***Abstract*—In the rapidly evolving and growing environment of the internet, website owners aim to maximize interest for their website. In this article we propose a model, which combines the static structure of the internet with activity based data, to compute a website interest ranking. This ranking can be used to gain more insight into the flow of users over the internet, optimize the position of a website and improve strategic decisions and investments. The model consists of a static centrality based component and a dynamic activity based component. These components are used to create a Markov Model in order to compute a n-th order ranking.**

Keywords: web graph; website interest; centrality; user flow; Markov Model

I. Introduction

The internet is a rapidly growing and evolving environment, currently it contains approximately 20 billion indexed webpages[1] and only 24% of the world's population has access to it[2] . Users are entering the internet on a web site and use the available hyperlinks to travel to other pages and web sites. Simultaneously web site owners are constantly updating their existing web sites and creating new web sites. Over time web sites might also cease to exist (although others might have created cached copies). In short, users follow the structure created by web masters and others while this structure is constantly evolving.

Several theories have been developed to study the internet. One of these theories models the internet as a graph, called the web graph [1]. Webpages are represented as nodes in this graph and hyperlinks between pages as edges. The graph can both be directed, incorporating the direction of the hyperlink, or undirected. Since the internet is now modeled as a normal graph, all known graph theory can be applied to it. A restriction of the web graph to note is that page content is not incorporated in the graph model. The webgraph only contains connectivity information. Nevertheless, studies have proven that clusters in the webgraph usually are about the same topic. Therefore, discarding the actual content of the website and only looking at the graph structure, allows us to make claims about the content of the pages.

This web graph can be used to derive all kinds of interesting properties. Several techniques exist to determine an importance value for a node (website) in the graph. This is known as the centrality of a node. In a more social context, prestige can also be used as a measure of importance [2]. Identifying communities in this graph by local algorithms is also a well studied topic [3].

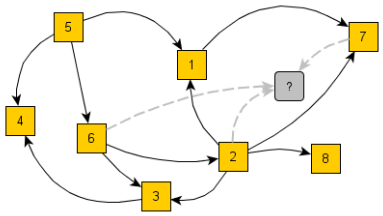
An important issue for website owners is: how to maximize the interest for their page. This can be seen as the problem to optimally position a website in the webgraph. See figure 1 for an example. The grey node (labeled as "?") is the website that has to be inserted as optimal as possible. The question now is, which links are required to achieve this goal?

Several website properties can be used to define what an optimal position for a website is:

[1]http://www.worldwidewebsize.com

[2]http://www.internetworldstats.com/stats.htm

Fig. 1. Example graph.



- *Content*
  The content of a website has to be of high quality for it's customers. If the content does not meet the customers requirements, they loose interest for the website. But, before content comes into play, customers have to reach the website first.
- *Links*
  Links from other pages to a website define the reachability of a website. A link from a website with high traffic will increase interest more than a link from a low traffic website.
- *Advertisement*
  The more people know about a website, the more people will be interested in it. Again the traffic of the potential candidate for advertisement is important to maximize the yield of the advertising.
- *Other*
  There are other aspects that might improve the interest for a website, e.g. search engine rankings, but those are not considered in this research project.

In this article we want to investigate how we can gain more insight into the static structure of the internet and the dynamic flow of users through this structure. This results in a flow potential score for a website. The improved insight, based on the flow potential score, can result in more strategic decisions and investments. Think about questions like "where should I advertise for my website". By placing adverts on external website, you are basically creating hyperlinks to your own website with the goal to attract visitors. With the flow potential score you can make a better choice about which website is best to advertise on in order to maximize the visitors to your website.

Flow potential, which is more than just flow if it also depends on properties of the underlying structure, will be referred to as website interest. The research question in this article is:

- How can website interest be measured based on static and dynamic properties?

In order to answer this question, the following sub questions have to be answered:

- What are the static and dynamic properties of websites?
- How can these properties be measured?
- How can these two types of properties be combined?

The rest of this article will try to answer these questions. In order to do so we will start with looking into interest and defining a model used for interest optimization in section two. This model consists of two components. One of these

components, centrality, will be discussed in the third section. The other component will be related to the discussion of existing work in section four. In section five an experiment is presented to test the proposed ideas. Section six will present the results of this experiment and in section seven we will conclude this article.

II. The model

In this section we will propose a model which can be used to optimize the interest for a webpage. Before we do this, we will first define what interest is, in this article.

A. Webpage Interest

Webpage interest is a very broad concept to describe which webpage [3] or website [4] is important for a user. In the context of this research we will restrict ourselves to websites and not at single pages. However, as it turns out, single pages have been used to research methods to optimize navigation through a webpage, based on (amongst other properties) the importance of the pages of the webpage. In later sections, this existing work will be discussed as well as the relation to this article.

Some research has already been conducted on the subject of measuring explicit and implicit user interest. This research mainly focussed on single pages, but this can easily be extended to sites. Two types of interest can be distinguished:

- Explicit: user specifies what he/she thinks about the webpage after being asked to do so.
- Implicit: infer what the user thinks about the webpage without asking it. Information used generally are things like: time spend on a page, the amount scrolled on a page, the amount of clicks on a page and/or any combination of these properties [4].

Both have disadvantages. Explicit interest requires the user to stop what he/she is doing and write down what he/she thinks about the page. This is very invasive. Implicit interest solves this problem, the user is not required to stop what he/she is doing. The main disadvantage of measuring implicit user interest is that this method requires a custom build browser or plugin which gathers the required information (clicks, scrolls, etc) on the client side.

In the context of our research we don't want to ask the user to specify his/her interest in a webpage. Most likely we won't have access to the webpage we want get information about. We also do not want to measure this interest on the client side. This would mean releasing our own browser and we would only be able to gather data from the user base using this browser. Therefore, both implicit and explicit interest as mentioned above, do not offer what we need.

We would like to use the webgraph, and therefore loose content information, to measure the user interest for a webpage. The two disadvantages mentioned above, are not relevant anymore when using the webgraph. By writing a spider application we can construct this graph (or a relevant part of it) ourselves, or we can buy this data from an external party specialized in this matter.

As a first approach we will use the page importance, calculated from the webgraph, as the measure for interest in a webpage. In the remainder of this paper, this measure will be extended and refined.

B. The Model

The model, proposed in this section, derives the webpage interest value $R(p)$ of a webpage p from the following two components. The first component is the webpage importance, $I(p)$, which is measured relative to other webpages. The second component $P(p)$ is a property that quantifies the interest in page p. These components are combined by a function called R_c:

$$R(p) = R_c(I(p), P(p))$$

The two components combine static and dynamic properties of webpages respectively. The importance function I is a static property of the (web) graph, and may be measured by centrality, which is a known concept from graph theory. In section three we will discuss the concept of centrality in depth and in a later section we will also see what types of centrality are suitable for applying to the proposed model.

The interest function P is a flexible, dynamic component. P depends on the domain of the problem we are looking at, but it should be an indication of the activity of the nodes. Link traversal counts, a count of how often users follow a specific link, would be the best activity based measure. It can measure incoming traffic and outgoing traffic and it can be used to compute the number of user entering and leaving a certain domain. Unfortunately this information is not publicly available. We will propose a solution to convert activity based data for single nodes into probabilities of traversing a link. This will be discussed moro thoroughly in section four, in relation to the existing work. Section four will be concluded with a solution for the relation between $I(p)$ and $P(p)$.

III. Centrality in a graph

In this section we will discuss the concept of centrality in a graph. As mentioned in the previous section, centrality is one of the two components of the proposed model. First, a short introduction into graph theory is given. All concepts for the later sections will be discussed. After this introduction the four well-known concepts of centrality will be explained. Then we will relate these concepts to the proposed model and finally we will explain a little about how to calculate theses centrality measures.

A. Basic Graph Theory

A (web) graph G is defined as an ordered pair $G = (V, A)$ where V is the set of vertices or nodes and A is the set of (directed) arcs between the nodes in the graph: $A \subseteq V^2$. We will also write $A(v, w)$ to denote arc $(v, w) \in A$. In the case of a webgraph, the nodes of the graph are the webpages and the arcs are the hyperlinks between them.

[3] webpage = a single page of one website.

[4] website = the collection of all (single) webpages of one website.

