User-Driven Navigation Pattern Discovery from Internet Data*

M. Baumgarten[1], A.G. Büchner[1], S.S. Anand[2], M.D. Mulvenna[3] and J.G. Hughes[1]

[1] Northern Ireland Knowledge Engineering Laboratory, University of Ulster
{m.baumgarten, ag.buchner, jg.hughes}@ulst.ac.uk
[2] School of Information and Software Engineering, University of Ulster
ss.anand@ulst.ac.uk
[3] MINE*it* Software Ltd, Faculty of Informatics, University of Ulster
maurice@mineit.com

Abstract. Managers of electronic commerce sites need to learn as much as possible about their customers and those browsing their virtual premises, in order to maximise the return on marketing expenditure. The discovery of marketing related navigation patterns requires the development of data mining algorithms capable of the discovery of sequential access patterns from web logs. This paper introduces a new algorithm called M*i*DAS that extends traditional sequence discovery with a wide range of web-specific features. Domain knowledge is described as flexible navigation templates that can specify generic navigational behaviour of interest, network structures for the capture of web site topologies, concept hierarchies and syntactic constraints. Unlike existing approaches M*i*DAS supports sequence discovery from multidimensional data, which allows the detection of sequences across monitored attributes, such as URLs and http referrers. Three methods for pruning the sequences, resulting in three different types of navigational behaviour are presented. The experimental evaluation has shown promising results in terms of functionality as well as scalability.

1 Introduction

Direct marketing is the process of identifying likely buyers of products or services and promoting them accordingly [12]. The difference between traditional and electronic commerce marketing is the availability of more detailed data, the necessity for the incorporation of web marketing-specific domain knowledge, the potential application of more sophisticated direct marketing strategies, and, thus, the requirement for more assorted data mining goals [16]. A key to discovering marketing intelligence in electronic businesses is that of finding navigational patterns, which can be used for online promotion and personalisation activities. The objective of this

* This research has partly been funded by the ESPRIT project Nº 26749 (MIMIC — Mining the Internet for Marketing IntelligenCe).

paper is to describe a novel method for discovering marketing-driven navigation patterns from Internet log files.

The outline of the paper is as follows. In Section 2, the structure and content of web log files is described, which is accomplished by web-specific domain knowledge, namely navigation templates, topology networks, concept hierarchies and syntactic constraints. In Section 3, the algorithmic navigation pattern discovery, which has been termed M*i*DAS (Mining Internet Data for Associative Sequences) is described. In Section 4, a case study is presented, which demonstrates the application of the proposed research. In Section 5, the experimental evaluation of M*i*DAS is presented, which includes complexity measurements, as well as performance results. In Section 6, related work is evaluated, before Section 7 concludes with a summary of contributions and the outline of further work.

2 Web Data and Domain Knowledge

This section describes the structure and content of web log files as well as different types of supported web-specific domain knowledge, which include syntactic constraints, navigation templates, network topologies and concept hierarchies.

2.1 Web Log Files

The data available in web environments is three-fold and includes server data in the form of log files, web meta data representing the structure of a web site, and marketing transaction information, which depends on the products and services provided. For the purpose of this paper it is assumed that goal-orientated materialised views have been created *a priori*, for instance, as part of a web log data warehouse [6]. Thus, this paper concentrates on the core activity of discovering navigational patterns in the form of web-specific sequences from pre-processed Internet log files.

The data input for M*i*DAS is a set of navigations sorted by primary and secondary key. The structure of a log file consists of a primary key (for instance, session id, customer id, cookie id, etc.), a secondary key (date and time related information, such as login time), and a sequence of hits, which holds the actual data values (for example, URLs or http referrers). Web meta data is treated as domain knowledge by M*i*DAS rather than input data (see Section 2.2.3).

Definition 1. A log file L is defined as a sequence of navigations $L = <N_1, N_2, N_3, \ldots>$. Each N_i is of the form (a, b, H), a representing the primary key, b the secondary key, and H a non-empty sequence $H_i = <h_1, h_2, h_3, \ldots>$, where each h_i is a hit which represents a single web page access.

Table 1 below shows five records of an example log file that is used throughout the paper for demonstration purposes.

Host	Date / Time	Referrer	Hit	Hit	Hit
1	01/06/99 16:48:27	ecom.infm.ulst.ac.uk/	/	/products	/products-emw
1	12/06/99 14:08:43	kdnuggets.com/sift/t-textweb.html	/products-emw	/products	/products-capri
2	24/05/99 06:34:24	kdnuggets.com/solutions/internet-mining.html	/	/company	
3	03/06/99 12:14:20	kdnuggets.com/sift/t-textweb.html	/products-emw	/products	/products-capri
3	03/06/99 15:47:03	kdnuggets.com/solutions/internet-mining.html	/	/company	

Table 1 Example Log Data

In Table 1, the primary key is the host identifier and the secondary key is the navigation session, which is identified by the Date/Time field. The host identifier may be an IP address, cookie, login name or any other key that identifies an individual as the browser of the web site.

2.2 Domain Knowledge Specification

In order to discover web-specific sequential patterns, domain knowledge may be incorporated with the objective to constrain the search space of the algorithm, reduce the quantity of patterns discovered and increase the quality of the discovered patterns [2]. For the purpose of discovering marketing intelligence from Internet log files, four web-specific types of domain knowledge are supported, namely syntactic constraints, navigation templates, topology networks, and concept hierarchies.

2.2.1 Syntactic Constraints

Syntactic constraints for M*i*DAS are expressed as the threshold sextuple $\tau = (\sigma, \delta, \lambda^-, \lambda^+, \gamma^-, \gamma^+)$. $\sigma \in [0,1]$ represents the minimum support and $\delta \in [0,1]$ the minimum confidence, which are identical to their counterparts in traditional association and sequence discovery algorithms. λ^- and λ^+ specify the minimum and maximum length of a sequence, respectively. Through this mechanism, it is possible to eliminate shallow navigational patterns of non-active users, as well as their opposite. γ^- and γ^+ represent the minimum and maximum time gap between two hits, which facilitates the extirpation of search robots (with a very small gap) and also enables a limit to be set for the time a browser spends on one page.

