1030, Beijing, China, 2008. ACM.

[64] P. Dmitriev and C. Lagoze. Automatically Constructing Descriptive Site Maps. In *Frontiers of WWW Research and Development, APWeb 2006*, pages 201 – 212. Springer Berlin, Heidelberg, 2006.

[65] P. Dmitriev, C. Lagoze, and B. Suchkov. Finding the boundaries of information resources on the web. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, page 1124, New York, USA, 2005. ACM Press.

[66] Pavel Dmitriev, Carl Lagoze, and Boris Suchkov. As we may perceive: inferring logical documents from hypertext. In *Proceedings of the sixteenth ACM conference on Hypertext and Hypermedia*, pages 66–74. ACM, 2005.

[67] B. Dong and H. Liu. Enterprise Website Topic Modeling and Web Resource Search. In *Sixth International Conference on Intelligent Systems Design and Applications*, volume 3, pages 56–61. IEEE, October 2006.

[68] B. Dong, H. Liu, Z. Hou, and X. Liu. Topic-based website feature analysis for enterprise search from the web. In *Web Information Systems*, volume 4255, pages 84–89, Berlin, Heidelberg, October 2006. Springer Berlin, Heidelberg.

[69] B. Dong, G. Qi, and X. Gu. Domain-specific website recognition using hybrid vector space model. In *Proceedings of the 6th international conference on Advances in Web-Age Information Management*, volume 3739, pages 840–845, Berlin, Heidelberg, October 2005. Springer Berlin Heidelberg.

[70] C. Doyle, editor. *A Dictionary of Marketing.* Oxford University Press, 3rd edition, 2011.

[71] M. H. Dunham. *Data Mining: Introductory and Advanced Topics.* Prentice Hall PTR Upper Saddle River, NJ, USA, 2002.

[72] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, 19(2):248—-264, 1972.

[73] N. Eiron and K. S. McCurley. Untangling compound documents on the web. In *Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, pages 85–94, New York, USA, 2003. ACM Press.

[74] P. Erds and A. Rényi. On the evolution of random graphs. *Evolution*, 5(1):17–61, 1960.

[75] M. Ester, H-P. Kriegel, and M. Schubert. Web site mining: a new way to spot competitors, customers and suppliers in the world wide web. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 249—-258, New York City, USA, 2002. ACM.

[76] M Ester and HP Kriegel. A density-based algorithm for discovering clusters in large spatial databases with noise. In *2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[77] U. Fayyad and G. Piatetsky-Shapiro. From data mining to knowledge discovery in databases. *AI magazine*, pages 37–54, 1996.

[78] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *Proc. 2nd International Conference on Knowledge Discovery and Data Mining*, pages 82–88, 1996.

[79] U. Fayyad, G Piatetsky-Shapiro, and P. Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, November 1996.

[80] C. Fellbaum, R. Poli, M. Healy, and A. Kameas. *Theory and Applications of Ontology: Computer Applications.* Springer, Netherlands, 2010.

[81] W. Feller. Introduction to probability theory and its applications. *WSS*, vol. 1, 1968.

[82] D. Fensel, F. M. Facca, E. Simperl, and I. Toma. Semantic Web Services. In *Semantic Web Services*, pages 87–104. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[83] L.R. Ford and D.R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399—-404, 1956.

[84] L.R. Ford and D.R. Fulkerson. *Flows in Networks.* Princeton University Press, Princeton, NJ., 1962.

[85] H.W. Fowler and F.G. Fowler. *Concise Oxford English Dictionary.* Oxford University Press, Oxford, 11th edition, 2004.

[86] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129—-1164, November 1991.

[87] A. Gibbons. *Algorithmic Graph Theory.* Cambridge University Press, 1985.

[88] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring Web communities from link topology. In *Proceedings of the ninth ACM conference on Hypertext and hypermedia*, pages 225–234, New York, USA, 1998. ACM Press.

[89] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. In *Proceedings of the National Academy of Sciences*, pages 7821—-7826, 2002.

[90] K. Golub and A. Ardö. Importance of HTML structural elements and metadata in automated subject classification. In *In Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries*, pages 368–378. LNCS, 2005.

[91] D. Gomes and M.J. Silva. Modelling Information Persistence on the Web. In *6th International Conference on Web Engineering*, pages 193 – 200. ACM Press, 2006.

[92] M. T. Goodrich and R. Tamassia. *Algorithm Design: Foundations, Analysis, and Internet Examples.* Wiley, 2001.

[93] M. Gori and A. Pucci. Research Paper Recommender Systems: A Random-Walk Based Approach. In *International Conference on Web Intelligence*, pages 778–781. IEEE, December 2006.

[94] S. Halford, C. Pope, and L. Carr. A Manifesto for Web Science. In *Extending the Frontiers of Society On-Line.* ACM, 2010.

[95] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[96] D. Harel and Y. Koren. A Fast Multi-scale Method for Drawing Large Graphs. In *International Symposium on Graph Drawing*, pages 183–196. Springer-Verlag, September 2000.

[97] B. He, M. Patel, Z. Zhang, and K. C-C. Chang. Accessing the deep web. *Communications of the ACM*, 50(5):94–101, May 2007.

[98] M. Henzinger. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 284–291. ACM, 2006.

[99] M.R. Henzinger. Tutorial Web Information Retrieval. Technical report, Tutorial from 16th International Conference on Data Engineering, 2000.

[100] D. Ince. *Dictionary of the Internet.* Oxford University Press, 2nd edition, 2003.

[101] M Jeong. Conceptualizing Web site quality and its consequences in the lodging industry. *International Journal of Hospitality Management*, 22(2):161–175, June 2003.

[102] G. H. John. *Enhancements to the data mining process.* PhD thesis, PhD thesis, Stanford, CA, USA, 1997.

[103] D. R. Karger. Global min-cuts in RNC, and other ramifications of a simple min-out algorithm. In *Proceedings of the fourth annual ACM-SIAM*. ACM, 1993.

[104] M. Keller, A. Blanchard, and M. Nussbaumer. Decomposing the Web Graph: Towards Information Architecture Mining, 2011.

[105] M. Keller and M. Nussbaumer. MenuMiner: Revealing the Information Architecture of Large Web Sites by Analyzing Maximal Cliques. In *Proceedings of the 21st international conference companion on World Wide Web*, page 1025, New York, USA, April 2012. ACM Press.

[106] J.M. Kleinberg. Hubs, authorities, and communities. *ACM Computing Surveys*, 31(4es):5–es, December 1999.

[107] W. Koehler. An analysis of web page and web site constancy and permanence. *Journal of the American Society for Information Science*, 50(2):162–180, 1999.

[108] W. Koehler. Web page change and persistence A four-year longitudinal study. *Journal of the American Society for Information*, pages 162–171, 2002.

[109] M. Kovacevic and M. Diligenti. Visual adjacency multigraphs-a novel approach to web page classification. In *SAWM04 workshop*, 2004.

[110] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1):1–58, March 2009.

[111] HP Kriegel and Matthias Schubert. Classification of websites as sets of feature vectors. In *International Conference databases and applications*, pages 127–132, 2004.

[112] R. Kumar. Trawling the Web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, May 1999.

[113] R. Kumar, K. Punera, and A. Tomkins. Hierarchical topic segmentation of websites. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 257–266, New York, USA, 2006. ACM Press.

[114] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the Web graph. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 57–65, Washington, DC, USA, 2000. IEEE Computer Society.

[115] O Kwon. Text categorization based on k-nearest neighbor approach for Web site classification. *Information Processing & Management*, 39(1):25–44, January 2003.

[116] O-W. Kwon and J-H. Lee. Web page classification based on k-nearest neighbor approach. In *Proceedings of the fifth international workshop on on Information retrieval with Asian languages*, pages 9–15, New York, New York, USA, 2000. ACM Press.

[117] O.-W. Kwon and J H Lee. Text categorization based on k-nearest neighbor approach for web site classification. *Information Processing and Management*, 39.(1):25–44, January 2003.

[118] T. Lavergne, T. Urvoy, and F. Yvon. Detecting Fake Content with Relative Entropy Scoring. In *International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection*, volume 377 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[119] B. Lavoie and H. F. Nielsen. Web Characterization Terminology & Definitions Sheet. Technical report, The World Wide Web Consortium (W3C) Working Draft, May 1999.

[120] Jonathan Law, editor. *A Dictionary of Business and Management.* Oxford University Press, 5th edition, 2009.

[121] S. Lawrence, D.M. Pennock, G.W. Flake, R. Krovetz, F.M. Coetzee, E. Glover, F.a. Nielsen, A. Kruger, and C.L. Giles. Persistence of Web references in scientific research. *IEEE Computer*, 34(3):26–31, March 2001.