A graph can be either directed or undirected. In the case of a directed graph, the direction of the arc is important. Therefore $(v, w) \neq (w, v)$ holds. In the case of an undirected graph, the direction of the arc does not matter and $(v, w) = (w, v)$ holds, in this case we also speak of edges. In this article however, we will only use the term arc and assume the graph is directed unless we specify otherwise. If we look at the webgraph, a hyperlink indicates a directed link between two webpages. If one page links to another page, it does not necessarily mean this also holds the other way around.

Let $P(v, w)$ be the set of all paths between nodes v and w. The path between two nodes v and w through another node z can then be defined as $P(v, w, z,) = P(v, z) \circ P(z, w)$. We have introduced a new operator here: $\circ$. This operator is used to concatenate two sets of paths in such a way that the resulting set has every possible combination of paths. The number of paths between two nodes is equal to the number of paths in the set, $p(v, w) = |P(v, w)|$ and $p(v, w, z) = |P(v, w, z)|$.

A path itself is a concatenation of arcs. It we consider this an ordered set of arcs, the length of this set is the length of the path, $|p| \in P(v, w)$.

A geodesic is a shortest path between two nodes. In a later section we will need information about the geodesics between two nodes v and w. Therefore the function $G(v, w)$ is defined:

$$G(v, w) = \{p \in P(v, w) | \forall_{q \in P(v,w)} [|p| \leq |q|]\} \quad (1)$$

As with the path definitions, we will also include a definition for geodesics between two nodes v and w, passing through node z:

$$G(v, w, z) = G(v, z) \circ G(z, w) \quad (2)$$

If we are interested in the number of geodesics between two nodes, we can look at the size of the set with geodesics: $g(v, w) = |G(v, w)|$ and $g(v, w, z) = |G(v, w, z)|$.

Lemma III.1. *Let* $v, w, z \in V$ *then* $g(v, w, z) = g(v, z) * g(z, w)$

Proof: According to (1) $G(v, z)$ is the set with shortest paths from node v to node z. If a shorter path existed, that would be the set $G(v, z)$ because $|p| \leq |q|$ has to hold to qualify as geodesic. The same holds for $G(z, w)$. The total number of shortest paths is the number of shortest paths from v to z, $g(v, z)$, multiplied by the number of shortest paths between z and w, $g(z, w)$. Therefore $g(v, w, z) = g(v, z) * g(z, w)$ holds. ■

The distance [5] of two nodes v and w is the length of the geodesic between those two nodes. This is defined as follows:

$$d(v, w) = |p| \text{ with } p \in G(v, w) \quad (3)$$

And the distance of a geodesic between two nodes v and w, through node z:

$$d(v, w, z) = |p| \text{ with } p \in G(v, w, z) \quad (4)$$

Lemma III.2. *Let* $v, w, z \in V$ *then* $d(v, w, z) = d(v, z) + d(z, w)$

Proof: According to (3) $d(v, z)$ is the distance of the shortest path between node v and z. The same holds for $d(z, w)$. If there is a shorter path between v and z, that would have been in the set of geodesics. It doesn't matter which of the shortest paths is selected from the set. The total length of the shortest path is then the length of the two shortest sub paths, $d(v, z) + d(z, w)$. Therefore $d(v, w, z) = d(v, z) + d(z, w)$ holds ■

Another well known concept in graph theory is the adjacency matrix. The adjacency matrix represents the set A of arcs as a matrix where the non zero entries represent the existing arcs. So $A[v, w] \neq 0 \Leftrightarrow (v, w) \in A$. By extending this principle more information can be stored in the adjacency matrix. If arcs have non-zero weights assigned, then these weights may be stored in the adjacency matrix. Another option is that the adjacency matrix stores probabilities corresponding to the likelihood of folowing that arc when traversing the graph.

B. Centrality

A natural question in a webgraph is: "How important is some page?". To answer this question the concept of centrality has been introduced. Centrality is a measure to calculate the importance of a webpage in the webgraph. This importance measure may be used in the webpage interest model as the static component I. The main approaches to centrality are:

- (1) Degree centrality.
- (2) Betweenness centrality.
- (3) Closeness centrality.
- (4) Eigenvector centrality.

The first three have been discussed extensively in a article by Freeman [5]. Eigenvector centrality is discussed by Borgatti [6] and is based on the work of Bonacich [7]. Later on, the well known PageRank [8] algorithm has been been based on this centrality measure.

1) Degree Centrality: Degree centrality will look at the degree value of a node. The degree of a node is defined as the number of links for that specific node. The indegree, C_d^{in}, is the number of incoming links and the out degree, C_d^{out}, the number of outgoing links. Formally the degree centrality is defined as follows:

$$\begin{aligned} C_d^{in}(v) &= \sum_{w \in V} A(w, v) \\ C_d^{out}(v) &= \sum_{w \in V} A(v, w) \end{aligned}$$

In the matrix representation we have:

$$\begin{aligned} C_d^{in} &= \mathbf{1}^T A \\ C_d^{out} &= A\mathbf{1} \end{aligned}$$

where $\mathbf{1}$ is a (column) vector of all ones. This degree centrality depends on the size of the graph. The maximum value for indegree and outdegree is $n - 1$[6], where n is the number of

[5] distance and length are synonymous

[6] This is based on the assumption there are no point cycles.

nodes in the graph. To be able to use this measure to compare different graphs, the C_d formula needs to be normalized:

$$C^{in}_{dNorm} = \frac{1}{n-1}C^{in}_d$$
$$C^{out}_{dNorm} = \frac{1}{n-1}C^{out}_d$$

The degree centrality measures the potential of a point to be part of a flow in a graph. The higher the degree, the more connections to other points there are. This means it is more likely that this node is part of some flow in the graph. In the sections about eigenvector centrality we will see a definition of centrality which is the opposite of this definition.

2) Betweenness Centrality: Betweenness centrality,C_B, counts the number of geodesics (= shortest paths between two nodes) a specific node is part of. The probability that point z is part of a randomly selected geodesic linking v with w.

$$B(v,w,z) = \frac{g(v,w,z)}{g(v,w)}$$

Based on this probability we can compute the betweenness centrality:

$$C^{in}_b(z) = \sum_{v \neq w \in V-z} B(v,w,z)$$
$$C^{out}_b(z) = \sum_{v \neq w \in V-z} B(w,v,z)$$

Freeman [5] has shown the maximum value for This is normalized against the number of pairs of vertices that do not include z (see Freeman [5]):

$$\frac{(n-1)(n-2)}{2}$$

leading to:

$$C^{in}_{bNorm} = \frac{2}{(n-1)(n-2)}C^{in}_b$$
$$C^{out}_{bNorm}(z) = \frac{2}{(n-1)(n-2)}C^{out}_b$$

When using this formula, the betweenness centrality can be seen as the potential of a point to control the flow in the graph. A point with maximum betweenness centrality has maximum control over the flow in the graph, since all paths go through this node. As an example, think of a 4 point star with one point in the middle. The middle point has maximum control.

This betweenness centrality considers all shortest paths in a graph. For large graphs this is computationally not attractive. In order to optimize this computation, an alternative approach has been proposed by Everett and Borgatti in [9]. They have proposed to compute betweenness centrality in the ego graph rather than the entire graph. So if we want to compute the ego betweenness centrality of a node v, $C_{EB}(v)$, we have to focus on the ego v and extract the ego graph of v from the entire graph. Then we can compute the betweenness centrality of v in it's ego graph. Everett and Borgatti have shown there appears to be some relation between the ego betweenness centrality and the normal betweenness centrality, but they didn't define this relation yet. They have shown that it's likely for this relation to exists, with an experiment.

Ego networks have been formally defined by Freeman [10] as follows: A graph G is a k-star if the following holds: The graph $G = (V, E)$ where V is the set of n nodes (or vertices) and E is the set of e symmetrical edges linking pairs of points. If $n > 2$ and there are $n-1$ edges such that some point p^* is directly connected or adjacent to all others. A centered graph is then defined as any graph of n points with a k-star. Any ego network is structurally a centered graph.

Centered graph have some interesting properties:

- Centered graphs have two extremals. The minimal k-star has $k-1$ edges connecting p^* to the $k-1$ other points. The maximal k-star is the complete graph where all edges are present with $\begin{pmatrix} k \\ 2 \end{pmatrix} = \frac{1}{2}k(k-1)$ edges connecting each point to all of the others.
- Centered graphs are connected. There is a path from any point to all others.
- The longest geodesic linking any pair of points in a centered graph is ≤ 2. This is also called the diameter of the graph.
- Any geodesic has either a length of one or two. If a path has a length of one, the nodes involved are adjacent. If the length of the path is two, the points are linked by a point in the middle.