2.2.2 Navigation Templates

In order to perform goal-driven navigation pattern discovery it is expected that a virtual shopper has passed through a particular page or a set of pages. This can include start, end, as well as middle pages. A typical start locator is the home page; a typical middle page of a site is a URL providing information about a special marketing campaign; and a regularly specified end page, where a purchase can be finalised. For simplification, all three constructs are accumulated to *navigation templates*, where a template consists of constants, wildcards, and predicates restricting the permissible values of the wildcards.

Definition 2. A navigation template T is a generalised navigation of a web site, defined as $T = \{t_1, t_2, t_3, \ldots\}$. Each t_i is a non-empty sequence of hits h_i, where each item h_k is either a page or a placeholder taken from {*, ?}. The placeholders * and ? have the same semantics as in classic string matching.

An example here illustrates the specified additions to standard sequences, in order to specify regular expression constraints in the form of navigation templates. Imagine the analysis of a marketing campaign within an online bookstore, introducing reduced gifts (line 1, item 3). Only those customers who have navigated through the site's home page (line 1, item 1) are of interest, and only transactions that have led to purchases are to be considered (line1, item 5) at the same or at a different visit. Furthermore, the standard special offers are to be excluded from the analysis (lines 2-4). This navigation template is shown in Fig. 1.

```
[
(1) <index.htm | * | /offers/gifts.htm ; * , purchase.htm | ?>
(2) ^<* ; offers/reduced.htm ; *>
(3) ^<* ; offers/junk.htm ; *>
(4) ^<? ; offers/2ndhand.htm ; *>
]
```

| *same visit*	, *across visits*	; either same or across visit
* *wildcard*	? *place holder*	
^ *negation*		

Fig. 1. Example Navigation Template

Consider the following extract from a log file:

Host	Date / Time	Hit	Hit	Hit
1	01/06/99 16:48:27	/index.htm	offers/gifts.htm	
1	12/06/99 14:08:43	purchase.htm	offers/gifts.htm	
2	24/05/99 06:34:24	/index.htm	offers/gifts.htm	purchase.htm
3	03/06/99 12:14:20	/index.htm	offers/junk.htm	
3	03/06/99 15:47:03	purchase.htm		

Given the navigation template in Figure 1, only the navigation by Host 1 will satisfy this template. Host 2 also carries out a similar navigation, however, as the purchase is undertaken on the same visit as the hit on 'offers/gifts.htm' this navigation violates the template that specifically specifies that the purchase must be in a separate visit by using the comma. The use of the semi-colon in (1), before the asterisk, implies that the browser may navigate the web site on a number of separate visits between the visit in which he/she visits the 'offers/gifts.htm' page and the visit to purchase. Additionally he/she may carry on viewing pages on the site in the same visit as the one in which he/she visited the 'offers/gifts.htm' page. Host 3 satisfies the negation term, (3), in the template and therefore his/her navigation does not match the template.

The given example makes use of the URL document name only, and thus the field name does not have to be specified. However, it is also possible to handle more than one field in navigation templates. For instance, when tracking visitors who have come from a specific site, say a popular search engine at which a banner ad has been placed, the template and the discovery algorithm must distinguish between URL names and http referrers. M*i*DAS supports both mechanisms: the formal specification of navigation templates, which include field names, is omitted for simplicity.

2.2.3 Network Topologies

The second type of taxonomical domain knowledge is that of *network* structures, which is useful when the topology of web site or only a sub-network of a large site has to be used for the discovery of knowledge. Within M*i*DAS it is used to include or exclude certain parts of a web site, as shown in Fig. 2.

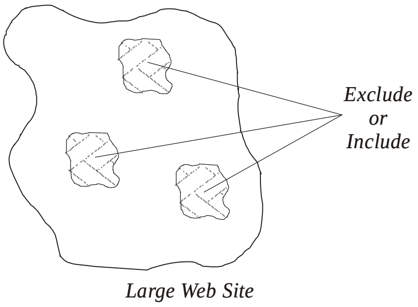
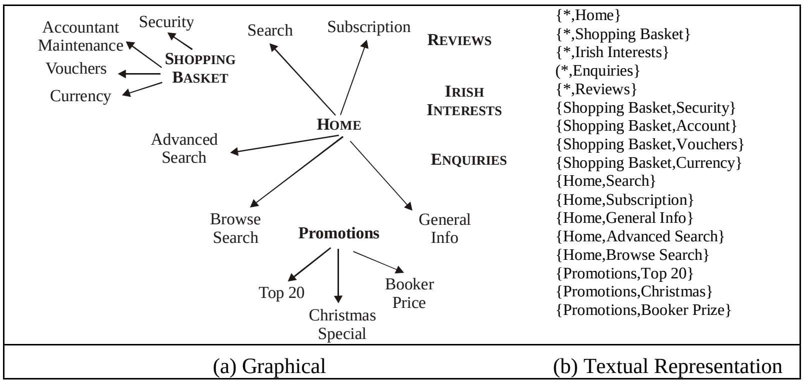


Fig. 2. Usage of Network Domain Knowledge

Definition 3. A network $w(N, E)$ is a directed, connected, cyclic graph, which is defined by a set of nodes $N = \{n_1, n_2, n_3, \ldots\}$ and connecting edges $E = \{e_1, e_2, e_3, \ldots\}$. Each node n_i represents a web page and $e_{ij} = <n_i, n_j>$ represents the existence of a hyperlink from page n_i to n_j .

