UNIVERSITY OF VESZPRÉM

Department of Computer Science

World Wide Web Retrieval

an advanced course

Sándor DOMINICH

2005

Characteristics of the Web

Web retrieval owes a considerable part of its success to classical IR and AI. IR has gone through major periods of development from the Boolean Model though the Vector Space Model and Probabilistic Model to present day Language Model. Also, IR has gained a formal theoretical foundation and has thus become an applied formal (mainly mathematical) discipline too (Dominich, 2001; van Rijsbergen, 2004). In the 1980s, knowledge-based expert systems were developed to generate user models in order to assist searching. In the 1990s, ANN-based techniques appeared in IR. Since the Internet and the World Wide Web have become robust and reliable depositing a huge amount of valuable information, Web retrieval has become an important part of everyday life. Popular search engines, like Google, Altavista, Yahoo!, are spidering the Web indexing and retrieving Web pages. However, these search engines suffer from information overload (too many pages to crawl and to show to the user) and low precision. Also, they deviate more and more from their original searching technology roots, and are becoming 'media companies' stressing advertising (Chen, 2003a, 2003b).

There are major differences between classical IR and Web retrieval, they are summarized below:

a) Most Web documents are in HTML (Hypertext Mark Up Language) format, containing many tags. Tags can provide important information about the page; for example, a bold typeface markup (<b>) usually increases the importance of the term it refers to. Tags are usually removed in the indexing phase.
b) In traditional IR, documents are typically well-structured (e.g., research papers, abstracts, newspaper articles) and carefully written (grammar, style). Web pages can be less structured and are more diverse: they can differ in language, grammar, style and length; they can contain a variety of data types including text, image, sound, video, executable. Many different formats are used, such as HTML, XML, PDF, MSWord, mp3, avi, mpeg, etc..
c) While most documents in classical IR tend to remain static, Web pages are dynamic: they can be updated frequently, they can be deleted or added, they can be dynamically generated.
d) Web pages can be hyperlinked, this generates a linked network of Web pages (referred to as Web graph). A URL (Universal Resource Locator) from a Web page to another page, anchor text, the underlined, clickable text can provide additional information about the importance of the target page.
e) The size of the Web, i.e., the number of Web pages and links between them, is orders of magnitudes larger than the size of corpuses and databases used in classical IR.

f) The number of users in the Web is much larger, and they are more diverse in terms of interest, search experience, language, and so on.

These characteristics represent challenges to Web searching. Web retrieval should be able to address these characteristics, to cope with the dynamic nature of the Web, and to scale up with size.

User Problems

- The users do not exactly understand how to provide a sequence of words for the search.

- The users may get unexpected answers because he/she is not aware of the input requirement of the search engine.

- The users have problems understanding Boolean logic.

- Novice users do not know how to start using a search engine.

- The users do not care about advertisements, so the search engine lacks funding.

- Around 85% of users only look at the first page of the result, so relevant answers might be skipped.

In order to solve the problems above, the search engine must be easy to use and provide relevant answers to the query.

Searching Guidelines

- Specify the words clearly, stating which words should be in the page and which words should not be in the page.

- Provide as many particular terms as possible: page title, date, and country.

- If looking for a company, institution, or organization, try to guess the URL by using the www prefix followed by the name, and then (.com, .edu, .org, .gov, or country code).

- Some search engines specialize in some areas. For example, the users can use ResearchIndex (www.researchindex.com) to search research papers.