[122] R. Lewis. Glossary of Terms for Device Independence. Technical report, The World Wide Web Consortium (W3C) Working Draft, August 2003.

[123] R. Lewis. Glossary of Terms for Device Independence. Technical report, The World Wide Web Consortium (W3C) Working Draft, January 2005.

[124] T Li and D Ruan. An extended process model of knowledge discovery in database. *Journal of Enterprise Information Management*, 20(2):169–177, 2007.

[125] W-S. Li, O. Kolak, Q. Vu, and H. Takano. Defining logical domains in a web site. In *Proceedings of the eleventh ACM on Hypertext and hypermedia*, pages 123–132, New York, USA, 2000. ACM.

[126] C. Lindemann and L. Littig. Coarse-grained classification of web sites by their structural properties. In *Proceedings of the eighth ACM international workshop on Web information and data management*, page 35, New York, USA, November 2006. ACM Press.

[127] C. Lindemann and L. Littig. Classifying web sites. In *Proceedings of the 16th international conference on World Wide Web*, page 1143, New York, USA, 2007. ACM Press.

[128] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications).* Springer, 2nd edition, 2011.

[129] Nan Liu and C Yang. Extracting a website's content structure from its link structure. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 345–346, New York, NY, USA, 2005. ACM.

[130] Nan Liu and C. Yang. Mining web site's topic hierarchy. *Special interest tracks and posters of the 14th international conference on World Wide Web*, page 980, 2005.

[131] Nan Liu and Christopher C. Yang. A link classification based approach to website topic hierarchy generation. *Proceedings of the 16th international conference on World Wide Web - WWW '07*, page 1127, 2007.

[132] Y. Liu, Y. Ouyang, H. Sheng, and Z. Xiong. An Incremental Algorithm for Clustering Search Results. In *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pages 112–117, Washington, DC, USA, November 2008. IEEE Computer Society.

[133] L. Lovász. Random walks on graphs: A survey. *YaleU/DCS/TR-1029*, 2:1–46, 1994.

[134] J MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium ...*, 1967.

[135] Z Markov and D T Larose. *Data Mining the Web: Uncovering Patterns in Web Content, Structure, and Usage.* Wiley-Interscience, 2007.

[136] H. Marshall. Creative Retirement; Guide to computing, online training module: Glossary of Computer Terms: 'website'. `http://www.crm.mb.ca/guide/glossary.html`, Nov 2011. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63OJma3pL`.

[137] J. Masanès. *Web Archiving.* Springer-Verlag, Secaucus, NJ, USA, 2006.

[138] S Mayhew. *A dictionary of geography Geography Oxford paperback reference.* Oxford University Press, 2004.

[139] D Millard and M Ross. Web 2.0: hypertext by any other name? In Uffe Kock Wiil, Peter J Nürnberg, and Jessica Rubart, editors, *Hypertext*, pages 27–30. ACM, 2006.

[140] G. Mishne, D. Carmel, and R. Lempel. Blocking Blog Spam with Language Model Disagreement. In *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web*, Chiba, Japan, May 2005.

[141] D. Mladenic. Turning Yahoo into an Automatic Web-Page Classifier. In *13th European Conference on Artificial Intelligence*, 1998.

[142] L. Moussiades and A. Vakali. Mining the Community Structure of a Web Site. In *Fourth Balkan Conference in Informatics*, pages 239–244. IEEE Computer Society, September 2009.

[143] M. Newman. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter ...*, 2004.

[144] M. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):5, June 2004.

[145] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2), February 2004.

[146] J. Nielsen. The Rise of the Subsite. *useit.com Alertbox for september 1996*, September 1996.

[147] G Pant, S Bradshaw, and F Menczer. Search engine-crawler symbiosis: Adapting to community interests. *Research and Advanced Technology for . . .*, 2003.

[148] C. H Papadimitriou and K Steiglitz. Combinatorial optimization: Algorithms and complexity. *Printice-Hall, New Jersey*, 1982.

[149] D T Pham, S S Dimov, and C D Nguyen. An Incremental K-means algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 218(7):783–795, January 2004.

[150] G Piatetsky-Shapiro. Knowledge Discovery in Real Databases: A Report on the IJCAI-89 Workshop. *AI Magazine*, 11(5):68–70, 1991.

[151] G Piatetsky-Shapiro, R Brachman, T Khabaza, W Kloesgen, and E Simoudis. An Overview of Issues in Developing Industrial Data Mining and Knowledge Discovery Applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), eds. J. Han and E. Simoudis*, pages 89–95, 1996.

[152] JM Pierre. On the automated classification of web sites. *Arxiv preprint cs/0102002*, 6(0), 2001.

[153] J. Pitkow and J. Hjelm. W3c web characterization activity statement. `http://www.w3.org/WCA/Activity`, November 1999. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/64PsFfJPr`.

[154] J. Pitkow and H F. Nielsen. W3c web characterization activity terminology sheet. `http://www.w3.org/WCA/Terminology.html`, December 1999. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/63jDa9l8a`.

[155] B. Poblete and R. Baeza-Yates. A content and structure website mining model. In *Proceedings of the 15th international conference on World Wide Web*, page 957, New York, USA, 2006. ACM Press.

[156] X. Qi and B. D. Davison. Web page classification. *ACM Computing Surveys*, 41(2):1–31, February 2009.

[157] D. Raggett, S. Boyera, and R. Lewis. Device independence: Access to a unified web from any device in any context by anyone. `http://www.w3.org/2001/di/`, March 2007. Accessed: 2012-08-29. Archived by WebCite: `http://www.webcitation.org/64gRVzABn`.

[158] S. V. Ramnath and P. Halkarnikar. Web Site Mining Using Entropy Estimation. In *International Conference on Data Storage and Data Engineering*, pages 225–229. IEEE, February 2010.

[159] T. O. Reilly. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications Strategies*, 65(65):17–37, 2007.

[160] E. M. Rodrigues, N. Milic-Frayling, and B. Fortuna. Detection of Web Subsites: Concepts, Algorithms, and Evaluation Issues. In *Web Intelligence*, pages 66–73. IEEE Computer Society, 2007.

[161] E. M. Rodrigues, N. Milic-Frayling, M. Hicks, and G. Smyth. Link Structure Graphs for Representing and Analyzing Web Sites, 2006.

[162] Eduarda Mendes Rodrigues. Web Site Structure Analysis: Concepts, Algorithms and Evaluation Issues, 2008.

[163] G. Salton, a. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, November 1975.

[164] S.E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[165] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, page 1177, New York, New York, USA, April 2010. ACM Press.

[166] R. Sedgewick and K. Wayne. *Algorithms*. Addison-Wesley Professional, 4th edition, 2011.

[167] P. Senellart. Website Identification. Masters thesis dea internship report, Université Paris XI, Orsay, France., September 2003.

[168] P. Senellart. Identifying Websites with Flow Simulation. In David Lowe and Martin Gaedke, editors, *ICWE*, volume 3579 of *Lecture Notes in Computer Science*, Orsay, France., 2005. Gemo, INRIA Futurs., Springer.

[169] J. Seo, F. Diaz, E. Gabrilovich, V. Josifovski, and B. Pang. Generalized link suggestions via web site clustering. In *Proceedings of the 20th international conference on World wide web*, page 77, New York, New York, USA, March 2011. ACM Press.

[170] N. Shadbolt and T. Berners-Lee. Web science emerges. *Scientific American*, 299(4):76–81, 2008.

[171] B. Shneiderman. Web science: a provocative invitation to computer science. *Communications of the ACM*, 50(6):25–27, 2007.

[172] K. Sigurdsson. Incremental crawling with Heritrix. In *Proceedings of the 5th International Web Archiving Workshop*, pages 1–12, 2005.

[173] C. Soanes and A. Stevenson, editors. *Oxford dictionary of English.* Oxford University Press New York, 2003.

[174] C. Soanes and A. Stevenson, editors. *The Concise Oxford English Dictionary.* Oxford University Press, Oxford., 12th edition, 2008.

[175] A. Stevenson, editor. *Oxford Dictionary of English.* Oxford University Press, Oxford, 3rd edition, 2010.

[176] P. N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining.* Pearson International Edition, 2006.

[177] Q. Tan, Z. Zhuang, P. Mitra, and C. L. Giles. Efficiently detecting webpage updates using samples. In *Proceedings of the 7th international conference on Web engineering*, pages 285–300, Berlin, Heidelberg, July 2007. Springer-Verlag.

[178] Y. Tian. A web site mining algorithm using the multiscale tree representation model. In *Proceedings of the 5th Webmining as a Premise to Effective and Intelligent Web Applications*, 2003.