In general, a network can be represented as a set of navigational patterns. The reason for distinguishing these two types of domain knowledge is that navigational templates are goal dependent and may change with each run of M*i*DAS. A network on the other hand is based on the structure of the web site and so is less likely to change with the same frequency. An example network of a bookstore is shown graphically in Fig. 3(a), where words in small caps describe pages that can be reached from any other page on the site. The textual counterpart is depicted in Fig. 3(b), where an asterisk denotes the set of all pages.

In addition to creating network topologies manually, which is only feasible in relatively small sites, two methods exists for creating the structure automatically. The first uses spider technology, which allows the discovery of all pages in an entire site, while the second derives the topology from log files themselves, based on all site internal http referrers – URL document name links [6]. The drawback of this approach is that only those links and pages are found which have been navigated through by at least a user-specified number of visitors.



{*,Home}
{*,Shopping Basket}
{*,Irish Interests}
(*,Enquiries}
{*,Reviews}
{Shopping Basket,Security}
{Shopping Basket,Account}
{Shopping Basket,Vouchers}
{Shopping Basket,Currency}
{Home,Search}
{Home,Subscription}
{Home,General Info}
{Home,Advanced Search}
{Home,Browse Search}
{Promotions,Top 20}
{Promotions,Christmas}
{Promotions,Booker Prize}

(a) Graphical (b) Textual Representation

Fig. 3. Example Network Topology

The advantage of this mechanism is that high hit pages, links and areas, as well as strong intra-topology connections can be found, which allows the discovery of sub-networks, as depicted in Fig. 2. Thus, we favour the log file based approach, which has proven sufficient in data mining exercises (see also case study in Section 4).

2.2.4 Concept Hierarchies

The third type of taxonomical domain knowledge that is supported is *concept hierarchies* [11].

Definition 4. A concept hierarchy is a tree with each level of non-leaf nodes, l_i, representing a generalised concept of the previous level, l_{i-1}. The leaf nodes, represented in level l_0 represent individual values of the attribute appearing within the log file. Nodes in level l_i are referred to as the parents of the nodes that they are connected to in level l_{i-1}.

In addition to marketing-related hierarchies, such as product categorisations or customer locations, a typical application is the topological organisation of Internet domain levels [6]. An example concept hierarchy is depicted in Fig. 4.

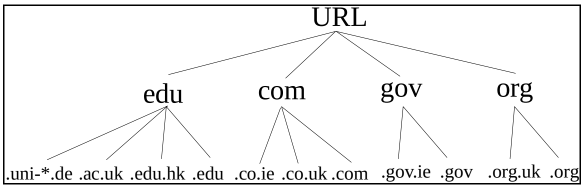


Fig. 4. Example Concept Hierarchy

The usage of multi-level concept hierarchies within M*i*DAS is based on concept levels, absolute or relative threshold values. These methods are described below.

- Concept Levels

  This method groups all nodes of the tree to a level greater than the user-specified cut-off threshold for their occurrence within the data.

- Absolute Threshold

  The absolute threshold is used to group items to their parent node. If the occurrence of an item is less then the threshold, only then the item will be grouped to its parent node.

- Relative Threshold

  The relative threshold is used to group items to its parent node. If the relative occurrence of an item is less then the threshold, then the item will be grouped to their parent node. Relative occurrence is calculated as follows.

$$relative\ \ occurence = \frac{occurrence\ \ of\ \ the\ \ current\ \ item * 100}{\sum occurrence\ \ of\ \ all\ \ items\ \ at\ \ the\ \ same\ \ level}$$

3 Navigation Pattern Discovery

This section describes the three stages of the M*i*DAS algorithm and defines the different navigation types that can be discovered, which are represented as contained in relationships.

3.1 Problem Statement and Notation

Given a log file that represents customer interactions on a web site, the objective is to discover navigation patterns in the form of sequences, using user-defined domain knowledge.

The log file L (as specified in Definition 1) is partitioned into P_1, P_2, P_3,…, where each partition can be uniquely identified by the primary key (Host, in example log file in Table 1). A partition is converted into a sequence S, using the secondary key (Date/Time, in example log file in Table 1) as a pivot. For instance, Host 1 in Table 1 would look as follows after the transformation:

```
<ecom.infm.ulst.ac.uk | / | /products | /products-
emw , kdnuggets.com/sift/t-textweb.html | /product-
emw | /products | /products-capri>
```

A navigational pattern is treated as a sequence and thus the two terms are used interchangeably. A sequence is defined as follows.

Definition 5. A sequence $S = \langle s_1, s_2, s_3,\ldots\rangle$, where each s_i represents a non-empty sub-sequence $\langle h_1\ h_2\ h_2,\ \ldots\rangle$, each h_i being an hit and each sub-sequence being a session/visit. A sequence of n sub-sequences is called an ***n*-*sequence***, while a sub-sequence consisting of ***m*** hits is called an ***m*-*sub-sequence***.

```
MiDAS ß Sorted log file $L$, thresholds $\tau(\sigma, \delta, \lambda^-, \lambda^+, \gamma^-, \gamma^+)$, domain knowledge $K(T, W, c)$

A priori
    $M$ = {all 1-sequences}                                  // Input data preparation
    Map $c$ onto $M$                                         // Concept Hierarchies
    $L_T = \{l_1, l_2, l_3 \ldots\}$                         // Data transformation (log file)

Discovery
    Foreach $l_i$ in $L_T$ do                                // Build pattern tree $P$ for each 1-sequence
            Foreach 1-sequence in $M$ do
                    $P_{1\text{-sequence}}$ := Update($l_i$, $P_{1\text{-sequence}}$) // Increase frequency counter if hit
                                                             $h \in l_i$ exists, add $h$ to $P$ otherwise
            End
    End
    Foreach $P_i$
            Read all n-sequences, with $\sigma$ from $P_i$ and append them to answer set $U$
    End

A posteriori
    Filter out all sequences in $U$ where $u_i \in T$ and $u_i \in$ W) // Navigation Templates &
                                                             Network Topology
    Delete all sequences in $U$ which are not maximal and satisfy $\delta, \lambda^-, \lambda^+, \gamma^-, \gamma^+$   // Pruning

MiDAS à $U$
```

Fig. 5. The M*i*DAS Algorithm

3.2 The M*i*DAS Algorithm

The M*i*DAS algorithm consists of three major phases, which are described in the following sub-sections. The algorithm itself is shown in Fig. 5.