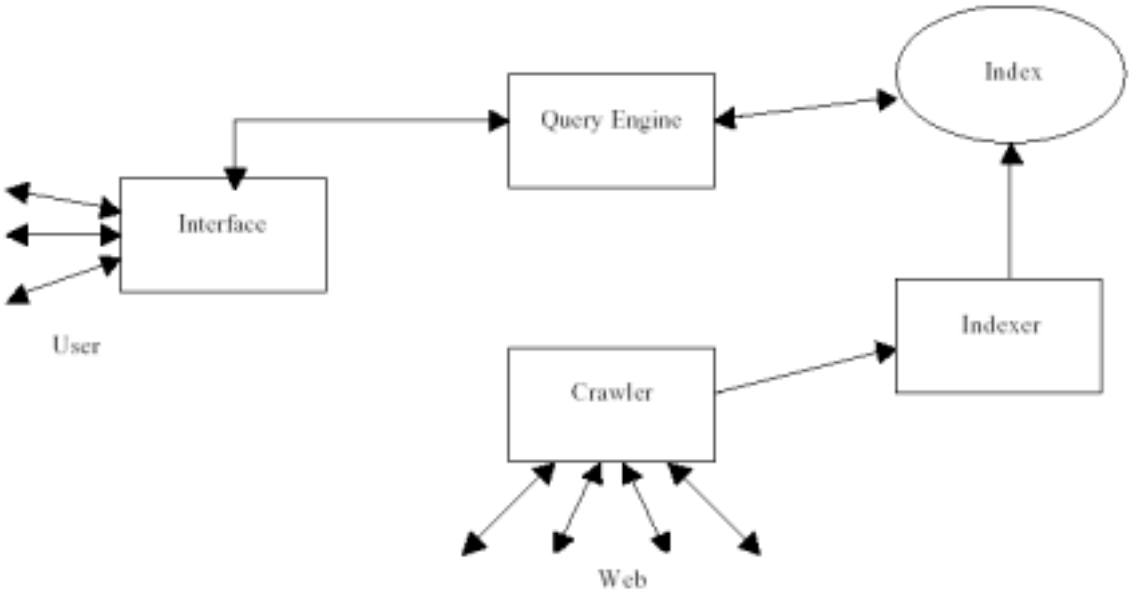
- If the users use broad queries, try to use Web directories as starting points.

- The user should notice that anyone can publish data on the Web, so information that they get from search engines might not be accurate.

Search Engine Architectures

Most search engines use centralized crawler-indexer architecture.

This section discusses the AltaVista search engine as an example for demonstrating how this architecture works. The crawler's duty is to run on a local machine and sends requests to remote Web servers. The index is used in a centralized fashion to answer queries from users. It can be divided into two parts. The first part consists of the user interface and the query engine. The second part contains the crawler and the indexer.