[179] Y-h. Tian, T-j. Huang, and W. Gao. Two-phase Web site classification based on Hidden Markov Tree models. *Web Intelligence and Agent Systems*, 2(4):249–264, 2004.

[180] M Toyoda and M Kitsuregawa. What's really new on the web?: identifying new pages from a series of unstable web snapshots. In *Proceedings of the 15th international conference on World Wide Web*, pages 233 – 241, New York City, USA, 2006. ACM.

[181] T. Urvoy, E. Chauveau, P. Filoche, and T. Lavergne. Web Spam Challenge 2007: France Telecom R&D Submission. *Web Spam Challenge 2007*, May 2007.

[182] T. Urvoy, E. Chauveau, P. Filoche, and T. Lavergne. Tracking Web spam with HTML style similarities. *ACM Transactions on the Web (TWEB)*, 2(1):1–28, February 2008.

[183] T. Urvoy, T. Lavergne, and P. Filoche. Tracking Web Spam with Hidden Style Similarity. In *2nd International workshop on Adversarial Information Retrieval on the Web*, pages 25–31, Seattle, Washington, USA, 2006.

[184] I. H. Witten and E. Frank. *Data Mining: practical machine learning tools and techniques.* Morgan Kaufman, 2005.

[185] Jie Wu. Using SiteRank for P2P Web Retrieval. Technical Report 5005, School of Computer and Communication Sciences Swiss Federal Institute of Technology, 2004.

[186] Jie Wu and Karl Aberer. Using SiteRank for Decentralized Computation of Web Document Ranking. In *Proceedings of the 3rd Intl. Conference on Adaptive Hypermedia and Adaptive WebBased Systems*, pages 265–274, 2004.

[187] Wensi Xi, Edward A Fox, Roy P Tan, and Jiang Shu. Machine Learning Approach for Homepage Finding Task. In Alberto H F Laender and Arlindo L Oliveira, editors, *SPIRE*, volume 2476 of *Lecture Notes in Computer Science*, pages 145–159. Springer, 2002.

[188] Christopher C. Yang and Nan Liu. Web site topic-hierarchy generation based on link structure. *Journal of the American Society for Information Science and Technology*, 60(3):495–508, March 2009.

[189] Ying Zhao and George Karypis. Clustering in Life Sciences. *Functional Genomics: Methods and Protocols, M. Brownstein, A. Khodursky and D. Conniffe (editors).*, 2003.

[190] Uri Zwick. The smallest networks on which the Ford-Fulkerson maximum flow procedure may fail to terminate. *Theoretical Computer Science*, 148(1):165–170, August 1995.

Appendix A

Real Data Graph (RDG) Cluster Examples

The work described in this chapter presents additional illustrations of the four RDG data sets. The Figure A.1 present an illustration of the LivChem RGD data set using a selection of graph layout methods to demonstrate the dense and complex characteristics of the real data. Figure A.2 present an illustration of the LivChem RGD data set using the same selection of graph layout methods but this time with a reduced scale, which demonstrate the high degree nodes (hubs/authorities) as dark patches. Figure A.3 present an illustration of the LivChem RGD data set again using the same selection of graph layout methods but this time the high degree nodes (hubs/authorities) are removed to demonstrate the complexities of the average degree nodes. Figure A.4 presents an illustrations of the target and noise clusters, the connections of low degree nodes are gradually removed to demonstrate the complex connection between the clusters.

The same illustrations for the LivChem data set above are shown in Figures A.5, A.6, A.7, A.8, Figures A.9, A.10, A.11, A.12, and Figures A.13, A.14, A.15, A.16 but with respect to the LivHistory, LivMath and LivSace data sets respectfully.

Figure A.1: The whole LivChem graph displayed using a selection of graph layout methods.

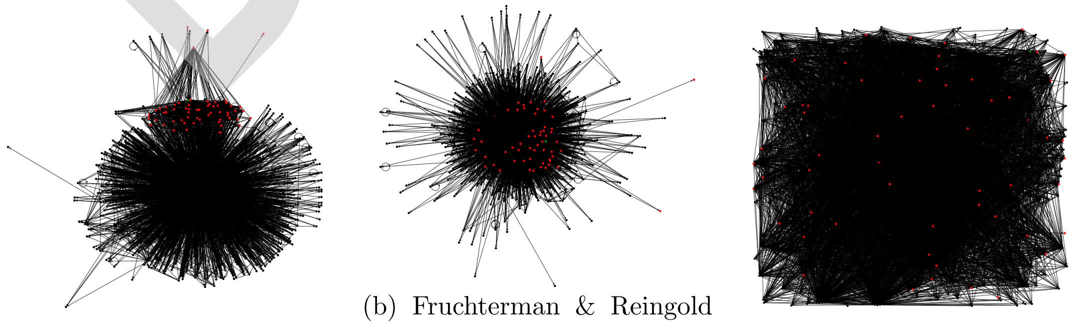

(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.2: The LivChem graph using a reduced scale to highlight the high degree of connectivity. Dark patches indicate vertices and edges with a high degree. Areas of nodes that have lower connectivity are greyed out of the image.

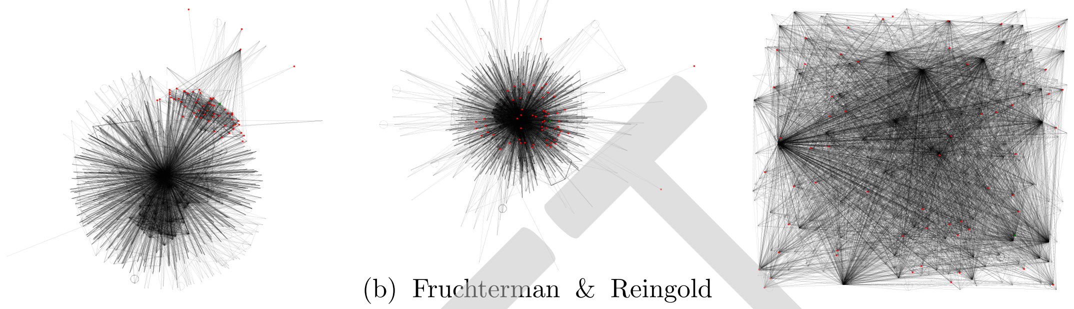

(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.3: The LivChem graph filtering vertices with degree > 30, displayed with layout method as described. This essentially greys out hubs/authorities with high degree, and shows the edges and vertices with average degree more clearly.

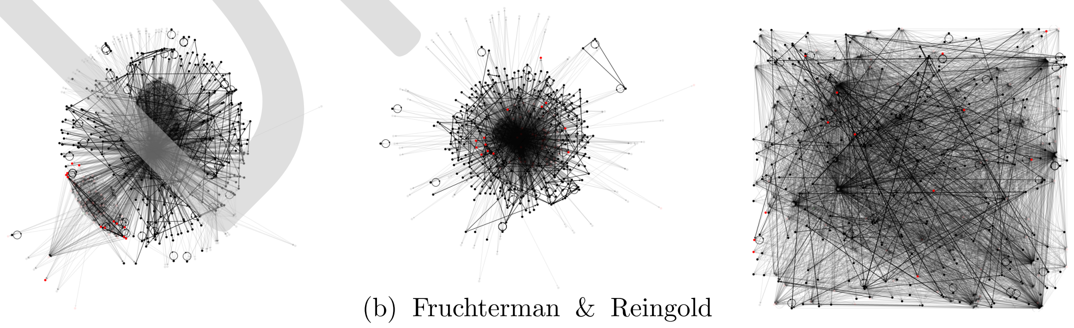
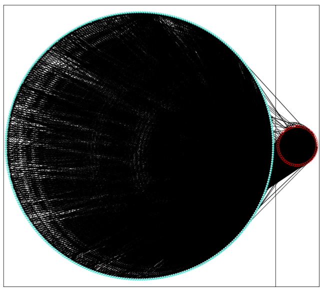
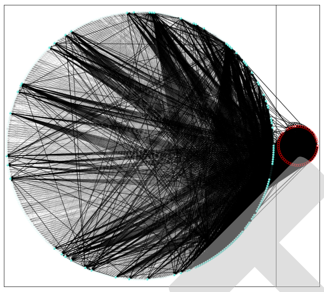
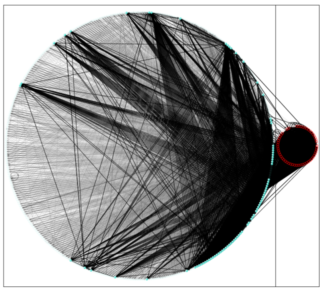
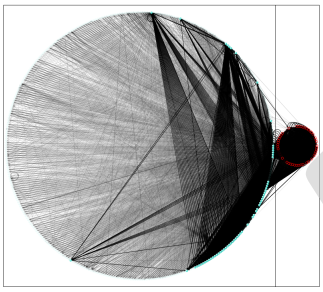

(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

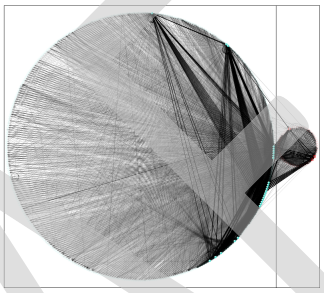
Figure A.4: The target and noise clusters of the LivChem graph displayed in a circular layout. The graphs are filtered to remove vertices and edges with degree less than 1, 15, 20, 30, 70 and 90, figures (a) to (f) respectively.