3.2.1 A Priori Phase

The first step of the *a priori* phase is the input data preparation, which consists of data reduction and data type substitution. The former counts the number of all item occurrences (hits) in L for each of the individual web pages and excludes the hits, which have a support less than σ. The latter replaces all hits in L with a hit identifier h_i. Let M be the set of all hits. Each h_i Î M represents a unique hit and its frequency. Each h_i is also the basis for the pattern tree construction phase, discussed later. Concept hierarchies defined on the web pages are used during data reduction using either concept level generalisation, absolute threshold or relative threshold generalisations. This further reduces the number of unique hits (see [11] for generation and refinement of concept hierarchies).

In the data transformation phase a new database L_T is created that includes only the hit identifiers for the values that are included in M. The database includes the primary and secondary key, as provided by the original log file. The transformed hits in L_T do not contain any field names, since this information is represented through hit identifiers, which is shown in the Table below.

Host	Date / Time	Referrer	Hit	Hit	Hit
1	01/06/99 16:48:27	1	4	6	5
1	12/06/99 14:08:43	2	5	6	8
2	24/05/99 06:34:24	3	4	7	
3	03/06/99 12:14:20	2	5	6	8
3	03/06/99 15:47:03	3	4	7	

Table 2 Example Log Data

3.2.2 Discovery Phase

The pattern tree is the core element of M*i*DAS used to discover sequences of hits. Simplified, the pattern tree is a directed, acyclic graph, where a node contains the properties of a hit and the arcs represent the relationship between two nodes. The depth *d* of a node also represents the position of a hit in an *n*-sequence. There exist two different link types for describing the relationships between two nodes. *Sequence arcs* connect two nodes that go across sub-sequences (multiple visits on a web site), and *tuple arcs*, which connect two nodes that are in the same sub-sequence (same visit). An example abstract pattern tree is shown in Fig. 6.

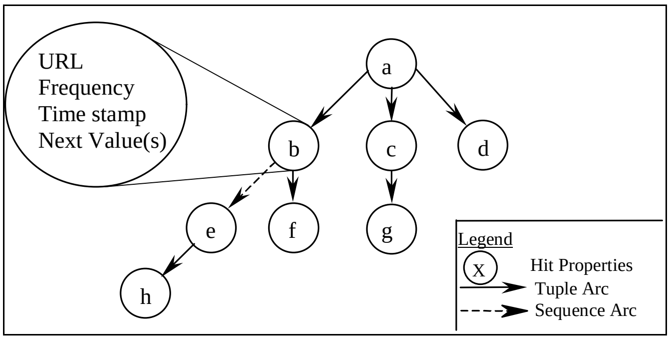


Fig. 6. Abstract Pattern Tree

As shown in Fig. 5 a pattern tree *P* is created for each 1-sequence $\in M$. Since the root node is always the first item in a sequence, which is stored in the current tree, it is used as anchor. For every hit $\in l_i$ that occurs after the anchor a hit node is created and then linked to their parent node or updated if already in existence. Due to the fact that M*i*DAS needs one pass over the database for creating each new tree level, the above process will loop until no new hit is found in each $l_i \in L$. Finally, a set of sequences *U* is created that satisfies the syntactic constraints defined in Section 2.2.

3.2.3 A Posteriori Phase

The first step in the *a posteriori* phase is to filter out all sequences that do not fulfil the criteria laid out in the specified navigation templates *T* and the topology network *W*.

The pruning phase is the last stage of the M*i*DAS algorithm. It removes all sequences that are not *maximal*.

Definition 6. In a set of sequences Q, a sequence Q_i is *maximal* if Q_i is not *contained* in any other sequence Q_j, that is $Q_i \not\mathbf{p}\ Q_j$.

M*i*DAS provides three different methods to decide when a sequence Q_i is contained in another sequence Q_j. It produces different kinds of result sequences, which can be *associative*, *partial* or *full*.

To describe the different pruning methods, a set of input sequences $U = \{Q_1, Q_2, Q_3, \ldots\}$ is defined. Each Q_i is of the form $< H_1^i\ H_2^i\ H_3^i\ \ldots >$, where each H_j^i is a sub-sequence and declared as $H = < h_{j_1}^i\ h_{j_2}^i\ h_{j_3}^i\ \ldots >$, each element being a hit. If a sequence is contained in another sequence, then it is not maximal and will be removed.

Associative sequences represent patterns, which have maximal length, independent of their time ordering. These represent visited page sequences of customers during relatively long stays where the aim is to discover pages that are frequently hit in the same session. Clearly the time ordering of the hits is not important in this context. Similar methods have been applied in the context of web data in order to discover path traversal patterns [8].

Definition 7. A sequence Q_i is associatively contained in another sequence Q_j ($Q_i\ \mathbf{p}_\mathrm{a}\ Q_j$) iff $\bigcup_k H_k^i \subset \bigcup_k H_k^j$.

For example, <(d)(a)> $\mathbf{p}_\mathrm{a}$ <(a)(c d)(h f i)(b)>, since (d) $\subseteq$ (c d) and (a) $\subseteq$ (a). However, ⟨(e)(d)⟩ $\not\mathbf{p}_\mathrm{a}$ ⟨(a)(c d)(h f i)(b)⟩ because (e) $\not\subset$ (a), (e) $\not\subset$ (c d), (e) $\not\subseteq$ (h f i), and (e) $\not\subseteq$ (b).