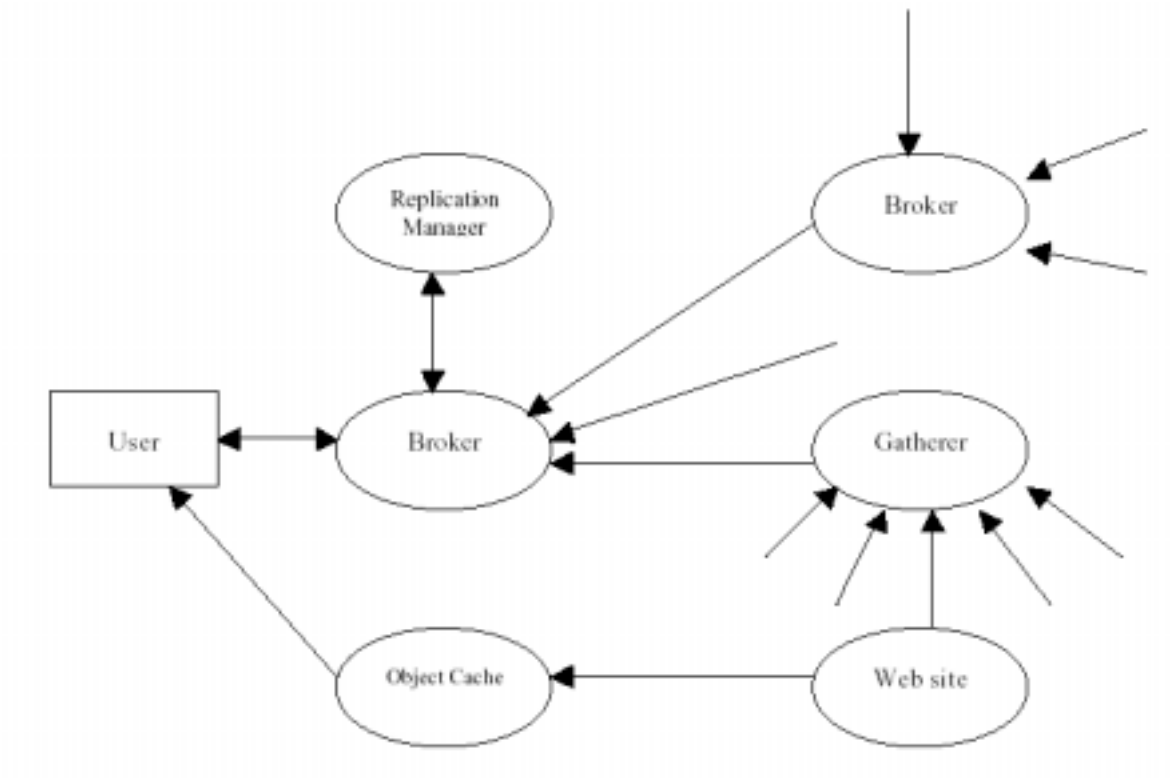


The typical crawler- indexer architecture.

Example. In 1998, AltaVista was running on 20 processors. All processors have 130 Gb of Ram and over 500 Gb of hard disk space. Only the query engine uses more than 75% of these resources.

There are two problems with this architecture. The first problem is data gathering in the dynamic Web environment, which uses saturated communication links, and high load at Web servers. The second problem is the volume of the data. The crawler- indexer architecture does not cope with Web growth in the near future.

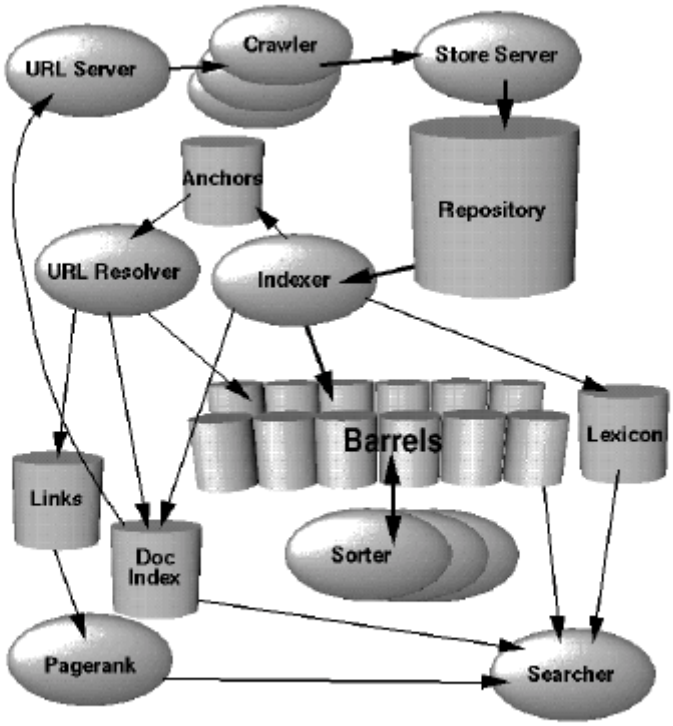
There are several variants of the crawler-indexer architecture. One of the variants is called Harvest. Harvest is the most important variant that uses distributed architecture to gather data and distribute data. It is used by CIA, NASA, the US National Academy of Sciences, and the US Government Printing.



Harvest architecture

Harvest introduces two main elements: gatherers and brokers. The job of gatherers is to collect and extract indexing information from one or more Web servers. Gathering times are specified by the Harvest system. The times are periodic as suggested by its name, Harvest. The job of brokers is to provide the indexing mechanism and the query interface to the data gathered. Brokers receive information from gatherers or other brokers to update their indices. Also, brokers can filter information and send it to others, so that other brokers are saved time. Depending on the configuration of gatherers and brokers, server's workload and network traffic can be balanced. The harvest system builds topic-specific brokers and focuses the index contents thereby avoiding many of the vocabulary and scaling problems of generic indices. In addition, the system provides a replication manager (to replicate servers for enhance user-base scalability) and an object cache (to reduce network and server load).