(a)


(b)


(c)


(d)


(e)

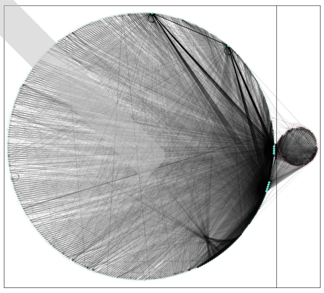
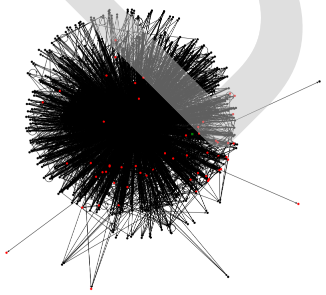
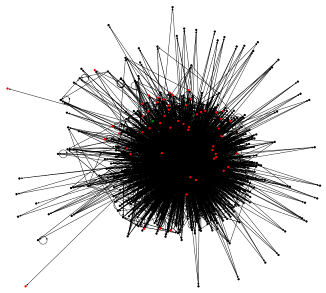

(f)

Figure A.5: The LivHistory graph displayed using a selection of graph layout methods.


(a) Harel & Koren [96] layout


(b) Fruchterman & Reingold [86] layout

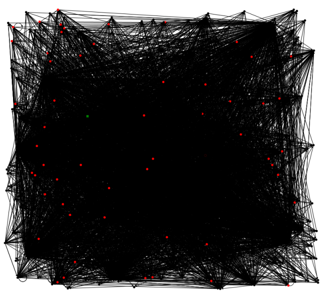
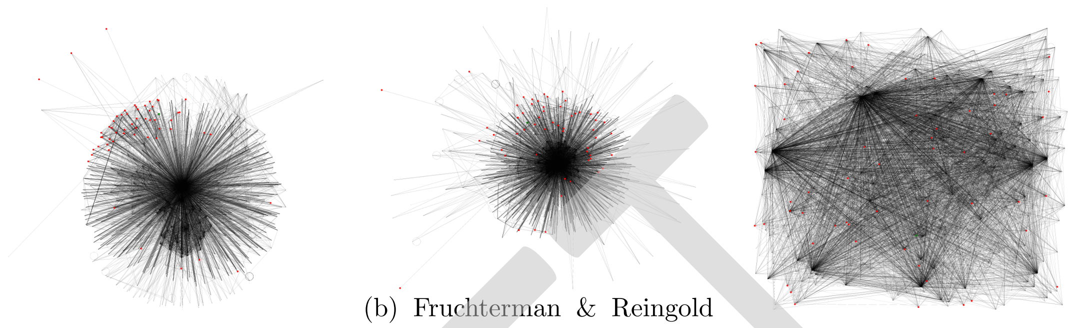
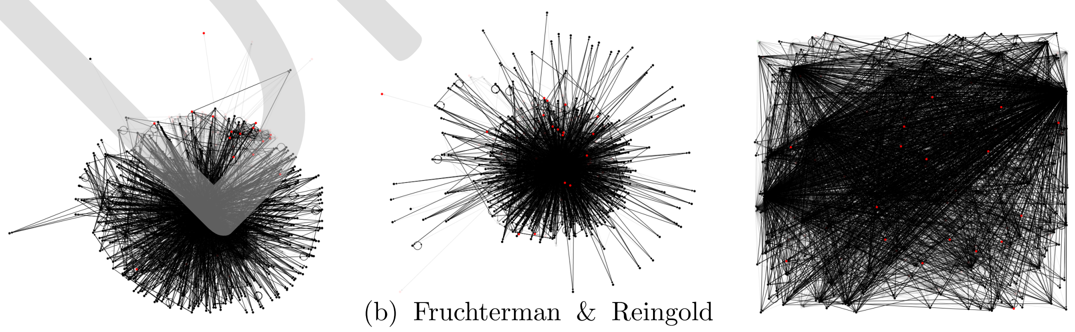

(c) Random layout

Figure A.6: The LivHistory graph using a reduced scale to highlight the high degree of connectivity, Dark patches indicate vertices and edges with a high degree. Areas of nodes that have lower connectivity are greyed out of the image.



(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.7: The LivHistory graph filtering vertices with degree > 30, displayed with layout method as described. This essentially greys out hubs/authorities with high degree, and shows the edges and vertices with average degree more clearly.



(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.8: The target and noise clusters of the LivHistory graph displayed in a circular layout. The graphs are filtered to remove vertices and edges with degree less than 1, 15, 20, 30, 70 and 90 figures (a) to (f) respectively.

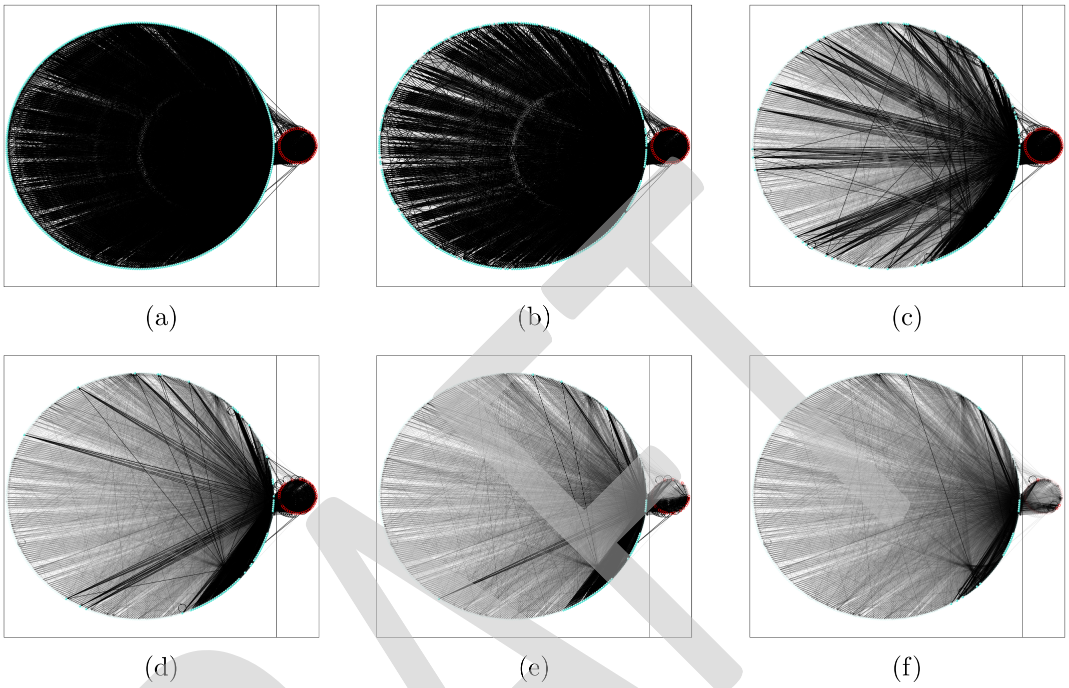
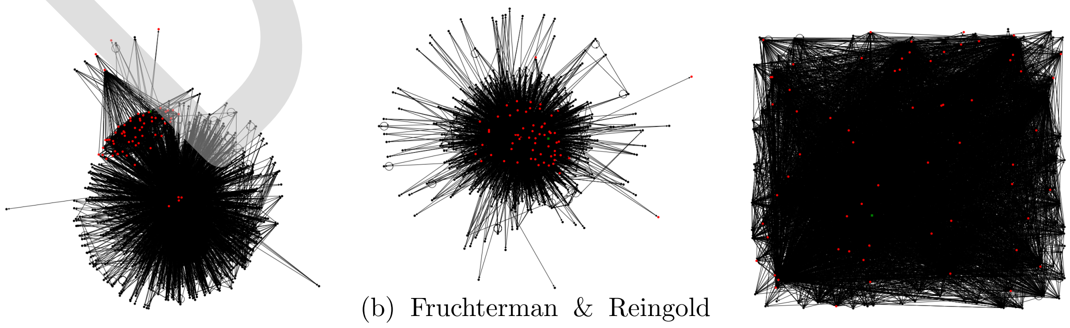
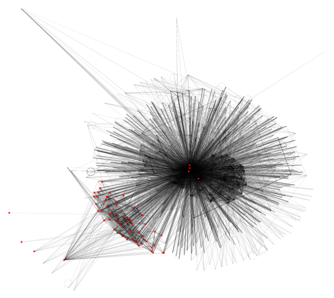


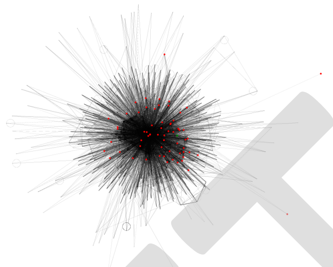
Figure A.9: The LivMath graph displayed using a selection of graph layout methods.