Considering the last property, computing the ego betweenness is fairly straightforward in the ego graph. Every pair of non-adjacent nodes must have a geodesic of length two, through p^*. Everett and Borgatti have described a method to compute the ego betweenness from the adjacency matrix of the the ego graph.

$$A_{EB} = A^2[1 - A]$$

where A is the adjacency of the ego graph and 1 is a matrix with only 1's with the same dimensions as A. The ego betweenness is the sum of the reciprocals of the non zero entries in A_{EB}.

An important thing to note is how $A^2[1-A]$ is computed. Each matrix position in A^2 is multiplied with the same matrix position in $1-A$. So $A^{EB}_{i,j} = A^2_{i,j} \times [1-A]_{i,j}$ where $0 \leq i \leq$ rows and $0 \leq j \leq$ columns

3) Closeness Centrality: According to Freeman both the degree centrality and the betweenness centrality are useful as and index for flow control potential. The last centrality measure discussed by Freeman is a bit different. Closeness centrality is based on the degree to which a point is close to all the other points in the graph and therefore is able to avoid the control potential of others.

The most simple formula for closeness centrality, was first presented by Sabidussi as the decentrality of a point z:

$$C_c(z)^{in} = \frac{1}{\sum_{v \in V} d(v,z)}$$
$$C_c(z)^{out} = \frac{1}{\sum_{v \in V} d(z,v)}$$

For this formula to work, the graph needs to be connected. Otherwise infinite paths are included. If we consider the webgraph, this need not be the case. Therefore additional

work is required to use this formula for the webgraph. The normalized form is:

$$C'_c(z)^{in} = \frac{n-1}{C_c(z)^{in}}$$

$$C'_c(z)^{out} = \frac{n-1}{C_c(z)^{out}}$$

4) Eigenvector Centrality: The last centrality measure is the eigenvector centrality. Based on the work of Katz [11] and Hubble [12], Bonacich eventually developed a new approach to degree centrality [7]. It is interesting to see that Bonacich definition is almost the opposite of Freemans definition of degree centrality.

Freeman claimed that having a high degree meant it would be very likely to be part of the flow in the graph. Bonacich does agree with this, but regards this as a (possible) negative feature. If having a high degree means being important, depends on the nodes you are connected to. To be more specifically, it depends on the degree of those nodes. Being connected to nodes with a low degree, makes you more powerful and being connected to nodes which are themselves connected to a lot of other nodes, makes you only a little bit more important.

An example might clarify this. Consider Bill and Fred. They each have five close friends (meaning they are directly connected to them in the graph representation). Bill's friends are isolated people, Fred's friends also have lot's of friends themselves. It is easy to imagine Bill has more influence on his friends than Fred on his. If Fred tells his friends something, they will also hear things from their other friends. Therefore they might not believe Fred. In Bill's case, his friends will probably believe what he tells them. Bill is more influential then Fred.

We define the centrality of a node v as $C_E(v)$, C_E is a vector in this case and $C_E(v)$ is the element in this vector at position v. If we combine this with the influence of the neighbors of v, we get the following formula:

$$C_E(v) = \frac{1}{\lambda} \sum_{w \in V \setminus v} A(v,w) \times C_E(w)$$

where λ is a constant and A is the adjacency matrix of the graph. If we define $C_E(w)$ as the vector of centralities $x = (w, z, \dots)$, the formula can be rewritten as follows:

$$\lambda x = Ax$$

and this is the eigenvector equation and therefore this centrality measure is called eigenvector centrality.

C. Back to the model

In the previous section we have seen an overview of methods to compute the importance of a node in a graph. The importance component $I(p)$ of the model $R(p) = R_c(I(p), P(p))$ can be instantiated with one of these methods. Which method is preferred depends on the interest property $P(p)$. If, for example, we are interested in webpage traffic as the property for our model, the betweenness centrality looks like a promising candidate to use for measuring importance. Nodes or webpages with a lot of potential to control the flow, meaning that a lot of other webpages are connected to them, are good candidates to connect to in order to get a good amount of traffic to your webpage. Another good option could be the eigenvector centrality. By using the eigenvector centrality of a webpage, you can compute it's influence. Connecting to webpages with a lot of influence might result in more traffic to your webpage.

At first it is a good option to choose the centrality measure for the $I(p)$ component based on the conceptual meaning. At the end of this article we will test our model and the choices we made on a real data set.

Another challenge in order to compute the $I(p)$ component is the size of the webgraph. The webgraph is an incredibly large graph and it is growing larger every day. It's very unlikely to claim we can maintain an up-to-date graph of the internet, and providing a solution for this problem is beyond the scope of this article. The graph we use will always be obsolete as soon as we have modeled it. Moreover, computing the importance measures on such a large graph, might require a lot of computational time. This obviously depends on the centrality measure, degree centrality is more easy to compute than for example betweenness centrality. Therefore it might be good to use other techniques to create subgraphs which are relevant for us, in order to decrease the total size of the graph used to calculate our model.

A good example is the addition to betweenness centrality about ego betweenness centrality. By focussing on the ego graph instead of the entire graph, the complexity to compute the betweenness decreases and is therefore more easy to compute.

It might also be a good option to just focus on a specific part of the total webgraph. Several techniques exists which show that by using the static graph structure, logically connected subgraphs can be extracted. In order to create such subgraphs, techniques like community identification could be used. A community is a related subset of the total webgraph. By "zooming" in on only a small interesting part of the total internet, the problem to solve is made easier from the beginning.

IV. Existing work

Several techniques are available to analyze the popularity of single pages inside a single website. Based on this measure and the history of the current user's navigation, a prediction is made about which pages are of most interest to this user. The technique and model proposed in this paper aims to do something similar, but on a larger scale, the scale of the internet or a relevant part of the internet. Community identification is an example of a technique to obtain a relevant subset of the webgraph.

Analyzing the webgraph and constructing rankings based on the static structure of the graph is pretty common now. On the other hand, looking at activity based criteria is also pretty common. Several measures exist to indicate the number of visitors to a website, the duration of their visits and so on. A measure which combines these properties is not so common.

In the past years several studies have been conducted in the direction of adaptive websites. The studies are trying to improve the static website structure by incorporating dynamic activity based information.

In this section, several of these existing techniques, focussed on single websites, will be discussed. In the next section we will propose our definitive model, based on some of the techniques discussed in this section.

We will now list, part of, the evolution in the field of adaptive websites. We will start with some very basic techniques which will evolve into more sophisticated solutions.

A. Website optimization and adaptation

In the article about adaptive websites [13], the authors propose a system which is able to increase the effectiveness of websites based on several actions: (1) promotion and demotion, (2) highlighting, (3) linking and (4) clustering.

1. Promotion and demotion is the process of placing links to pages of the website into some reserved space on the front page. This operation is restricted to this reserved space (usually a boxed area) mainly to ensure web masters the algorithm cannot rearrange the entire structure of the website. The algorithm can only create new links into the existing website structure, called promotion. And it can only remove links it has created itself, called demotion. In order to promote a page v, the page has to be popular but not very accessible. The popularity of a page v is defined as the number of page accesses, which are extracted from the web servers logs.

$$pop(v) = \text{\# page accesses}$$

The accessibility *acc* on the other hand is a measure for how far away a page v is to the *frontpage*.

$$acc(v) = \frac{1}{d(frontpage, v)^2}$$

where $d(frontpage, v)$ is the distance function as defined in (3).

Based on the popularity and the accessibility, the promotion score *pro* is defined as follows:

$$pro(v) = \frac{pop(v)}{acc(v)}$$

A page will be promoted to the reserved area on the front page if $pro(v) > pro(w)$ with w being a page already in the box, and v being a page not in the box and $pro(v) > \pi$, with π being a threshold value. By using a certain threshold value, a certain promotion score is needed in order to be promoted. This definition causes pages which have many visitors but are far away from the *frontpage* to score a high promotion score.

2. Highlighting is emphasizing certain links by altering the font, color or graphics. This is a lightweight operation and can be done on all pages. The set $L(v)$ is defined as the set of all pages, a page v links to. The links in this set L are ordered by their access value. Based on this ordering, the top $x\%$ links on page v are highlighted.