Partial sequences are similar to their associative counterparts, but take into account the time ordering of individual sessions attributed to a browser (user of the web site). The intuitive motivation for these types of sequences is that often we are interested in discovering as to how the browsers navigation behaviour is changing over time. Partial sequences provide us within this knowledge as they utilise the time ordering across sessions. The “partially contained in” relationship is identical to the “contained in” relationship proposed by [1].

Definition 8. A sequence Q_i is set to be partially contained in another sequence Q_j, ($Q_i\ \mathbf{p}_\mathrm{p}\ Q_j$) iff $\forall k \exists H_v^j \ni H_k^i \subset H_v^j \wedge k < v \wedge H_l^i \not\subset H_v^j \quad (\forall l = 1, \mathbf{K}\, k-1)$.

For instance, <(a)(h i)(b)> $\mathbf{p}_\mathrm{p}$ <(a)(c d)(h f i)(b)>, since (a) $\subseteq$ (a), (h i) $\subseteq$ (h f i) and (b) $\subseteq$ (b). However, <(a h)(i)(b)> $\not\mathbf{p}_\mathrm{p}$ <(a)(c d)(h f i)(b)> because (a h) $\not\subseteq$ (a), (a h) $\not\subseteq$ (c d), (a h) $\not\subseteq$ (h f i) and (a h) $\not\subseteq$ (b). Generally, this means that the sequence ⟨(x)(y)⟩ $\not\mathbf{p}_\mathrm{p}$ ⟨(x y)⟩ and vice versa.

The difference between partial and *full sequences* is that in the latter, the time-ordering of hits within a session is also considered and missing ‘hits’ (pages which have not been visited, hence skipped) are considered in the discovery of sequences.

Definition 9. A sequence Q_i is set to be fully contained in another sequence Q_j ($Q_i\ \mathbf{p}_\mathrm{f}\ Q_j$) iff $\exists k, p \ni H_k^i \subset H_p^j \wedge \exists r, s \ni h_{(r+v)k}^i = h_{(s+v)p}^j \wedge H_{k+g}^i \approx H_{p+d}^j$.

The symbol ≈ denotes the notion of equivalent sub-sequences, which is a special case of the strict containment relationship. H_k^i is said to be strictly contained in H_l^j ($H_k^i \mathbf{p}_s\ H_l^j$) iff $\forall r, s \ni h^i_{(r+v)k} = h^j_{(s+v)l}$. If $r = 1$ and $s = 1$ we say that H_k^i and H_l^j are equivalent sub-sequences ($H_k^i \approx H_l^j$) making it a special case of the strict containment relationship.

For example, <(d)(h f i)(b)> $\mathbf{p}_f$ <(a)(c d)(h f i)(b)>, since (d) ⊆ (c d), (h f i) ⊆ (h f i) and (b) ⊆ (b). However, <(c)(h f i)(b)> $\not\mathbf{p}_f$ <(a)(c d)(h f i)(b)> because d follows c in (c d).

4 Case Study

In this section a case study is presented, which exemplifies the usage of the M*i*DAS algorithm in one particular type of web mining goal.

4.1 Objective of Study

The overall objective of the study carried out was to discover interesting navigational behaviour from the log file of the MINEit Software Ltd web site (*www.MINEit.com*).

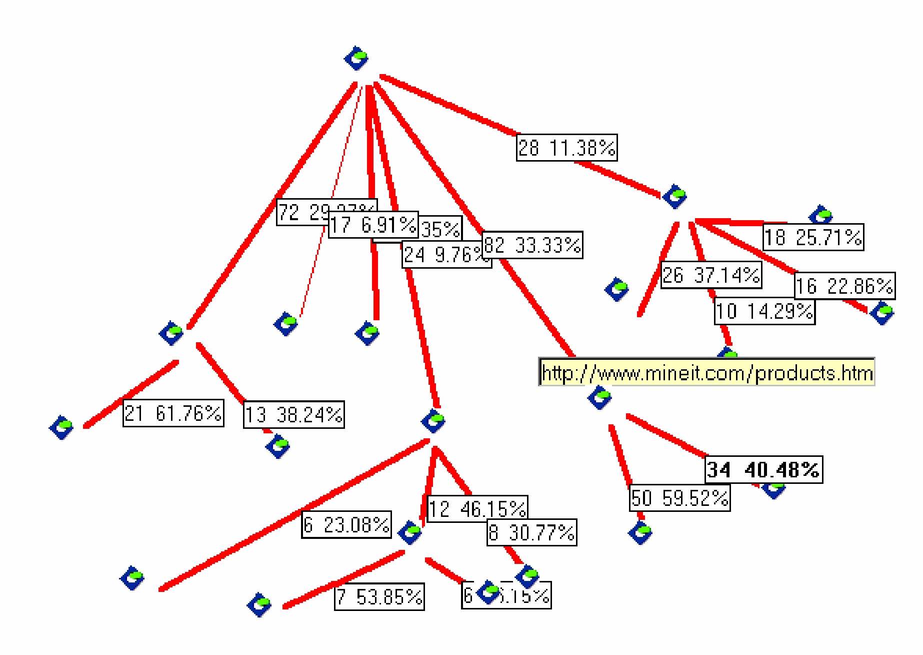


Fig. 7. Topology of analysed web site

More specifically, the goal of the web mining was to discover the behaviour of visitors coming from the *www.kdnuggets.com* site (where two new product entries and an entry on MINEit were placed prior to the analysis) against visitors from elsewhere. The site topology is shown in Fig. 7, where each node represents a page on the web site, and each is allotted the number of visitors as well as its percentage of all hits.

Using M*i*DAS (as any other traditional sequential pattern discoverer) without domain knowledge resulted in non-actionable results. Either the sequences were very specific (very high support), or a multitude of navigational patterns was returned (very low support). Thus, it was decided to incorporate domain knowledge that is tailored towards the topology seen above and the sub-tree for products in particular.