The word Google comes from the word googol, which means 10100. The Google search engine (www.google.com) heavily uses the structure present in hypertext. It claims that it produces better results than other search engines today. It references about 24 million
Pages. Google is mainly written in C/C++ for efficiency reasons. It can run on Solaris or Linux platforms.



The Google Architecture

The URL Server sends lists of URLs to be fetched by the crawlers. The crawlers download pages according to the list and send the downloaded pages to the Store Server. The Store Server compresses the pages and stores them in the repository. Every Web page has an associated ID number called a docID, which is assigned whenever a new URL is parsed out of a Web page. The index performs an indexing function. It reads the repository, uncompresses the documents, and parses them. Each page is converted into a set of word occurrences called hits. The hits contain information about a word: position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of “barrels” and creates a partially sorted forward index (like bucket sort). It parses out all the links in every Web page and stores important information about them in an anchors file. The anchors file contains information about where each link points from and to and the text of the link. After that, the URL Resolver reads the anchors file and converts relative URLs into absolute URLs and in turn into docID. It puts the anchor text into the forward index, associated with the docID. It generates a links database for storing links and docIDs. The database is used to compute PageRanks for all the documents. The Sorter takes the barrels and resorts them by wordID instead of docID in order to generate the inverted index. Also, the Sorter produces a list of wordIDs and offsets into the inverted index. A program called DumpLexicon takes this list together with the lexicon produced by the indexer and generates a new lexicon to be

used by the searcher. The searcher is run by a Web server and uses the lexicon built by DumpLexicon together with the inverted index and the PageRanks to answer queries.

Crawlers are also called robots, spiders, worms, wanderers, walkers, and knowbots. The first crawler, Wanderer was deve loped by Matthew Gray in 1993. Due to the competitive nature of the search engine business, the designs of these crawlers have not been publicly described. There are several crawling techniques available in public. The simplest one is to start with a set of URLs and from that extracts other URLs recursively in breath-first or depth-first manner. So, search engines allow users to submit top Web sites that will be added to the URL set. A variation to this technique is to start with a set of popular URLs, because the pages for these URLs have information frequently requested. Both techniques work well with one crawler. But when we face multiple crawlers, we have to avoid them crawling the same page. One solution is to send crawlers to different country codes or Internet names to avoid duplicated work. There are problems a crawler needs to cope with. The first problem is that Web pages change dynamically, so the page that the index points to may not exist anymore. That's why the search engine keeps track of date, and shows the date to the query result. Also, the search engine refreshes pages when they are outdated. The fastest crawlers are able to traverse up to 10 million Web pages per day. As mentioned earlier, there are two policies used to traverse Web pages. The first one is breath- first policy. It looks at all the pages linked by the current page and so on. The coverage will be wide but shallow. This may cause the Web server to have many rapid requests. The second is depth- first policy. We follow the first link of a page and we do the same on that page until we cannot go deeper. After that, it returns recursively. The advantage of using depth- first search is deep and space complexity is cheaper. But the disadvantage of using it is narrow.

The Google search engine uses multiple machines for crawling. The crawler works as follows. The crawler consists of five functional components which run in different processes. A URL server process reads URLs out of a file and forwards them to multiple crawler processes. Each crawler process runs on a different machine, which is single threaded. It uses asynchronous I/O to fetch data from up to 300 Web servers in parallel. The crawlers transmit downloaded pages to a single StoreServer process, which compress the pages and store them to disk. Then the indexer process reads pages from disk. It extracts links from the pages and saves them to a different disk file. A URL Resolver process reads the link file, analyzes the URLs contained therein, and saves the absolute URLs to the disk file that is read by the URL server.