3. Linking is connecting two previously unconnected pages or disconnecting two previously connected pages. The idea is to connect pages which are highly correlated. Two pages are highly correlated when a users visit both of them frequently. Let $P(v)$ be the probability that page v will be visited. If page v and w are not already linked, if $\rho(P(v), P(w)) > \delta$ the pages should be linked. Where δ is a threshold value.

4. Clustering associates a collection of related pages and makes them accessible as a group on a newly created page. The authors propose a set of rules to group pages, which are not grouped already, based on their similarity in filename and correlation of user visiting path.

The disadvantage of the proposed approach is promotion and demotion is limited to certain boxed areas on a page. This limitation however, is mainly introduced because web masters are not considered to be ready for fully adaptive websites. An advantage of this approach is both creating, removing and highlighting links look promising. Unfortunately, no empirical evidence is available to indicate how this approach performs.

The authors in the next article, website optimization using page popularity, [14] propose a method to measure page popularity for offline web rearrangement, in order to create a more accessible and effective website. The idea is very similar to the basic concept of promotion and demotion in the previous discussed article. This approach however, rearranges the entire website structure and is not limited to certain boxed areas.

Absolute page accesses, *pop*(v), are proposed as a measure for page popularity. However, this can be a misleading measure. The closer to the home page a certain page is, the more absolute accesses it will get. The authors define three properties which should be taken into consideration. (1) The depth of the page v relative to the front page z, $d(z, v)$, (2) the number of pages of the same depth as the page being examined, $n(v)$ and (3) the number of references to this page from any other page of the website, $C_{in}(v)$.

All three of these parameters can be combined into a new measure, the relative page access $pop'(v)$, defined by the following formula:

$$pop'(v) = c_1 * d(z, v) + c_2 * \frac{n(v)}{C_{in}(v)} * pop(v)$$

where c_1 and c_2 are constants which depend on the structure of the webpage. What a proper value could be for these constants is not sure yet, in the articles case study $c_1 = 1$ and $c_2 = 1$ is used. Therefore

$$pop'(v) = d(z, v) + 3 * \frac{n(v)}{C_{in}(v)} * pop(v)$$

$d(z, v)$ is an indication of how special a page v is. The further away from the *frontpage*, the more special page v is. The other component, $\frac{n(v)}{C_{in}(v)}$, defines page v as special when there are many pages at the same depth (it is chosen out of a large collection) and v has little incoming links (more incoming links increase the probability a user reaches page v and therefore decreases how special it is).

Three other properties are of interest, $t(v)$ which is the time a user spends on a specific page v thus $T = \sum_{v \in V} t(v)$

is the total time a users have spend on the entire website. These two properties are used to discard cases where users spent very little or too much time on a webpage. A very short time indicates a user clicking on a link immediately after entering page v or going back to the previous page by using the browsers back functionality. A very long time is an indication of a user leaving the webpage v open, but doing something else. The third property q is a measure for the number of different pages a user visits in one session. These properties are used the measure the effectiveness of the website structure. The goal is to increase page accesses, $t(v)$ and T while decreasing q.

Based on these definitions an application has been written by the authors: SOALA. This application has been used to test their proposed algorithm: If the relative page popularity of a given page exceeds the relative page popularity of at least one ancestor, the two pages are swapped. This approach assumes all pages in the website are ordered as a tree.

However, there are some limitations to the algorithm. In some situations automatic changes in structure are not possible because of the logical structure of the data. Think of a lyrics website, where songs are ordered alphabetically, the links A to Z should not be changed.

The initial experimental results look promising. The test application has been tested on several ad hoc generated webpages with different structures. The improvements where good. Next a webpage has been developed and logs were gathered for 15 days. After measuring this base situation the webpage was optimized by executing the SOALA application and again logs where gathered for 15 days. The new structure showed an improvement of approximately 14% in absolut page accesses and an 11% in the average page time (the time a user spends on a page).

B. Website link structure evaluation and improvement based on user visiting patterns

The author's in this article about Website link structure evaluation [15], describe a method where the website link structure results in a user preference measure. This measure is based on user visiting patterns which are used to construct a weighted directed graph. This weighted graph can then be used to compute connectivity between pages. By optimizing this connectivity, by adding or removing hyperlinks, the website structure is improved.

Web servers can be configured to create web logs of users visiting the server. These web logs are divided in user visiting sessions, where information of one user in a certain time period is stored. Let $M(s, v) = 1$ if page v was visited in user session s and $M(s, v) = 0$ otherwise.

A row in this matrix M contains the request status for each page in a certain user visiting session. A column contains the page accesses of a certain page for all user visiting sessions. The associate degree, $H(v, w) \in [0, 1]$ between page v and page w is the number of sessions in which both v and w have been accessed (h') divided by the sessions in which only v has been accessed (h). This is defined as follows:

$$H(v,w) = \begin{cases} \frac{h'(v,w)}{h(v)} & \text{if } h(v) \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

where

$$h(v) = \sum_{s \in S} M(s,v)$$

$$h'(v,w) = \sum_{s \in S} M(s,v) * M(s,w)$$

The weighted directed graph, $G = (V, A, W)$, can now be constructed where V is the set of pages in the website, A is the set of links between the pages and W is the set of weights for each link. A weight *weight* for a link $A(v, w)$ is defined as follows:

$$\textit{weight}(v,w) = \frac{H(v,w)}{H'(v)}$$

where $H'(v, w)$ is the sum of all associate degrees for the outgoing links of v:

$$H'(v) = \sum_{v \to z} H(v,z)$$

The weight of a route l from node v to node w, is defined as $\textit{weight}_{route}(l) = \textit{weight}(v, z_0) \times \ldots \textit{weight}(z_n, w)$. Let L be the set of all routes between v and w, the connectivity from node v to node w is defined as

$$C(v,w) = \sum_{l \in L} \textit{weight}_{route}(l)$$

Based on the weighted graph and connectivity function, an average connectivity score, E, can be calculated. The function E is defined as follows:

$$E = \sum_{v \neq w} \frac{C(v,w)}{n(n-1)}$$

where $n(n-1)$ is the total number of page pairs in the website. The higher the value for E is, the better the link structure of the site is.

The goal was to optimize the link structure of a website. Therefore E has to be optimized. The authors propose an algorithm which computes E for the base situation. Then it will add new links randomly and keep the link if the E for this new structure has increased. If E has decreased, the link is discarded again.

The algorithm proposed by the authors only tries to add links. It might also be interesting to remove existing links and see if E improves.

C. Markov Models

This section will give an introduction to Markov models. A Markov model is a stochastic process with the Markov property. This means the future states of the process are independent of the previous states and depend only on the current state. Based on the Markov property, a Markov chain is defined as a sequence of random variables, the states S, with the Markov property:

$$\begin{aligned} &P(X_{n+1} = x | X_n = x_n, \ldots, X_1 = x_1) = \\ &P(X_{n+1} = x | X_n = x_n) \end{aligned} \tag{5}$$

A Markov chain can also use a memory. The size of the memory is called the order of the Markov chain. Therefore, a m-th order Markov chain is defined as follows:

$$P(X_n = x_n | X_{n-1} = x_{n-1}, \ldots, X_{n-m} = x_{n-m}) \quad (6)$$

In relation to this article the nodes in a graph can be seen as the states in the Markov model and the value of $P(X_n|X_{n-1})$ can be seen as the probability of following a link from node X_{n-1} to node X_n. If $X_n = v$ and $X_{n-1} = w$, then $P(X_n|X_{n-1} = P(v|w)$. Since we are mainly interested in graphs, the Markov model memory can be seen as the path followed by a user to reach a certain node. If a user has followed a path, of n steps, to node w, $z_1 \rightarrow z_2 \rightarrow \cdots \rightarrow z_{n-1} \rightarrow w$, then this history can be used to see what the probability is the user will go to node v given this history:

$$P(v|z_{n-1} \rightarrow w, \ldots, z_1 \rightarrow z_2) \quad (7)$$

If the state space is finite, which is the case for our webgraph, the transition probability of going from one state to another state is modeled with a one-step transition probability matrix Q. In this matrix element $(v, w) = P(v|w) = P(w \rightarrow v)$. Besides the transition probability, an initial probability distribution L is needed as well. This is used to determine a starting situation. The initial probability distribution is just a vector with for each state the probability to start in that state, $\forall_{v \in S} P(v)$.

With these definitions in place, the Markov model is defined as $< S, Q, L >$ where S is the state space, Q is the one-step transition matrix and L is the initial probability distribution.