4.2 Domain Knowledge Incorporation

In order to get meaningful results a concept hierarchy shown in Fig. 8 is introduced. The hierarchy consists of two main branches; first representing the visitors who have come from www.kdnuggets.com, second representing all other users. The branches are used to discover different behaviour from targeted users versus non-targeted users.

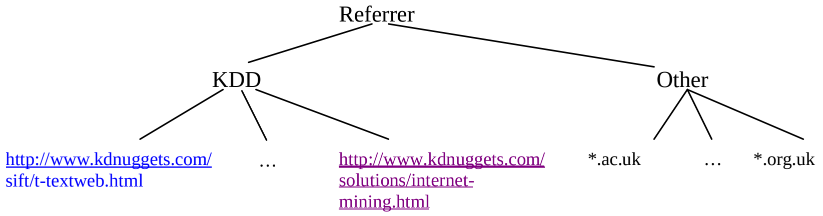

Referrer
KDD
Other
http://www.kdnuggets.com/sift/t-textweb.html
…
http://www.kdnuggets.com/solutions/internet-mining.html
*.ac.uk
…
*.org.uk

Fig. 8. Referrer Concept Hierarchy

In order to filter out all other unwanted visitors, the following navigation templates have been specified, which are depicted in Fig. 9 and Fig. 10, respectively.

```
^[<Referrer = kdd>;<*>; < Referrer = mineit>;<*>]
^[< Referrer = kdd>;<*>; < Referrer = kdd>;<*>]
[<Referrer = kdd>;<*>;<DocName = /products-emw.htm>]
[<Referrer = kdd>;<*>;<DocName = /products-capri.htm>]
[<Referrer = kdd>;<DocName = /products.htm>;<DocName = /products-emw.htm>]
[<Referrer = kdd>;<DocName = /products.htm>;<DocName = /products-capri.htm>]
```

Fig. 9. Targeted Visitors

```
^[< Referrer = other>;<*>; < Referrer = mineit>;<*>]
^[< Referrer = other>;<*>; < Referrer = other>;<*>]
[<Referrer = other>;<*>;<DocName = /products-emw.htm>]
[<Referrer = other>;<*>;<DocName = /products-capri.htm>]
[<Referrer = other>;<DocName = /products.htm>;<DocName = /products-emw.htm>]
[<Referrer = other>;<DocName = /products.htm>;<DocName = /products-capri.htm>]
```

Fig. 10. Un-targeted Visitors

Based on the above specified domain knowledge, as well as task-specific syntactic constraints in the form of M*i*DAS parameters, it is now possible to discover more user-driven navigation patterns.

4.3 Knowledge Discovery and Interpretation of Results

As indicated above, two separate runs were performed using M*i*DAS, one to discover navigational behaviour of targeted visitors and one to detect behaviour of their un-targeted counterpart. Some interesting sequences that were found are shown below, where the two numbers connate support and confidence, respectively.

```
(5.60%, 16.47%) Referrer=kdd | DocName=/products-capri.htm
(22.53%, 66.27%) Referrer=kdd | DocName=/products-emw.htm
(0.93%, 15.91%) Referrer=kdd | DocName=/ , DocName=/products-capri.htm
(2.40%, 40.91%) Referrer=kdd | DocName=/ , DocName=/products-emw.htm


(1.60%, 6.22%) Referrer=other | DocName=/products-emw.htm
(2.27%, 10.76%) Referrer=other | DocName=/ , DocName=/products-emw.htm
(4.27%, 20.25%) Referrer=other | DocName=/ , DocName=/products-capri.htm
```

The first set of chosen sequences shows that visitors who came from the kdnuggets site, either went directly to one of the two products or came to the home page and went to the product pages at a later stage. The second shows the counterpart of all other visitors. The result can be tabularised as follows, where the first value indicates support and the second confidence in percent.

Referrer	Easyminer Direct		Easyminer *via* Home page		Capri Direct		Capri *via* Home page	
KDD	22.53	66.27	2.40	40.91	5.60	16.47	3.93	15.91
Other	1.60	6.22	2.27	10.76	< 1.20	n/a	4.27	20.25

Table 3. Result Summary

The table shows that of all visitors who have come from kdnuggets, 22.5% went straight to the Easyminer home page, whereas only 2.4% came through the MINEit home page to get to the same URL. People from all other referrers with the same destinations only cover 1.6% and 2.3%, respectively. Similar behaviour was found for the Capri page, where 5.6% from kdnuggets went directly to the URL; no sequences were discovered for the given support threshold of 1.2%.

5 Evaluation of Relative Performance

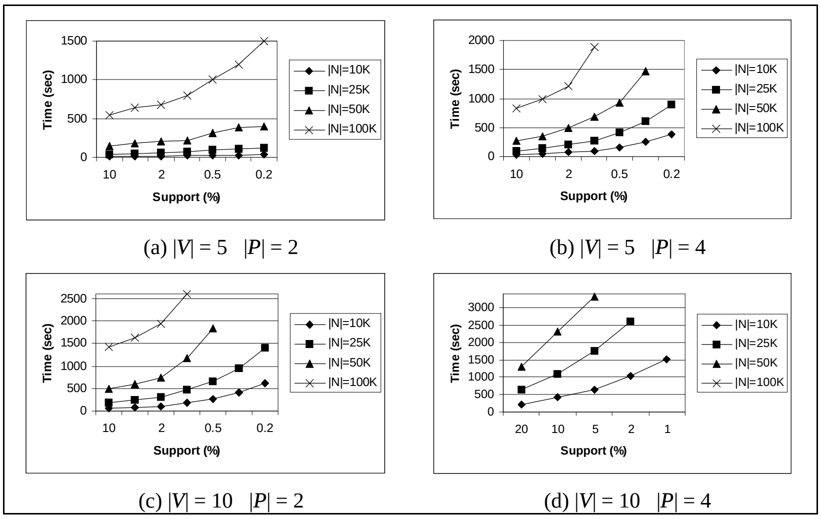
To assess the relative performance of M*i*DAS and study its scale-up properties a range of synthetic data sets were created. The parameters for the data generation program are shown in Table 4.