Indices

Most indices use variants of inverted files. An inverted file is a list of sorted words. Each word has pointers to the related pages. A logical view of the text is indexed. Each pointer associates a short description about the page that the pointer points to. There are approximately 500 bytes for storing a URL and a short description. So, search engines require 50Gb for indexing 100 million pages. In addition, search engines store answers in memory in case the user asks the same query again. This indexing technique helps to reduce the size of inverted files to about 30% of the size of the text. If we have 100 million pages, it saves us 150Gb space. Do not forget that compression can be used to further reduce the size of indices.

In general, a search engine uses a binary search on the inverted files for searching for a single keyword. If the user types multiple keywords in the query, the search engine searches each of the keywords independently and combine all the results to generate the final answer.

POS
1
10
20
30

A file is a list of words by position

- First entry is the word in position 1 (first word)
- Entry 4562 is the word in position 4562 (4562nd word)

a (1, 4, 40)
entry (11, 20, 31)
file (2, 38)
list (5, 41)
position (9, 16, 26)
positions (44)
word (14, 19, 24, 29, 35, 45)
words (7)
4562 (21, 27)

User Interfaces

The user interface of search engines consists of two parts: the query interface and the answer interface. The basic query interface is a box where a sequence of words is entered. The sequence of words entered into different search engines produces different results. Some search engines support complex query interface, which including Boolean operators and other features, such as phrase search, proximity search, URL searches, title search, date range, and data types search. About the answer interface, search engines usually return pages in the order of relevance to the query. In other words, the most relevant pages appear on the top of the list. Typically, each result entry in the list includes a title of the page, an URL, a brief summary, a size, a date, and a written language.

Ranking

Ranking is the heart of the search engine. In order to produce a good search engine, we need to know how to rank pages properly for the result documents. There is not much information available about this in the public. Today, most search engines use variations of the Boolean or vector model to do ranking. Recall that search engines do not allow access to the text, but only the indices, because it is too expensive in terms of time and space. So, when searching, ranking must use indices while not accessing the text. Besides that, there are also other difficulties as well. There might be too many relevant pages for a simple query. Also, it is difficult to compare two search engines, because of their continuous improvement.

There are three ranking algorithm that serve as basis for ranking: Boolean model, vector space model, and linked-based (connectionist). The first two are the normal ranking algorithms of the Boolean and vector model extended to include pages pointed to by a page in the answer or pages that point to a page in the answer. The third is based only on the terms included in pages having a link to pages in the answer.

Connectionist Web Retrieval Methods

The term soft computing (SC) refers to a family of techniques consisting of methods and procedures based on fuzzy logic, evolutionary computing, artificial neural networks, probabilistic reasoning, rough sets, chaotic computing. In many applications, uncertainty (the lack of precise and exact rules connecting causes and goals) and imprecision (partial or noisy knowledge about the real world) are important features. Also, in many applications the computational effort required by exact conventional methods (also referred to as hard computing) can make the problem at hand intractable

or is unnecessary (when the precision of the output is overweighed by other properties, e.g., more economical or feasible [13].

The application of soft computing techniques to Information Retrieval (IR) aims at capturing aspects that could hardly be satisfactorily modeled by other means. One example for such a technique is based on fuzzy sets theory, which allows for expressing the inherent vagueness encountered in the relation between terms and documents, and fuzzy logic makes it possible to express retrieval conditions by means of formulas in the weighted Boolean model; an overview can be found in [35]. Connectionism represents another approach. Basic IR entities (documents, terms) are represented as an interconnected network of nodes. Artificial neural networks (ANNs) and semantic networks (SN) are two techniques used for this; they are modeled mathematically using graphs. ANN learning allows to model relations between documents as well as documents and terms. It was used with the principal aim to increase the accuracy of document-term weights [3,4,5,15,23,37,39,55]. ANNs were also applied to query modification aiming at enhancing retrieval performance [14], to retrieval from legal texts [44,45], and to clustering for IR [12,21,30,31,44,53].

Three major methods used for Web retrieval tasks are: PageRank, Hubs and Authorities (HITS), Interaction Information Retrieval (I^2R). They are based on different paradigms originally: PageRank: normalized importance; HITS: mutual re-enforcement; I^2R: winner-take-all. But are really different and independent of one another? The paper shows that, albeit they stem originally from different paradigms, they can be integrated into one unified formal framework.