(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.10: The LivMath graph using a reduced scale to highlight the high degree of connectivity. Dark patches indicate vertices and edges with a high degree. Areas of nodes that have lower connectivity are greyed out of the image.


(a) Harel & Koren [96] layout


(b) Fruchterman & Reingold [86] layout

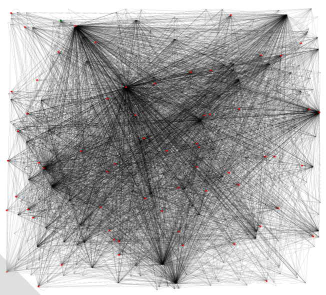
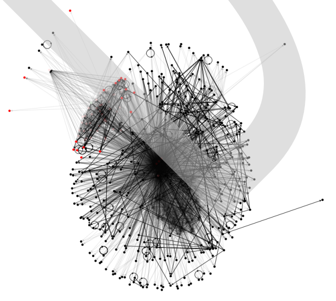
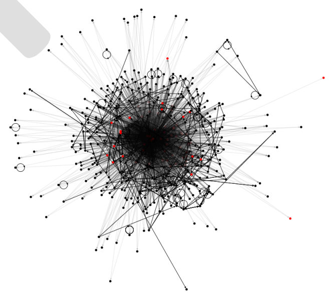

(c) Random layout

Figure A.11: The LivMath graph filtering vertices with degree > 30, displayed with layout method as described. This essentially greys out hubs/authorities with high degree, and shows the edges and vertices with average degree more clearly.


(a) Harel & Koren [96] layout


(b) Fruchterman & Reingold [86] layout

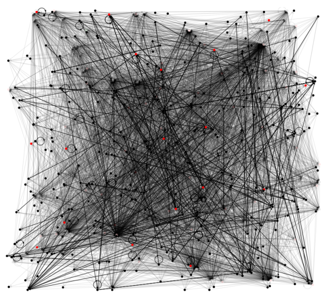
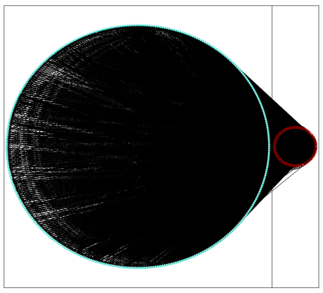
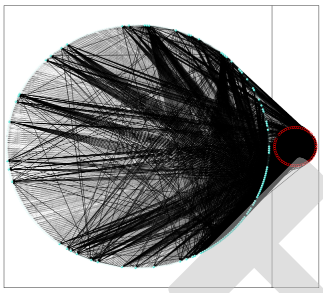
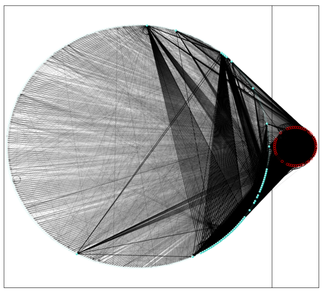
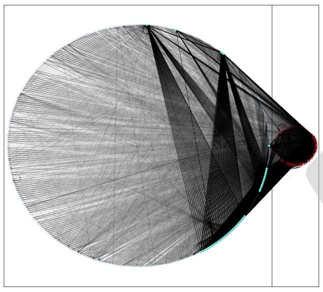

(c) Random layout

Figure A.12: The target and noise clusters of the LivMath graph displayed in a circular layout. The graphs are filtered to remove vertices and edges with degree less than 1, 15, 30, 45, 60 and 90, figures (a) to (f) respectfully.


(a)


(b)


(c)


(d)

(e)

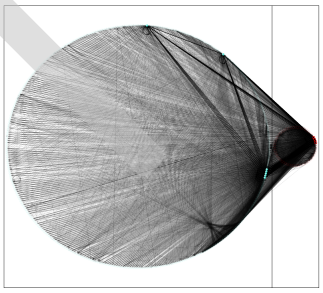
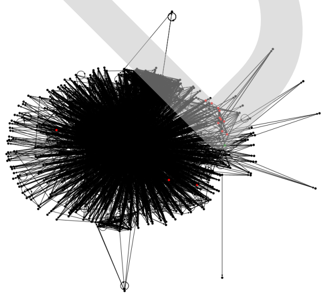
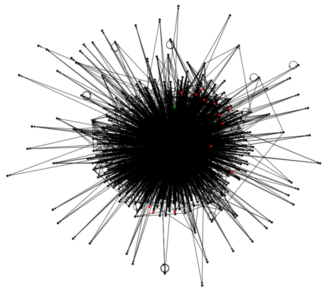

(f)

Figure A.13: The LivSace graph displayed using a selection of graph layout methods.


(a) Harel & Koren [96] layout


(b) Fruchterman & Reingold [86] layout

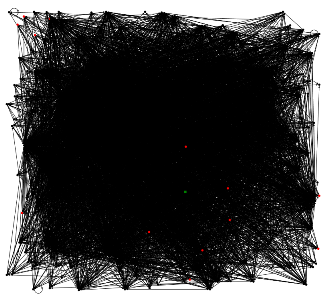
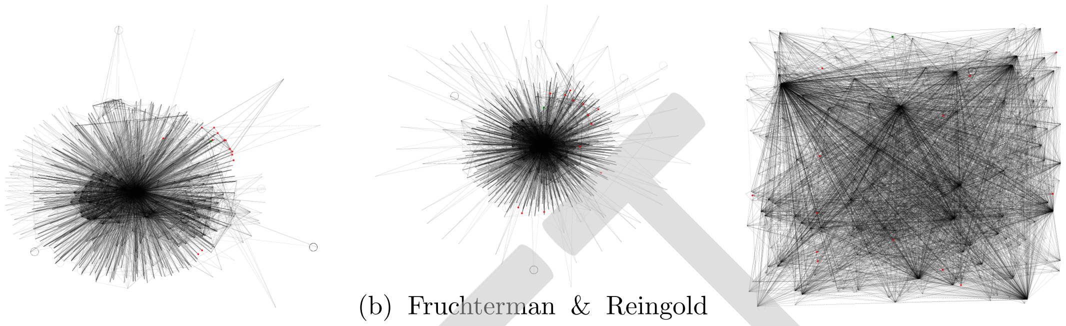
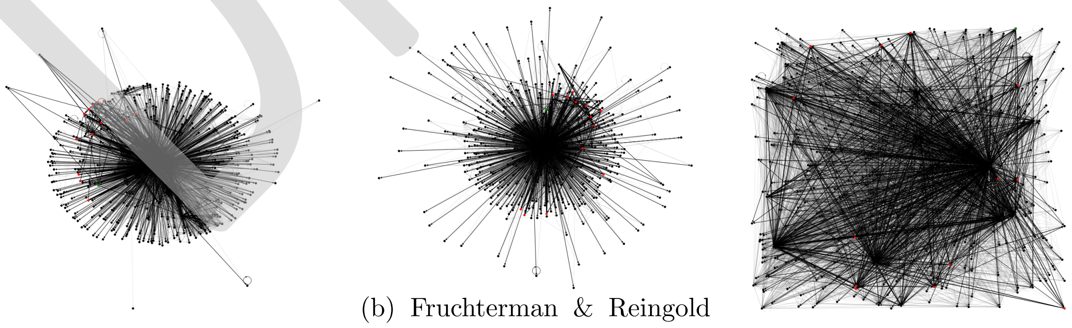

(c) Random layout

Figure A.14: The LivSace graph using a reduced scale to highlight the high degree of connectivity. Dark patches indicate vertices and edges with a high degree. Areas of nodes that have lower connectivity are greyed out of the image.