By multiplying the initial probability distribution with the one-step probability matrix, $L \times Q$, the probabilities of ending up in a certain node after one step are computed. The one-step probability matrix to the power of m, where m is the order of the Markov model, computes the probabilities of going from node v to node w in m steps. Therefore, $L \times Q^m$ computes the probabilities of ending up in a certain node after m steps.

By using the chain rule, the probability of a given path can be computed. The basic chain rule is:

$$P(v, w) = P(v|w)p(w)$$

By applying the chain rule multiple times, the probability of a path in a graph can be computed. Let the followed path of length n be $z_1 \rightarrow z_2 \rightarrow \cdots \rightarrow z_n$, then:

$$\begin{aligned} & P(z_1, z_2, \ldots, z_n) \\ &= P(z_n|z_1, z_2, \ldots, z_{n-1})P(z_1, z_2, \ldots, z_{n-1}) \\ &= P(z_n|z_{n-1})P(z_1, z_2, \ldots, z_{n-1}) \\ &= P(z_n|z_{n-1})P(z_{n-1}|z_{n-2})P(z_1, z_2, \ldots, z_{n-2}) \\ &= P(z_n|z_{n-1})P(z_{n-1}|z_{n-2}) \ldots P(z_2|z_1)P(z_1) \end{aligned}$$

In general, the chain rule can be written as follows:

$$P(z_1 \rightarrow \cdots \rightarrow z_n) = P(z_1) * \prod_{i=2}^{n} P(z_i|z_{i-1}) \quad (8)$$

By using the definition of conditional probability:

$$P(z_2|z_1) = \frac{P(z_1 \rightarrow z_2)}{P(z_1)} \quad (9)$$

and the result of applying the chain rule, the solution is:

$$\begin{aligned} & P(z_1 \rightarrow z_2 \rightarrow \cdots \rightarrow z_n) \\ &= P(z_n|z_{n-1}) \ldots P(z_2|z_1)P(z_1) \\ &= \frac{P(z_{n-1} \rightarrow z_n)}{P(z_{n-1})} \cdots \frac{P(z_1 \rightarrow z_2)}{P(z_1)} P(z_1) \end{aligned}$$

D. Using Markov Models for Website Link Prediction

The authors in the articles, about Markov Models for website link prediction [16] [17] [18], propose a method where a directed graph is created with the pages of the site being the nodes and the hyperlinks between the pages the arcs. By analyzing the web logs, weights are assigned to the edges representing the number of times a hyperlink has been followed by the users of the website. The weights in this graph are used to compute a probability transition matrix containing one-step transition probabilities between the nodes in the directed graph.

A Markov model is constructed by taking the nodes in the directed graph as the states S. Based on the directed weighted graph, a one step probability matrix can be constructed. A start and exit node are added to the graph to put weights on users entering or leaving the website. With this information, the one-step probability transition matrix is created. The one-step transition probability from page v to page w, $P(v \rightarrow w) = P(w|v)$ can be seen as the fraction of traversals, or weight, from v to w over the total number of traversals from v to other nodes:

$$P(w|v) = \frac{weight(v, w)}{\sum_{z|(v,z) \in A} weight(v, z)}$$

where $weight(v, w)$ is the weight of the link between node v and node w.

By using this one-step transition probability matrix, predictions can be made about which page is interesting to the user based on the users link history. Let a user currently be visiting page v and let the users visiting history be a sequence of m pages $z \rightarrow \cdots \rightarrow w \rightarrow v$, where z is the page m link traversals ago. This history is together with the transition matrix to calculate vector *ranking* for the probability of each page to be visited in the next step as follows:

$$\begin{aligned} \textit{ranking} \quad = \quad & a_1 \times L(v) \times Q + \\ & a_2 \times L(w \rightarrow v) \times Q^2 + \cdots + \\ & a_m \times L(z \rightarrow \cdots \rightarrow w \rightarrow v) \times Q^m \end{aligned}$$

where $a_1, a_2, \ldots, a_m$ are the weights assigned to the history vectors. These weights indicate the level of influence these history vectors have on the future. Normally $1 > a_1 > a_2 > \cdots > a_m > 0$ so the closer the history vector is to the present, to more influence it has on the future.

By adding a technique to compress the transition matrix, proposed by [17], even more steps into the future can be computed. Based on these principles a prototype, ONE (Online Navigation Explorer), has been build. It has been used to assist user navigation on the authors university website. The initial feedback from the users is very positive. It was more easy to find useful information by using ONE than not using it.

E. Web Path Recommendations based on Page Ranking and Markov Models

In this article [19] markov model's are, again, used to predict the users' navigation. Instead of using purely usage based statistics from http logs, or uniformly distributed probabilities, the authors propose a method to use structural properties. They use importance in the webgraph for this. The importance is calculated using a PageRank [20] style authority score, which is based on the eigenvector centrality and therefore on the structure of the graph. As in the previous article, web usage logs are used to compute the one-step transition probability matrix Q with the pages of the website as the states S.

The structural part of the approach proposed in this article is based on PageRank. PageRank models the behavior of a user which is surfing the web by choosing an outgoing link of the page he is currently visiting or by jumping to a random page on the web. The PageRank of a page then becomes the probability of the random surfer being at the current page in a particular time step $k > K$. This probability is also correlated with the importance of the page as it is defined as the number and the importance of the pages pointing to it.

$$PR = \epsilon * Q * PR + (1 - \epsilon)p$$

where $(1-\epsilon)$ is a dump factor with ϵ being very small, usually 0.15 and p is usually chosen as $p = \frac{1}{n}$.

The authors propose an algorithm which constructs a weighted tree-like structure, based on the user visiting patterns. A special root node, W, is introduced, the other nodes are instances of the S webpages and all branches terminate in a special leaf-node E. For each user session a branch in the tree is created. If part of this branch overlaps with a existing branch, the overlapping part is merged and the weights of the overlapping part are increased.

This tree structure can be transformed to a markov model and be used to compute the probabilities of the next page. Normally a Markov model is instantiated by assigning either equal probabilities to all nodes or as the ratio between the number of times this page has been visited as a first page and the total number of times a page has been visited as a first page (the total number of user sessions). The first approach favors unimportant pages while the seconds approach favors only top-level entry pages. A better approach, proposed by the authors, is an importance based measure, based on PageRank.

$$L(v) = (1 - \epsilon) \times o(v) + \epsilon \sum_{w \to v} (PR^{n-1}(w) \times o(w, v))$$

$weight(v, w)$ is the number of link traversals and $weight(v) = \sum_{z \to v} weight(z, v)$

Three variants are presented for $o(v)$ and $o(w, v)$:

1) *PR* (PageRank):

$$o(v) = \frac{1}{n}$$

$$o(w, v) = \frac{1}{C_d^{out}(w)}$$

2) SUPR (Semi-Usage PageRank):

$$o(v) = \frac{1}{n}$$

$$o(w, v) = \frac{weight(w, v)}{\sum_{w \to z} weight(w, z)}$$

3) UPR (Usage PageRank:

$$o(v) = \frac{weight(v)}{\sum_{z \in S} weight(z)}$$

$$o(w, v) = \frac{weight(w, v)}{\sum_{w \to z} weight(w, z)}$$

Experiments have been conducted to test the proposed method. In the test setup, five types of instantiations for the markov model have been used. Two purely usage based instantiations (START and TOTAL) and the three page-rank based instantiations (PR, SUPR and UPR). The experiments show promising results in favor of the PR, SUPR and UPR approach. The results however, depend heavily on the data set used. Further research is required in order to determine the superiority of these PR, SUPR and UPR methods.

F. Back to the model again

The early approaches used structural properties combined with usage data to estimate the next (set of)page(s) of interest to the current user. The more advanced approaches are able to incorporate the navigation history of the user to further improve the predictions for the next relevant (set of) page(s).

Based on the discussed techniques, we can describe the P component of the model in a bit more detail, although this property is very situation specific as we will see in the next section. On the other hand, the relation $(I(p), P(p))$ can be defined based on the techniques seen in this section. We have the static property $I(p)$, based on the structure of the graph and we have the dynamic property $P(p)$. If we can come up with a method of transforming these two properties in a initial probability distribution and a one-step transition matrix, Markov Models might be a very good method to define this relation.