Parameter	Description
|*N*|	Number of visitors
|*V*|	Average number of visits per visitor
|*P*|	Average number of pages per visit

Table 4. Parameters for Synthetic Data Generation

A permutation of the three parameters has been performed such that |*N*| has taken on the values 10K, 25K, 50K and 100K, |*V*| 5 and 10, and |*P*| 2 and 4, which has led to 16 data sets. The number of visits is picked from a Poisson distribution with mean μ = |*V*|, and the number of pages is selected from a Poisson distribution with mean μ = |*P*|. The number of pages per site has been set to 100, the average length of potentially large navigation to 5.

Fig. 11 shows the execution times of the M*i*DAS algorithm for the generated datasets as the minimum support is decreased from 10% down to 0.2%, except for Fig. 11(d) where the minimum support is decreased from 20% down to 1%. As expected, the performance decreases with lower minimum support. The increasing number of visitors shows the scale-up properties of M*i*DAS, which have shown similar behaviour to existing sequential data mining algorithms without web-specific functionality.



(a) |*V*| = 5 |*P*| = 2

(b) |*V*| = 5 |*P*| = 4

(c) |*V*| = 10 |*P*| = 2

(d) |*V*| = 10 |*P*| = 4

Fig. 11. M*i*DAS Execution Times

The qualitative evaluation has shown that the three types of different pruning methods provide a valuable filtering mechanism in different web mining exercises. The size of the result space can be controlled through the pruning type, where associative pruning produces the least and full pruning the most navigational patterns. The run-time behaviour of M*i*DAS has performed as expected. Because the algorithm requires one pass over the database for creating each tree level (see Section 3.2.2), the execution times depends on the maximum length of a sequence in the log file.

For the purpose of evaluating the relative performance of M*i*DAS, it would have been interesting to distinguish between the run-time behaviour with and without taxonomical domain knowledge. Due to the fact that the domain knowledge incorporation is dealt with at the *a priori* and *a posteriori* stages, respectively, scale-up comparison would not be feasible. That is, a higher degree of domain knowledge incorporation results in more expensive pre- and post-processing, which reduces the discovery phase, and vice versa. An appropriate objective evaluation can be carried out, when domain knowledge is considered in the discovery stage itself (see further work paragraph in Section 7).

6 Related Work

Efforts in the area of discovering sequential marketing intelligence from Internet log files can be sub-divided into two sub-areas. The first tackles the problem of discovering generic sequential patterns, while the second is concerned with the challenge of applying data mining techniques on Internet server data.

6.1 Sequential Patterns

The authors' research has mainly been influenced by that of Agrawal & Srikant [1] and their own extensions [19]. Sequential patterns are discovered, which can be constrained by a number of factors, such as support, minimum and maximum DateTime gaps (between user sessions), sliding DateTime windows (for user session merging) and concept hierarchies. Most of these constructs have been adopted and extended according to electronic commerce requirements. Neither navigational templates nor web topologies can be used in their algorithm, nor is field dependence supported, both features have proven useful in Internet environments. The host of proposed *a priori* and GSP algorithms have difficulties dealing with very large sequence length, which has been resolved in M*i*DAS using depth-dependent pattern trees. Furthermore, the given "contained in" relationship has been proven too limited for some electronic commerce data analyses.

Zaki [21] has parallelised the GSP algorithm using a lattice-based approach and equivalence classes. Although the proposed algorithm has improved the performance of the serial version, it still carries the drawbacks described above.

Somewhat related to sequence discovery is work carried out by Manilla & Toivonen [13, 14, 15] in the field of frequent episode discovery, which can be located between sequential and temporal patterns. An episode contains a set of events and an associated partial order, which can be seen as the equivalent to a sequence. However, their work is concerned with the discovery of frequent episodes in a single event sequence, while our work concentrates on the discovery of sequences across many different online customer sequences.

6.2 Mining Web logs

Žaïane *et al.* [20] have applied various traditional data mining techniques to Internet log files in order to find different types of patterns, which can be harnessed as electronic commerce decision support knowledge. The process involves a data cleansing and filtering stage (manipulation of date and time related fields, removal of futile entries, etc.) which is followed by a transformation step that reorganises log entries supported by meta data. The pre-processed data is then loaded into a data warehouse, which has an *n*-dimensional web log cube as a foundation. From this cube, various standard OLAP techniques are applied, such as drill-down, roll-up, slicing, and dicing. Additionally, artificial intelligence and statistically-based data mining techniques are applied on the collected data which include characterisation, discrimination, association, regression, classification, and sequential patterns. The overall system is similar to ours in that it follows the same process. However, the approach is limited in several ways. Firstly, it only supports one data source — static log files —, which has proven insufficient for real-world electronic commerce exploitation. Secondly, no domain knowledge (marketing expertise) has been incorporated in the web mining exercise, which we see as an essential feature. And lastly, the approach is very data mining-biased, in that it re-uses existing techniques that have not been tailored towards electronic commerce purposes.

Cooley *et al.* [9] have built a similar, but more powerful architecture. It includes intelligent cleansing (outlier elimination and removal of irrelevant values) and pre-processing (user and session identification, path completion, reverse DNA lookups, et cetera) for Internet log files, as well as the creation of data warehousing-like views [10]. In addition to Žaïane's [20] approach, registration data, as well as transaction information is integrated in the materialised view. From this view, various data mining techniques can be applied; including path analysis, associations, sequences, clustering and classification. These patterns can then be analysed using OLAP tools, visualisation mechanisms or knowledge engineering techniques. Although more electronic commerce-orientated, the approach shares some obstacles of [20]'s endeavour, mainly in the non-incorporation of marketing expertise.