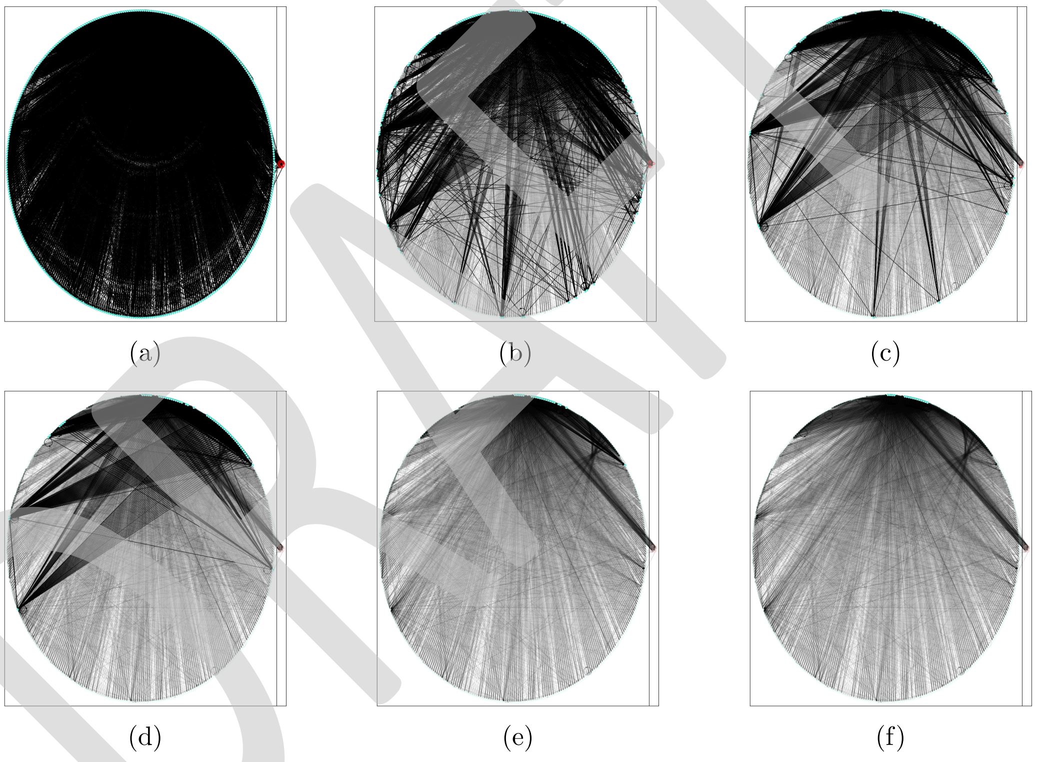
(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

Figure A.15: The LivSace graph filtering vertices with degree > 30, displayed with layout method as described. This essentially greys out hubs/authorities with high degree, and shows the edges and vertices with average degree more clearly.



(a) Harel & Koren [96] layout (b) Fruchterman & Reingold [86] layout (c) Random layout

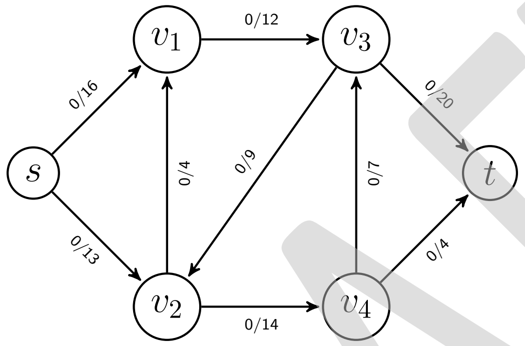
Figure A.16: The target and noise clusters of the LivSace graph in a circular layout. The graphs are filtered to remove vertices and edges with degree less than 1, 5, 10, 20, 40 and 90 figures (a) to (f) respectfully.

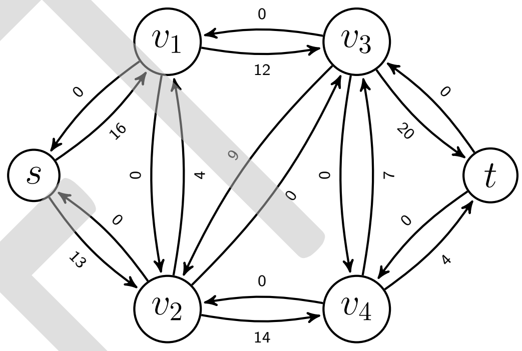


Appendix B

Ford Fulkerson Example

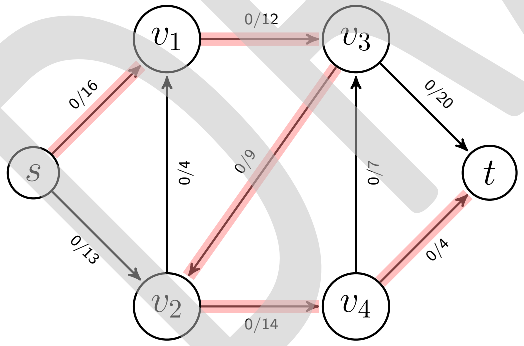
Ford-Fulkerson algorithm exmaple.

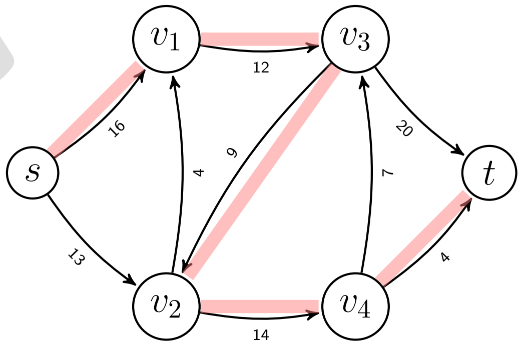


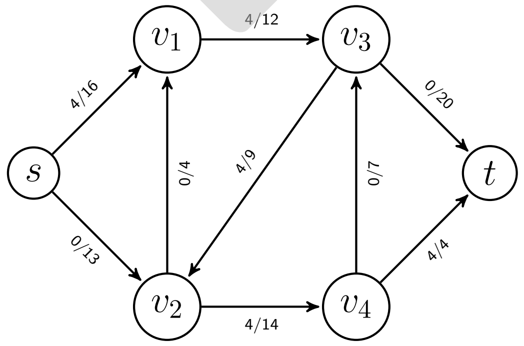


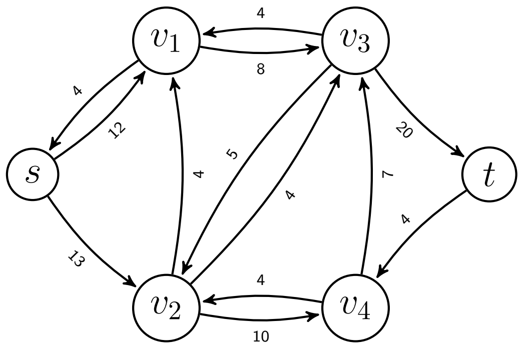
Given the network in Figure 6.10a, shown above is the corresponding flow network G (left) and residual network G_f (right). Notice there is no flow f in G at the initial point. In subsequent iterations edges with $c_f = 0$ in G_f are not shown for clarity.

Iteration 1









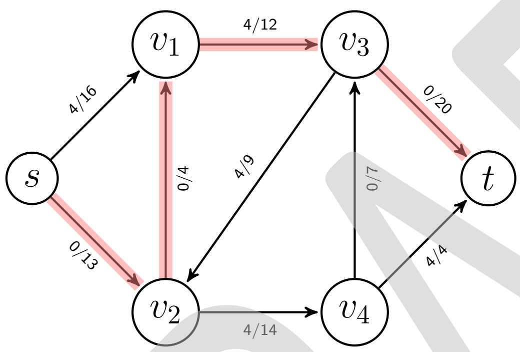
The first augmenting path p_1 in G_f is $\{s \to v_1 \to v_3 \to v_2 \to v_4 \to t\}$, as highlighted (right). The maximum residual capacity c_f of path p_1 in G_f is $c_f(p_1) = 4$ which is restricted by edge $(v_4, f) = 4$ maximum residual capacity. 4 units of flow is admitted along augmenting path p_1. The resulting flow network and corresponding residual network is shown.

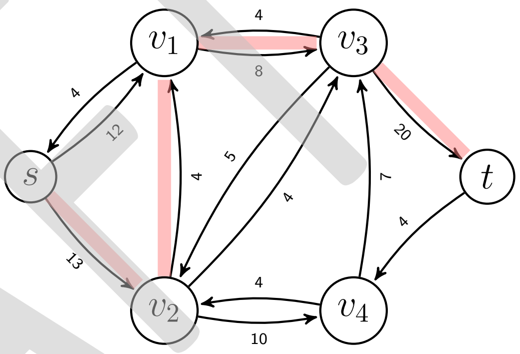
The residual network is construed as explained above. Notice for edge (v_4, t) there is no corresponding residual capacity c_f in G_f, After the flow has been admitted along path p_1 this edge could no longer be considered an as part of an augmenting path from s to t.