What about this $P(p)$ component? It is still quite vague what this should be. If we look at the discussed techniques, they are all more or less based on web logs. These logs contain the exact information of paths traversed by the users of the website and also the number of visits to the specific pages. For website and server administrators this information should be easily obtainable. But outside this web server, this information is very difficult to obtain (if obtainable at all). Some websites might choose to publish this information, but most choose to do not. And even if this information is published, it is susceptible to manipulation since it is very difficult to verify. But for now we will assume we can obtain this information outside the web server, on the scale of the internet or communities on the internet.

This information of traffic and traffic flows over the internet can then be used to construct a weighted graph, which can be

transformed into a one-step probability matrix. This is pretty similar to the approach discussed in the section about Markov Models.

In order to use the model to obtain a ranking of websites which are of the most interest to us, we have to look at all the websites we are not connected to. For each of these websites the mth Markov Model can be used to compute the probability a user will be at that site after following m links. The websites with the highest probability are the most interesting since they have the highest probability of user arriving at their website in m steps.

V. The model into action

So far we have proposed a model to calculate webpage interest, based on a static $I(p)$ and a dynamic $P(p)$ component, and discussed these $I(p)$ and $P(p)$ components related to existing theories and work. This has resulted in a solution where a Markov Model defines the relation between the two components. The Markov Model can be used to compute a mth-order ranking. In this section we will propose a specific situation for which the model can be used. Based on this example we will show how to combine the discussed topics so far, describe an possible implementation of an algorithm and give arguments for the choices we made.

A. The Experiment

The experiment will be situated around the Dutch blogosphere [7]. The blogograph, the graph with all blogs and the links between blogs, data for this experiment is supplied by the company SiteData [8]. Blogs in general are very diverse by nature. Some people use their personal blogs [9] to write about their daily lives and others use them to talk about their ideas. Another category of blogs are the blogs, where entire teams of people work to acquire and present interesting stories for their readers. These blogs can be non profit, run be volunteers, but they can also be funded by commercial companies.

One important aspect is the fact that bloggers like to adopt each others stories. If someone publishes an interesting article, there is a big chance other people will read it and start writing about it as well and thereby creating links (or references) to the original blog.

Another important aspect of the blogosphere is that most bloggers like to discuss blog entries. Most blogs allow other people to post comments about the original post or about other comments. The results in very active user communities around popular blogs.

This fits quite nicely to the proposed approach where we combine the static and dynamic properties. Situated in this setting, we want to use our model to create a ranking of the blogs in the data set, based on combining the two properties. The data set consists of approximately 75K blogs, which will make the nodes in our graph. If we only take into consideration the blogs which actually have links to other blogs, this amount is more reduced a lot. Approximately 12K blogs have links with other blogs (either incoming or outgoing). These 12K blogs are good for approximately 35K directed edges in the graph and approximately 1750K links in total.

Blogs without any links to other blogs are very isolated and can only contribute to our proposed ranking by their dynamic information. They don't have links which enable users to reach their blogs, or reach other blogs from their blog. User will enter their blog by jumping directly to it. Because of this, we will not use these blogs in our ranking.

The amount of unique links between blogs is lower than the total amount of links. This is caused by the fact most blogs feature several articles which we have bundled into one blog.

In the following sections we will discuss how the model can be instantiated and used to compute an actual ranking, situated in the context we have just described.

B. The static property

Based on the static graph structure we have to compute a centrality score for each node. Four major alternatives to compute centrality have been discussed in this article. Based on their conceptual meaning, betweenness centrality, the potential of a point to control the flow in the graph, seems like a very promising candidate.

In order to optimize the calculation for betweenness, we will use ego betweenness. n ego networks have to be computed but the ego networks themselves will be relatively small. The data set consists of approximately 75K blogs, which are the nodes in our graph. If we only take into consideration the blogs which actually have links to other blogs, this amount is reduced drastically. Approximately 12K blogs have links with other blogs (either incoming or outgoing). These 12K blogs contain approximately 35K directed edges in the graph and approximately 1750K links in total. This is because a blog is made up of several postings which link to others. In this experiment we will look at the blog as a single entity. A blog v can have 10 links to blog w, but we are only interested in the fact there is a link.

How do we extract the ego network, $G_{ego} = (V_{ego}, A_{ego})$, for a node $v \in V$ from graph $G = (V, A)$? Based on this definition, two properties hold: (1) $V_{ego} \subseteq V$ and (2) $A_{ego} \subseteq A$ and based on the definition of Freeman all direct neighbors and their arcs of v need to be included.

We perform two steps to extract the ego network. First we will get the set with all nodes in the ego network for a certain node v:

$$V_{ego} = \{v\} \cup \{w \in V | (v, w) \in A \vee (w, v) \in A\}$$

Second, based on the set with nodes in the ego network, V_{ego}, we can construct the set of arcs in the ego network, A_{ego}. If an arc $(v, w) \in V_{ego}$, exists in A then it should also exist in A_{ego}:

$$A_{ego} = \{(v, w) \in A | v \in V_{ego} \wedge w \in V_{ego}\}$$

Now that we have the ego network in place, the actual ego betweenness score, C_b, can be computed. Let $B_{ego} = A^2_{ego} \times (1 - A_{ego})$ where 1 is a matrix with only ones of the

[7] See http://en.wikipedia.org/wiki/Blogspace for more information about the blogosphere.

[8] Sitedata: http://www.sitedata.nl.

[9] The term blog is a contraction of the word web log.

same dimension as A_{ego} and $\times$ is the cellwise multiplication operator for matrices. The ego betweenness is the sum of the reciprocals for the non zero entries in B_{ego}:

$$C_b = \frac{1}{\parallel B_{ego} \parallel_1} \tag{10}$$

If the ego betweenness is computed for all nodes in the graph, the result will be a vector with these scores for each node. Next, this vector is transformed into the initial probability distribution by dividing each centrality score by the sum of all centrality scores:

$$L = \frac{1}{\parallel C_b \parallel_1} C_b \tag{11}$$

Degree centrality could also be an interesting measure to use. It is the potential of being part of a flow in the graph. This claim is weaker than betweenness centrality. Being part of the flow, but not able to control it is not as good. The betweenness measures the participation of a website on shortest paths between two other sites. This means that a higher betweenness score means more of the shortest paths between two points go through the website we are interested in. People generally take the shortest path from A to B, so with a higher betweenness score the chance of people going through the website we are interested in is bigger. With a high degree centrality lot's of routes pass through the website, it has a lot of connection. But if none of this connected are part of a shortest path, people will take the other,shorter, routes. And that will result in less people visiting the website we are interested in. However, it could be interesting to compare results using the two centrality measures on the same data set to compare the differences. We expect the method based on betweenness centrality to perform better.

Closeness centrality is the potential of a point to avoid being part of the flow. Since we are interested in optimizing the flow to our own website (by using the highest ranked website), we are not so much interesting in websites which can avoid the flow of other websites. This might however be a desirable measure if information independence is very important. You could about a ranking with the most independent blogs.

Eigenvector centrality is a bit of an outsider here. This centrality measure will look at the influence of the other nodes and incorporate this into it's centrality value. This is similar to the effect of incorporating multi step paths as the user history when computing our ranking with a mth order Markov Model. Especially if we look at a page rank style approach, it will also incorporate the random aspect of people starting and leaving the website. Mixing these properties, the centrality measure incorporates similar effects as the Markov Model, might result in mixed results. It could however be interesting to look at how this could be effectively incorporated, since page rank style ranking has proven to be pretty reliable.

C. The dynamic property

As mentioned already, it would be ideal to have usage data of all websites on the internet, or just the blogosphere in our case. Unfortunately this is not possible. We have come up with a different approach to work around this problem. Blogs often have the option to post reactions with a topic. These reactions can be characterized by a time stamp on the page. For our experiment we will gather all time stamps associated with a blog and use this as the activity measure for the dynamic property. This approach is based on the assumption that reactions to a blog are related with the traffic of that blog.

This approach has a big advantage, it's easy to add into the crawling process which analyzes the blogs to construct the graph structure. By using these time stamps as a measure for the number of reactions on a blog, we have an easy way to obtain a measure for the activity on a blog.

While reactions are reasonably easy to obtain, they also have some unwanted properties. Spam bots could leave messages and therefore trouble the measured reactions since they aren't real users. Some blogs might have better solutions in place to handle bots than others. At this time, we don't really have a solution for this problem, although we estimate its impact relatively small. It is a disadvantage of our choice to take reactions as an activity measure. However, by increasing the logic used to extract reactions this could improve.