Computational complexity (CC) is an under-treated topic in IR. Most IR researchers and developers understand CC as meaning physical running time. This interpretation of the term CC is justified by and important in practice. However, a correct mathematical justification for the complexity of IR algorithms can be obtained only by properly analyzing complexity. In an ANN, the winner-take-all (WTA) strategy can yield circles, and so one may be tempted to assume — based on the result that finding circles in a graph is ‘hard’ — that an ANN-based IR algorithm may not always tractable (i.e., its CC is not polynomial). But is it really and necessarily the case? The paper shows that intuition may be misleading, and at the same time the CC analysis of the WTA-based retrieval method can serve as a model that may be followed in the analysis of other methods.

Artificial Neural Network

The underlying idea of ANNs goes back to [29], where it is stated, as a fundamental principle, that the amount of activity of any artificial neuron depends on its weighted input, on the activity levels of artificial neurons connecting to it, and on inhibitory mechanisms. This idea gave birth to a huge literature and many applications, especially due to the results obtained in, e.g., [22,25,27]. An *artificial neuron* is a formal processing unit abstracted from real, biological neurons (Fig. 1a). An

artificial neuron ν has inputs $I_1,\ldots,I_n$, these can be weighted by the weights $w_1,\ldots,w_n$. The total input I is a function of the inputs and their weights, usually a linear combination of them, i.e., $I = \sum_i I_i w_i$. The total input I stimulates the neuron which can thus be activated and can take on a *state z*. The state z, also referred to as an *activation level*, is a function g of I, $z = g(I)$. For example, the function g can be a

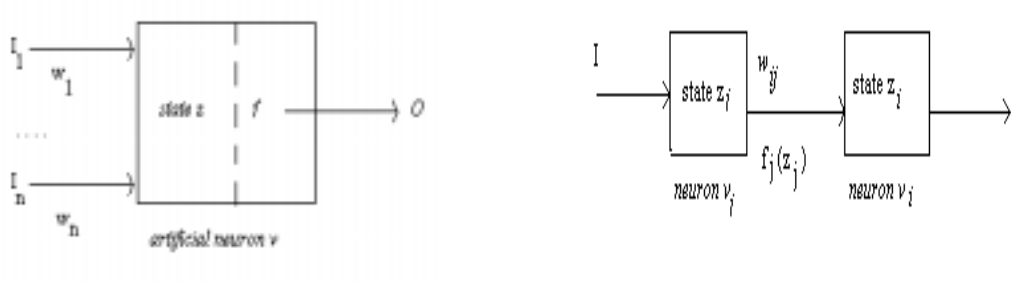
a) threshold function: $z = 1$ if $I > k$, and $z = 0$ if $I \le k$, where k is a threshold value;

b) identity function: $z = I$.

The artificial neuron produces an output O via its *transfer function f* depending on its state z, i.e., $O = f(z)$. The transfer function f can take on several forms, e.g.,

c) identity function: $O = f(z) = z$,

d) sigmoid function: $O = f(z) = \dfrac{1}{1+e^{-z}}$.



a) artificial neuron b) artificial neural network

Fig. 1. **a)** An artificial neuron. A linear combination of the weighted (w_i) inputs (I_i) activates the neuron v which takes on a state z and produces an output O via its transfer function f.
b) ANN. Interconnected artificial neurons (ν_j, ν_i). I is an input to neuron ν_j, and $f_j(z_j)$ is its output, which is an input to neuron ν_i weighted by the quantity w_{ij}, i.e., it is $w_{ij} \cdot f_j(z_j)$

Artificial neurons can be connected to each other thus forming an *Artificial Neural Network* (*ANN*; Fig. 1b). Given two interconnected neurons ν_i and ν_j in an *ANN*, the *output* $f_j(z_j)$ of ν_j can be transferred to ν_i via the connection between them which can alter $f_j(z_j)$ by a weight w_{ij}. The quantity $w_{ij} \cdot f_j(z_j)$ reaches the artificial neuron ν_i for which it is an input.