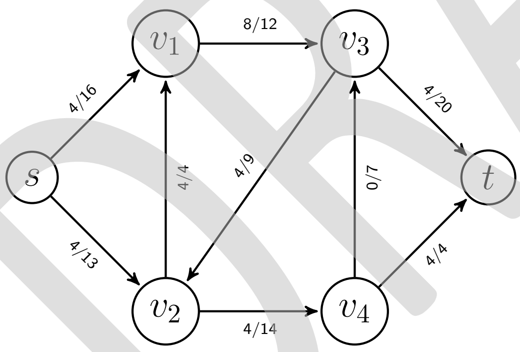
$$\begin{aligned} &v_4 \to t \\ &c_f(v_4,t) &&= c(v_4,t) - f(v_4,t) \\ & &&= 4 - 4 \\ & &&= 0 \end{aligned}$$

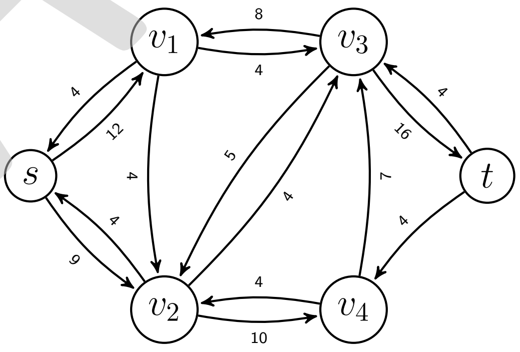
$$\begin{aligned} &t \leftarrow v_4 \\ &c_f(v_4,t) &&= f(v_4,t) \\ & &&= 4 \end{aligned}$$

Iteration 2









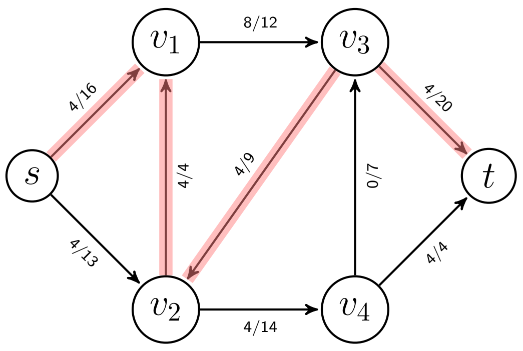
The augmenting path p_2 in G_f is $\{s \to v_2 \to v_1 \to v_3 \to t\}$, as highlighted (right). The maximum residual capacity $c_f(p_2) = 4$ which is restricted by edge (v_2, v_1) maximum residual capacity. 4 units of flow is admitted along augmenting path p_2. The resulting flow network and corresponding residual network is shown.

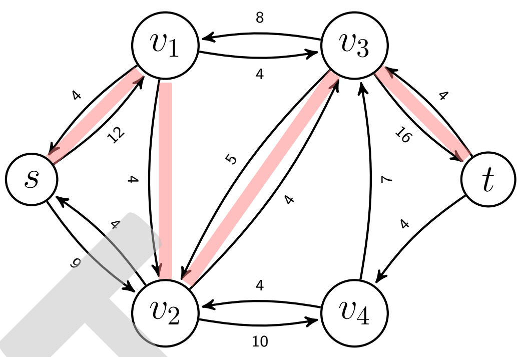
Notice the edge (v_2, v_1) in the resulting residual graph G_f . The available residual capacity c_f effectively changes direction from $c_f(v_2, v_1) = 4$ to $c_f(v_2, v_1) = 0$ and $c_f(v_1, v_2) = 0$ to $c_f(v_1, v_2) = 4$. This is due to the canselation effect.

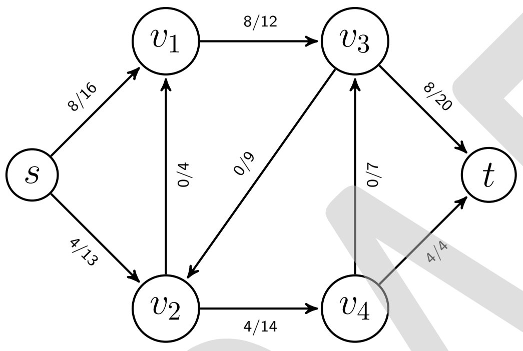
$$\begin{aligned} &v_2 \rightarrow v_1 \\ c_f(v_2, v_1) &= c(v_2, v_1) - f(v_2, v_1) \\ &= 4 - 4 \\ &= 0 \end{aligned}$$

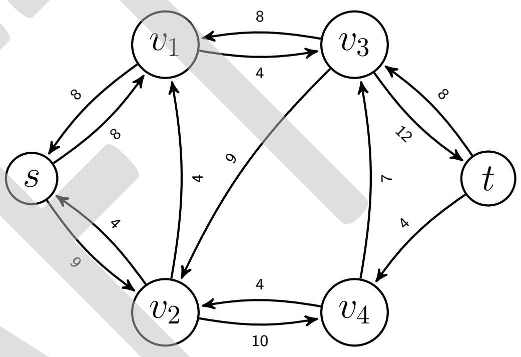
$$\begin{aligned} &v_1 \rightarrow v_2 \\ c_f(v_1, v_2) &= f(v_1, v_2) \\ &= 4 \end{aligned}$$

Iteration 3









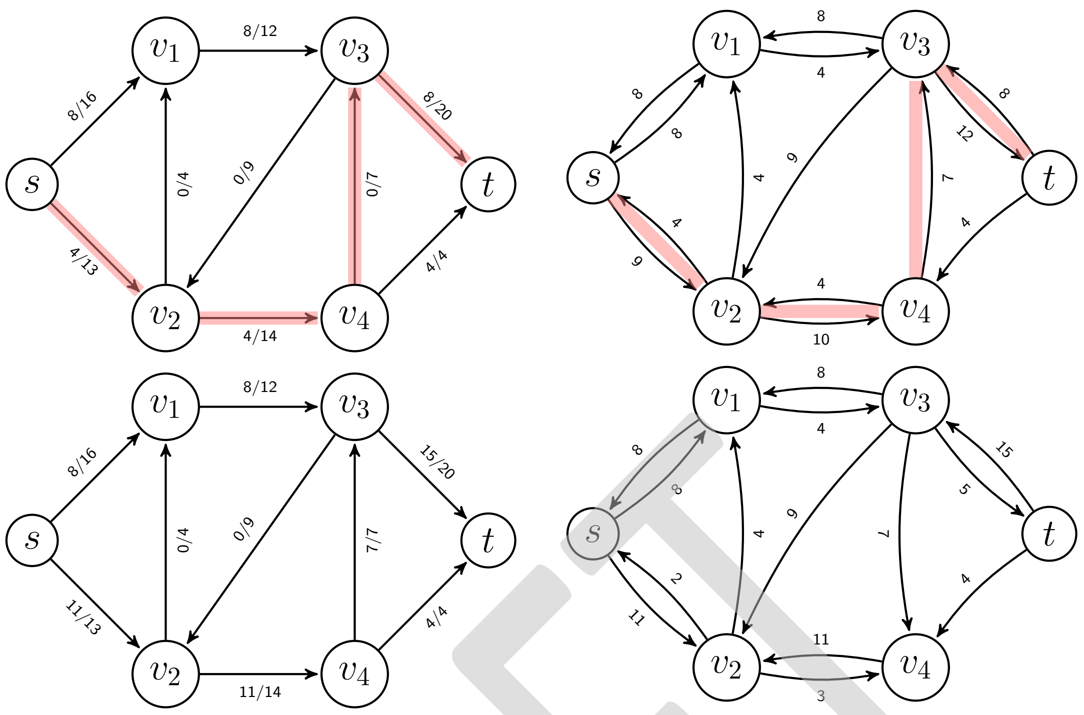
The augmenting path p_3 is $\{s \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow t\}$, as highlighted (right). The maximum residual capacity $c_f(p_3) = 4$ which is restricted by edges (v_1, v_2) and (v_2, v_3) maximum residual capacity. 4 units of flow is admitted along augmenting path p_3. The resulting flow network and corresponding residual network is shown.