The method we use to extract reactions also has some side effects. There could easily be other time stamps on a page, next to the ones associated with reactions. Every blog entry is usually associated with a time stamp as well. This will add to the activity of the blog but since nearly all blogs suffer from this problem, the relative scores will not change. Another issue with the time stamps is the usage of all kinds of widgets[10]. These widgets can display information which can also contain time stamps. These time stamps should not be counted as reactions to a posting, but the current parse method will include these time stamps as reactions. Not all blogs use the same widgets, therefore the number of incorrect reactions included will differ for each blog. This is a fact we have to take into account when we interpret the results. It could be solved by more complex parsing methods to get the number of reactions on a page.

Another thing to consider, as we only look at time stamps, is the fact we cannot see the number of different users. Why is that important? Some blogs have very small, yet very active, user bases. Two or three users could post a lot of reactions. This would be less valuable with respect to popularity of a blog than just a handful of posts by different users. This problem could also be solved by using a different parsing method, which searches for (username,time stamp) combinations.

We will use the number of time stamps found on a blog as a measure for its popularity or activity. This is a choice based on the fact that real traffic information is not obtainable and reactions to blog postings seem like a reasonable alternative to measure this popularity.

The provided usage data is in the form of $(blog, \#reactions)$. In order to create a weighted graph we will have to assign weights to the arcs, which represent a traversal from website v to website w. The provided data however, provides information about the websites itself and not about the traversals. In order to come with a solution to

[10]See http://en.wikipedia.org/wiki/Web_widget for more information about widgets.

this problem we will make the assumption that, on average, more people will leave website v for a website w with higher traffic. Since a destination website that is active has more traffic, more people are going there. To create a weight based on this assumption, the ratio between the activity of destination website w and the total activity of all destination websites from v has to be computed.

Let $G = (V, A)$ be a graph consisting of a set nodes V and a set arcs, $A \subseteq V^2$. We will denote $(v, w) \in A$ as $v \to w$. For each node $v \in V$ the function $r(v)$ returns the activity measure for the supplied node v. In our case this activity measure is the number of reactions that were posted on a blog. Blog visitors are traveling this network structure. Let $P(w|v)$ be the probability the visitor follows a link to node w given the fact he is currently in node v. These probabilities have to be estimated from the activity measure. So basically we are constructing a flow network, where each link has an unbounded capacity.

Besides by following links, the activity measure of a blog will originate from visitors starting in that particular node and visitors will stop in certain blogs. This is modeled by adding two nodes *source* and *sink* to the graph and create arcs from *source* into each blog and also links from each blog to *sink*. The resulting graph is denoted as $G' = (V', A')$. The activity measure of the new nodes still needs to be defined. Of course, $r(\textit{source})$ is the number of unique visitor to the blog graph. Obviously $r(\textit{source}) = r(\textit{sink})$.

We assume the flow through a link (v, w), from node v to node w, amounts to: $P(w|v)r(v)$. When traversing a link, we assume it is more likely to take a link to a node with a higher activity measure. In other words: $P(w|v) \geq P(z|v)$ if and only if $r(w) \geq r(z)$. Based on this assumption $P(w|v)$ is defined as follows:

$$P(w|v) = \begin{cases} d(v)\frac{r(w)}{R(v)} & \text{if } v \to w \in A \\ 1 - d(v) & \text{if } w = \textit{sink} \\ 0 & \text{if } w = \textit{source} \end{cases} \tag{12}$$

where

$$R(v) = \sum_{w \neq \textit{sink} \in V' : v \to w} r(w) \tag{13}$$

and $d(v)$ is a damping factor that determines the likelihood a visitor stops in a particular blog. It should hold that the sum of all probabilities equals to one, this is shown in the following proof:

$$\begin{aligned} & \sum_{w \in V' : v \to w} P(w|v) \\ = & \sum_{w \in V : v \to w} P(w|v) + P(\textit{sink}|v) + P(\textit{source}|v) \\ = & \sum_{w \in V : v \to w} d(v)\frac{r(w)}{R(v)} + (1 - d(v)) \\ = & \frac{d(v)}{R(v)} \sum_{w \neq \textit{sink} \in V' : v \to w} r(w) + (1 - d(v)) \\ = & d(v) + 1 - d(v) \\ = & 1 \end{aligned}$$

The flow conservation law states that incoming flow and outgoing flow, of a node v, should be equal. This should also hold for this model:

$$\sum_{w \neq \textit{sink} \in V' : w \to v} P(v|w)r(w) = r(v) = \sum_{w \neq \textit{source} \in V' : v \to w} P(w|v)r(v)$$

If we look at the incoming flow we can derive some properties by looking at the following cases:
1) if $v \in V$:

$$\begin{aligned} r(v) &= \sum_{w \neq \textit{sink} \in V' : w \to v} P(v|w)r(w) \\ &= \sum_{w \neq \textit{sink} \in V' : w \to v} d(v)\frac{r(v)}{R(w)}r(w) \\ &= d(v)r(v) \sum_{w \neq \textit{sink} \in V' : w \to v} \frac{r(w)}{R(w)} \end{aligned}$$

we conclude:

$$\frac{1}{d(v)} = \sum_{w \neq \textit{sink} \in V' : w \to v} \frac{r(w)}{R(w)} \tag{14}$$

2) if $v = \textit{sink}$:

$$r(\textit{sink}) = \sum_{w \in V : w \to \textit{sink}} P(v|w)r(w)$$

we conclude:

$$r(\textit{sink}) = r(\textit{source}) = \sum_{w \in V : w \to \textit{sink}} (1 - d(w))r(w) \tag{15}$$

obviously, if $v = \textit{source}$ the sum results in 0.

D. The Ranking

Now that the basic components of the markov model are in place, we can look at how to compute the actual ranking. The one step probability matrix, Q, contains the probabilities of moving from one state to another state in one step. Q^m

contains the probabilities of moving from one state to another state in m steps. To compute the 1-step ranking, $Q \times L$ has to be computed. This results in a vector with the probabilities of being in a node after 1 step. This vector is our ranking. To compute the ranking after m steps, we have to compute $Q^m \times L$ to get our ranking vector.

E. The algorithm

Looking back at what we have discussed so far, we have made the following choices in the context of the proposed experiment.

1) the static property, $I(p)$, will be the initial probability distribution based on ego betweenness centrality: $I = \frac{1}{\|C_b\|_1} C_b$.
2) the dynamic property, $P(p)$, will be the one-step transition matrix based on the reactions (time stamps) found on the blog p.
3) The relation is defined by the Markov Model $< S, Q, L >$ where S is the set of blogs (nodes), Q is the one-step transition matrix and L is the initial probability distribution.

Based on these choices we have developed an algorithm to compute rankings on our data set. We will now introduce and explain the steps of the algorithm in pseudo code. The algorithm can be divided in several steps, the first three steps are the initialization steps and the fourth step is the actual ranking computation. This is a very basic description of the steps needed in the algorithm, no optimizations have been applied.

1) Construct the one-step probability matrix from the weighted graph. See algorithm 1
2) Compute the ego betweenness for all nodes, based on the static graph structure. See algorithm 2
3) Compute the initial probability distribution from the ego betweenness values. See algorithm 3
4) Compute the rankings for all nodes based on a history of m steps. See algorithm 4

In order to describe the actual steps of the algorithm, we will define some basic helper functions. Graph G is a pair of a set of nodes and a set of arcs, $G = (V, A)$, in the algorithm $G.V$ will refer to the set of nodes and $G.A$ will refer to the set of arcs in graph G. Also, $Matrix(G)$, will return the association matrix for the supplied graph G. The function $r(v)$ returns the total number of reactions for node v, based on the supplied data. A $\leftarrow$ is used to add an item to a set, $V_{ego} \leftarrow v$ means we add v to the set V_{ego}. A $\Leftarrow$ is used to assign a weight to a link, $(v, w) \Leftarrow weight$ assigns the value of $weight$ to the link (v, w).

VI. Conclusion and future work

Existing techniques mostly focus on one aspect. Either a ranking based on static data or on dynamic (activity) data. Very few techniques try to combine the two. This has been done on a website level in order to create adaptive websites. In this field promising results have been achieved by techniques based on Markov Models. Extend this to a larger scale to compute a ranking which is based on both static and dynamic properties.