Bhowmick *at al.* [5] have developed a web data warehouse (called WHOWEDA), which is based on their own web data model. From within that environment, various web mining activities can be performed, which are all based on traditional data mining mechanisms and which do not provide any support for domain knowledge incorporation.

Spiliopoulou [17] has developed a sequence discoverer for web data, which is similar to our M*i*DAS algorithm. Their GSM algorithm uses aggregated trees, which are generated from log files, in order to discover user-driven navigation patterns. The mechanism has been incorporated in an SQL-like query language (called MINT), which together form the key components of the Web Utilisation Analysis platform [18].

Borges & Levene [6] have also developed an algorithm to discover user navigation patterns. Their mechanism is based on hypertext probabilistic grammars, which is a subclass of probabilistic regular grammars and uses an entropy measure as an estimator of the statistical properties of each link.

7 Conclusions and Future Work

A new algorithm for discovering sequential patterns from web log files has been proposed that provides behavioural marketing intelligence for electronic commerce scenarios. New domain knowledge types in the form of navigational templates and web topologies have been incorporated, as well as syntactic constraints and concept hierarchies. Multi-dimensional data can be used as input, which allows the representation of hits from multiple attributes. Three different types of "contained in" relationships are supported, which leaves room for typical navigational browsing behaviour on the Internet, such as skipping pages or bookmarking pages for later usage. Also, hit duplicates can be handled, as they reflect browser refresh/reload operations. Finally, all newly proposed mechanisms have been applied in a large-scale electronic commerce data mining project [4] and performance tests on synthetically generated data have shown promising results.

Further work in the area of discovering marketing-driven navigation patterns is twofold. First concentrates on practical issues, which include horizontal and vertical diversification of digital behavioural data (such as Web TV, Internet channels, or wireless mobile devices) and a smoother interface to a web-enabled data warehouse. Second is concerned with the improvement of the algorithmic constituent. In order to leverage the knowledge in concept hierarchies and navigation templates for providing business intelligence, domain knowledge will be incorporated into the discovery phase.

References

1. Agrawal, R., Srikant, R.: Mining Sequential Patterns. Proc. Int'l Conf. on Data Engineering (1995) 3-14
2. Anand, S.S., Bell, D.A., Hughes, J.G.: The Role of Domain Knowledge in Data Mining. Proc. 4th Int'l ACM Conf. on Information and Knowledge Management (1995) 37-43
3. Anand, S.S., Büchner, A.G.: Decision Support using Data Mining. FT Pitman Publishers (1998)
4. Anand, S.S., Büchner, A.G., Mulvenna, M.D., Hughes, J.G.: Discovering Internet Marketing Intelligence through Web Log Mining. Unicom'99
5. Bhowmick, S.S., Madria, S.K., Ng, W.-K., Lim E.P.: Web Mining in WHOWEDA. Some Issues, Proc. PRICAI98 Workshop on Knowledge Discovery and Data Mining (1998)
6. Borges, J., Levene, M.: Data Mining of User Navigation Patterns. Proc. WEBKDD99 Workshop on Web Usage Analysis and User Profiling (1999) 31-36 (same volume)
7. Büchner, A.G., Mulvenna, M.D.: Discovering Internet Marketing Intelligence through Online Analytical Web Usage Mining. ACM SIGMOD Record, **27**:4 (1998) 54-61
8. Chen, M.S., Park, J.S., Yu, P.S.: Data Mining for Path Traversal Patterns in a Web Environment. Proc. 16th Int'l Conf. on Distributed Computing Systems (1996) 385-392
9. Cooley, R., Mobasher, R., Srivastava, J.: Web Mining: Information and Pattern Discovery on the World Wide Web. Proc. 9th IEEE Int'l Conf. on Tools with Artificial Intelligence (1997)
10. Cooley, R., Mobasher, R., Srivastava, J.: Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information Systems 1:1 (1999)

11. Han, J., Fu, Y.: Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases. Proc. KDD'94 (1994) 157-168
12. Ling, C.X., Li, C.: Data Mining for Direct Marketing: Problems and Solutions. Proc. KDD’99 (1998) 73-79
13. Manilla, H., Toivonen, H., Inkeri, A.: Discovery of Frequent Episodes in Event Sequences. Proc. 2nd Int’l Conf. on Knowledge Discovery and Data Mining (1995) 210-215
14. Manilla, H., Toivonen, H.: Discovering generalized episodes using minimal occurrences. Proc. 2nd Int’l Conf. on Knowledge Discovery and Data Mining (1996) 146-151
15. Manilla, H., Toivonen, H., Inkeri, A.: Discovery of Frequent Episodes in Event Sequences. Data Mining and Knowledge Discovery, 1:3 (1997) 259-289
16. Mulvenna, M.D., Norwood, M.T., Büchner, A.G.: Data-driven Marketing. The Int'l Journal of Electronic Commerce and Business Media, 8:3 (1998) 32-35
17. Spiliopoulou, M.: The laborious way from data mining to web mining. Int’l Journal of Computing Systems, Science & Engineering, March (1999)
18. Spiliopoulou, M., Faulstich, L.C., Winkler, K.A.: A Data Miner analyzing the Navigational Behaviour of Web Users. Proc. ACAI'99 Workshop on Machine Learning in User Modelling (1999)
19. Srikant, R., Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. Proc. 5th Int'l Conf on Extending Database Technology (1996) 3-17
20. Žaïane, O.R, Xin, M., Han, J.: Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. Proc. Advances in Digital Libraries Conf. (1998) 19-29.
21. Zaki, M.J.: Efficient Enumeration of Frequent Sequences. 7th Int’l ACM Conf. on Information and Knowledge Management (1998) 68-75.