Notice edge $(v_3 v_2)$ in which flow is sent from v_2 to v_3 producing a cancelling effect. The flow network shows that edge $(v_3 v_2)$ previously have flow $4/9$ in direction $v_3 \rightarrow v_2$. The augmented path p_3 has an edge (v_2, v_3) which admits 4 units of flow in the opposite direction $v_2 \rightarrow v_3$. The resulting net flow is 0 produced by the cancelling effect. The cancelling effect is essential for algorithms to change the direction of flow in order to maximise the amount of flow in any given direction.

$$\begin{aligned} &v_3 \rightarrow v_2 \\ c_f(v_3, v_2) &= c(v_3, v_2) - f(v_3, v_2) \\ &= 9 - 0 \\ &= 9 \end{aligned}$$

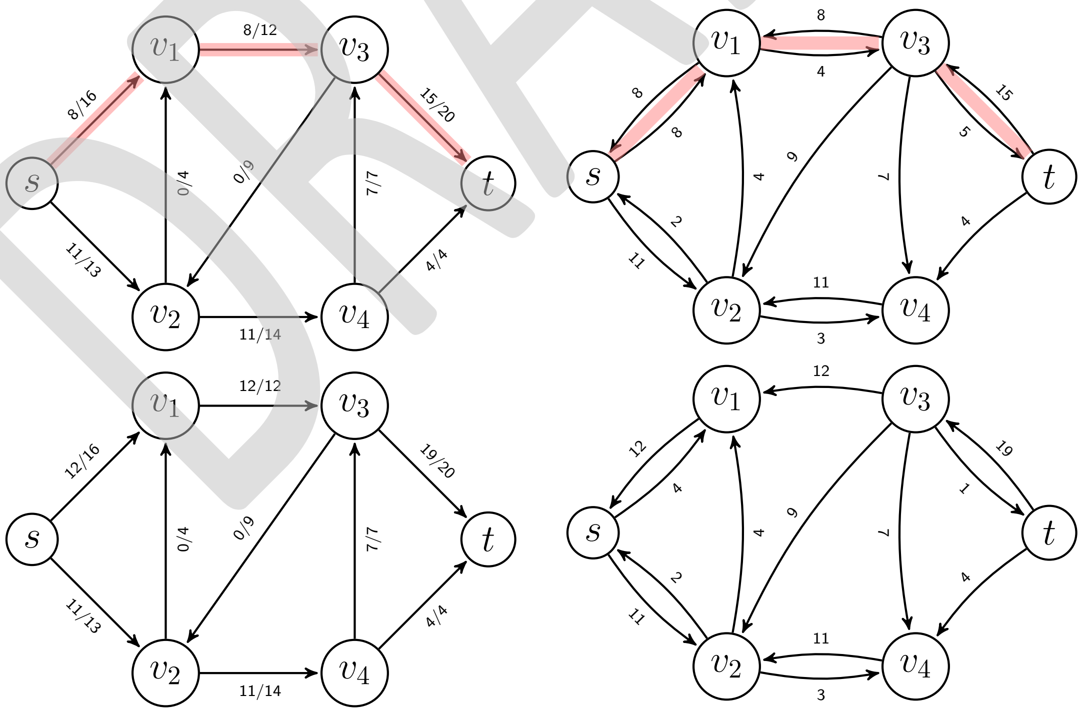
$$\begin{aligned} &v_2 \rightarrow v_3 \\ c_f(v_2, v_3) &= f(v_3, v_2) \\ &= 0 \end{aligned}$$

Iteration 4



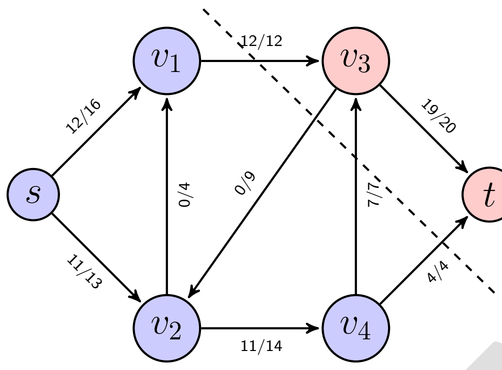
The augmenting path p_4 is $\{s \rightarrow v_2 \rightarrow v_4 \rightarrow v_3 \rightarrow t\}$, as highlighted (right). The maximum residual capacity $c_f(p_4) = 7$ which is restricted by edges (v_4, v_3) maximum residual capacity. 7 units of flow is admitted along augmenting path p_4. The resulting flow network and corresponding residual network is shown.

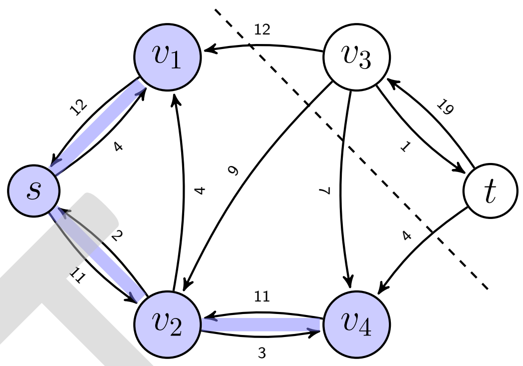
Iteration 5



The augmenting path p_5 is $\{s \rightarrow v_1 \rightarrow v_3 \rightarrow t\}$, as highlighted (right). The maximum residual capacity $c_f(p_5) = 4$ which is restricted by edges (v_1, v_3) maximum residual capacity. 4 units of flow is admitted along augmenting path p_5. The resulting flow network and corresponding residual network is shown.

There are no more augmenting paths from s to t in the residual network G_f. The resulting residual network can be used to make a cut producing set $S = \{s, v_1, v_2, v_4\}$ and set $T = \{v_3, t\}$.





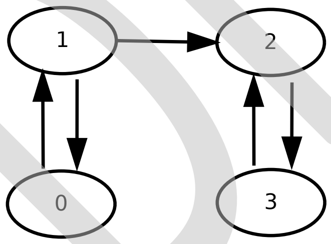
Appendix C

Newmans Graph Partitioning Algorithm Example

Newmans algorithm example. Recall section 6.1.2, given graph $G = (V, E)$ (represented in Figure C.1), on each iteration it will be shown how the Newmans algorithm calculates:

- e_{ii}. The fraction of edges in each cluster.
- a_i. The faction of edges of G then end in cluster i
- a_i^2. The value a^2 is the fraction of edges in cluster i if edges where connected uniformly at random in graph G.

Figure C.1: An example graph G1, the corresponding adjacency list and diagram is shown.



Diagram

Vertex	Edges
0	1
1	0, 2
2	3
3	2

Adjacency List

Iteration 0
The values mentioned above calculated at the start of the process, when each vertex is considered to represent a single group.
The Q value for this the graph at the start of the process, calculated as
$Q = (0.0 - 0.01) + (0.0 - 0.01) + (0.0 - 0.04) + (0.0 - 0.01)$
$= -0.01 - 0.01 - 0.04 - 0.01 = -0.07$.

Table C.1: Newman exmaple: Iteration 0

i	e_{ii}	a_i	a_i^2
0	0	0.1	0.01
1	0	0.1	0.02
2	0	0.2	0.04
3	0	0.1	0.01

Iteration 1

In the first iteration of the algorithm, there are six potential joins to be made ({0,1}, {0,2}, {0,3}, {1,2}, {1,3} and {2,3}). Using the Q value as the criteria, the best join configuration is {{0,1}, {2}, {3}} with $Q = 0.04$.

Table C.2: Newman example: Iteration 1

Groups			Internal links									
1	2	3	e_{11}	e_{22}	e_{33}	a_1	a_2	a_3	$a1^2$	$a2^2$	$a3^2$	Q
{0,1}	{2}	{3}	0.4	0	0	0.4	0.4	0.2	0.16	0.16	0.04	0.04
{0,2}	{1}	{3}	0	0	0	0.6	0.2	0.2	0.36	0.04	0.04	-0.44
{0,3}	{1}	{2}	0	0	0	0.4	0.2	0.4	0.16	0.04	0.16	-0.36
{1,2}	{0}	{3}	0.2	0	0	0.6	0.2	0.2	0.36	0.04	0.04	-0.24
{1,3}	{0}	{2}	0	0	0	0.4	0.2	0.4	0.16	0.04	0.16	-0.36
{2,3}	{0}	{1}	0.4	0	0	0.6	0.2	0.2	0.36	0.04	0.04	-0.04

Iteration 2

There are three possible configurations {{0,1,2}, {3}}, {{0,1,3}, 2} and {{0,1}, {2,3}}. The best join configuration is {{0,1}, {2,3}} with $Q = 0.28$.

Table C.3: Newman example: Iteration 2

Groups		Internal links						
1	2	e11	e22	a1	a2	$a1^2$	$a2^2$	Q
{0,1,2}	{3}	0.6	0	0.8	0.2	0.64	0.04	-0.08
{0,1,3}	{2}	0.4	0	0.6	0.4	0.36	0.16	-0.12
{0,1}	{2,3}	0.4	0.4	0.4	0.6	0.16	0.36	0.28

Iteration 3 (Final)

The third and final iteration is a simple process of combining the all vertices into one cluster, thus giving $Q = 0$. The best grouping configuration is said to be {{0,1}, {2,3}} with $Q = 0.28$. As this was the highest value of Q over all iterations of the algorithm.