Algorithm 1 $ComputeOneStepTransitionMatrix(G)$

$G' = G$
$G'.V \leftarrow$ *source*
$G'.V \leftarrow$ *sink*

for all $v \in G.V$ **do**
  $G'.A \leftarrow (source, v)$
  $G'.A \leftarrow (v, sink)$
  for all $w \in G'.V | (v, w) \in G'.A \wedge w \neq source$ **do**
    if $w \in G.V$ **then**
      $(v, w) \Leftarrow d(v) * \frac{r(w)}{R(v)}$
    else
      $(v, w) \Leftarrow 1 - d(v)$
    end if
  end for
end for

return $Matrix(G')$

Algorithm 2 $ComputeEgoBetweenness(G, v)$

Ensure: $V_{ego} \subseteq V \wedge A_{ego} \subseteq A$

$V_{ego} \leftarrow v$
for all $w \in G.V$ **do**
  if $(v, w) \in G.A \vee (w, v) \in G.A$ **then**
    $V_{ego} \leftarrow w$
  end if
end for

for all $v \in V_{ego}$ **do**
  for all $w \in V_{ego}$ **do**
    if $(v, w) \in G.A$ **then**
      $A_{ego} \leftarrow (v, w)$
    end if
  end for
end for

$G_{ego} = (V_{ego}, A_{ego})$
$B_{ego} = Matrix(G_{ego})^2 \times (1 - Matrix(G_{ego}))$

return $\frac{1}{\|B_{ego}\|_1}$

Algorithm 3 $CreateInitialProbDist(G)$

$b_{total} = 0$
for all $v \in V$ **do**
  $b = ComputeEgoBetweenness(G, v)$
  $b_{total} = b_{total} + b$
  $L \leftarrow b$
end for

for all $p \in L$ **do**
  $p = \frac{p}{b_{total}}$
end for

return L

In order to answer the questions, asked in the beginning

Algorithm 4 $ComputeRanking(G, k)$

$S \leftarrow G.V$
$Q \leftarrow ComputeOneStepTransitionMatrix(G)$
$L \leftarrow CreateInitialProbDist(G)$
$R = L * Q^k$

of this article, a model has been presented and a summary of the current state-of-the-art has been given. Based on these sections, we have presented a solution, situated around an example. The importance of a website, the static component, can be measured by using any of the known centrality measures. Based on their conceptual meaning, we have chosen to us betweenness centrality in our example. In order to optimize the algorithm we have implemented an ego betweenness algorithm. The dynamic property has been defined as the number of reactions to the postings of a blog in our example. In the most ideal situation however, we should have access to traversal information between websites. The relation between the static and dynamic property is defined by a Markov Model, inspired by research conducted into the field of adaptive websites. The dynamic property is used to construct the one-step probability transition matrix, *Q*, the static property is used to compute the initial probability distribution, *L*, and the states, *S* of the Markov Model are the unique blogs in our example.

In order to optimize interest to a given website, the Markov Model is used to compute a ranking based on a depth of m navigational steps. By creating links, advertising for example, to the highest ranking websites, we can optimize interest for the given website because it's positioned optimal in the graph based on the graph's structural properties combined with actual activity information.

A problem is the fact how to get this dynamic information. Traffic data is not freely available. We propose a solution for this problem in the domain of blogs (and possibly other community based areas). Instead of traffic we will measure reactions to posting, based on the number of time stamps encountered on the website. This has the big advantage to be easy to gather and work with, but two main disadvantages. (1) The results might be polluted with wrong time stamps. A page could contain time stamps which are not related to reactions. In fact, it is probably very common. This can be solved by crawling time stamps related to usernames, since most of the time the name of the user and the time of posting is displayed with an answer.

(2) the other disadvantage is the fact we actually need transition or traversal numbers. If we measure reactions, it's a activity measure of a node in the graph, not an arc. In order to translate the node activity numbers to traversal numbers, we made the assumption it is more likely for people to leave for a page with more visitors. What this means is we will look at the destination nodes of all outgoing links of a certain page. The weight of a specific outgoing arc, is the activity number of the destination node divided by the total activity number of all destination nodes.

The second disadvantage is the more fundamental one. As long as we don't get techniques to effectively measure traversal information, we have to come up with some estimation for this traversal information based on the information we have of the nodes. Therefore, if we could effectively measure traffic (which might be a possibility in the future), we still have this problem. The first disadvantage is only related to the fact we look at reactions instead of traffic and can be refined to get more adequate results. It could also be replaced with more accurate measures, like actual traffic numbers.

A. Future work

Based on the foundations presented so far, some topics are still open and others raised more questions. Therefore more work could be done into the following topics to gather more insight in the proposed model:

1) Perform the described experiment.
2) Incorporate other centrality measures for the static component.
3) Use a betweenness centrality algorithm which takes to whole graph into account.
4) Extend the measuring of reactions. Use a combination of time stamps and usernames to improve the rankings.
5) Research solutions to get actual traffic and/or traversal information.

At this moment, we are working on a prototype implementation to conduct the described experiment. Our intention is to present these results in a follow up article.

VII. Acknowledgements

The author is grateful to Theo van der Weide for his valuable supervision and discussions and to SiteData B.V.[11] for providing the dataset of the .NL-domain.

References

[1] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, "Graph structure in the web," Altavista Company, IBM Almaden Research Center, Compaq Systems Research Center, Tech. Rep., 2003.

[2] J. Rupnik, "Finding community structure in social network analysis - overview," Department of Knowledge Technologies, Jozef Stefan Institute, Tech. Rep., 2006.

[3] M. Hinne, "Local identification of web graph communities," in *Proceedings of the first International Conference on Theory of Information Retrieval (ICTIR)*, 2007, pp. 261–278.

[4] M. Claypool, P. Le, M. Wased, and D. Brown, "Implicit interest indicators," in *Proceedings of the 6th international conference on Intelligent user interfaces.* ACM Press, 2001, pp. 33–40.

[5] L. C. Freeman, "Centrality in social networks - conceptual clarification," *Social Networks*, vol. 1, no. 3, pp. 215–239, 1979.

[6] S. P. Borgatti, "Centrality and network flow," *Social Networks*, vol. 27, no. 1, pp. 55–71, January 2005.

[7] P. B. Bonacich, "Factoring and weighing approaches to status scores and clique identification," *Journal of Mathematical Sociology*, no. 2, pp. 113–120, 1972.

[8] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford Digital Library Technologies Project, Tech. Rep., 1998.

[9] M. Everett and S. P. Borgatti, "Ego network betweenness," *Social Networks*, vol. 27, no. 1, pp. 31–38, January 2005. [Online]. Available: http://dx.doi.org/10.1016/j.socnet.2004.11.007

[10] L. C. Freeman, "Centered graphs and the structure of ego networks," *Mathematical Social Sciences*, vol. 3, no. 3, pp. 291–304, October 1982. [Online]. Available: http://dx.doi.org/10.1016/0165-4896(82)90076-2

[11] http://www.sitedata.nl/

[11] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, pp. 39–43, 1953.
[12] C. H. Hubbell, "An input-output approach to clique identification," *Sociometry*, vol. 28, no. 4, pp. 377–399, December 1965.
[13] M. Perkowitz and O. Etzioni, "Adaptive sites: Automatically learning from user access patterns," Department of Computer Science and Engineering, University of Washington, Seattle, Tech. Rep.
[14] J. Garofalakis, P. Kappos, and D. Mourloukos, "Web site optimization using page popularity," University of Patras, Greece, Tech. Rep., 1999.
[15] B. Zhou, J. Chen, J. Shi, H. Zhang, and Q. Wu, "Website link structure evaluation and improvement based on user visiting patterns," in *in HYPERTEXT '01: Proceedings of the twelfth ACM conference on Hypertext and Hypermedia.* ACM Press, 2001, pp. 241–244.
[16] R. R. Sarukkai, "Link prediction and path analysis using markov chains," in *Proceedings of the 9th international World Wide Web conference on Computer networks : the international journal of computer and telecommunications netowrking*. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., 2000, pp. 377–386.
[17] J. Zhu, J. Hong, and J. G. Hughes, "Using markov models for web site link prediction," School of Information and Software Engineering, University of Ulster at Jordanstown, Tech. Rep., 2002.
[18] J. Zhu, J. Hong, and J. Hughes, "Using markov chains for link prediction in adaptive web sites," in *In Proc. of ACM SIGWEB Hypertext*. Springer, 2002, pp. 60–73.
[19] M. Eirinaki, M. Vazirgiannis, and D. Kapogiannis, "Web path recommendations based on page ranking and markov models," in *in WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management.* ACM Press, 2005, pp. 2–9.
[20] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," 1999.