The following general differential equation can be derived for the state z_i of neuron ν_i [16]:

$$\frac{dz_i(t)}{dt} = -z_i(t) + \sum_{j=1}^{n} f_j(w_{ij}, z_j(t), z_i(t)) + I_i(t) \qquad (1)$$

t denotes time, $z_i(t)$ denotes the activity level of the ith artificial neuron, w_{ij} denotes the weight of a link from the jth to the ith artificial neuron, $I_i(t)$ denotes external input to the ith artificial neuron, $f_j(z_j(t),w_{ij},z_i(t))$ denotes the influence of jth artificial neuron upon the ith artificial neuron. Eq. (1) is a generic equation, and can have different forms depending on the choice of I_i, f_j, w_{ij} corresponding to the particular case or application where the ANN is being used. For example, when applied to neurons, z_i denotes membrane voltage, I_i means an external input, w_{ij} is interpreted as a weight associated to the synapse, whereas f_j takes the from of a product between the weight and z_j. For analogue electric circuits, z_i denotes the potential of a capacitor, the left hand side of the equation is interpreted as a current charging a capacitor to potential z_i, whereas the summed terms mean potentials weighted by conductance. Because eq. (1) can be written for every i = 1, 2,...,n, we have a *system of differential equations*. The study of an *ANN* is done assuming that *initial states* z_0 are known at some initial point t_0. It can be shown that in a small enough vicinity $|z-z_0|$ of z_0 and $|t-t_0|$ of t_0, the system (1) has a unique solution. From a practical point of view, the existence of solutions of (1) can be answered positively due to the Cauchy-Lipschitz theorem [41], and is stated here without proof:

THEOREM 1. *Let*

$$F(t,z)=\frac{1}{\mu_i}(I_i(t)-z_i(t)+\sum_j f_j(z_j(t),w_{ij},z_i(t)))$$

where μ_i *is a coefficient. Consider the initial condition* $z(t_0) = t_0$. *If the function* $F(t,z)$ *is continuous in a region* $\Omega \subset IR^2$ (IR^2 *denotes the real plane*), *and the function* $F(t,z)$ *is a local Lipschitz contraction, i.e.,* $\forall P \in \Omega \ \ \exists K \subset \Omega$ *and* $\exists L_K > 0$ *constant such that* $|F(t,z_1)-F(t,z_2)| \leq L_K|z_1-z_2|$, $\forall(t,z_1), (t,z_2) \in K$, *then there exists a vicinity* $V_0 \subset \Omega$ *of the point* (t_0,z_0) *in which the equation has a unique solution satisfying the initial condition* $z(t_0) = t_0$, *which can be obtained by successive numeric approximations*.□

Eq. (1) gives the state of every neuron at time t. By letting time t to evolve, a sequence $z_i(t)$, i = 1,…,n, of states results. This is referred to as the *operation* of *ANN*. Normally, an *ANN* evolves in time towards a state that does not change anymore. This is called an *equilibrium* and is defined as $dz_i/dt = 0$, i = 1,2,...,n. One important mode of operation of an ANN is referred to as the *winner-take-all* (*WTA*) strategy which reads as follows: only the neuron with the highest state will have output above zero, all the others are 'suppressed'. In other words, *WTA* means to select the neuron

that has maximum state and deactivate all the others; formally the WTA can be expressed as follows:

$$(z_i = 1 \text{ if } z_i = \max_j z_j) \wedge (z_k = 0 \text{ if } z_k \neq \max_j z_j).$$

Artificial neurons can be grouped together to form structures called *layers*. A collection of layers forms a *processing structure* to *solve a problem*. Usually, there are the following types of layers (Fig. 2):

(a) *input layer*: it accepts the problem input data which is called an *input pattern*; there is usually one input layer,

(b) *output layer*: it delivers a solution (which is called an *output pattern*) to the problem to be solved, there is usually one output layer,

(c) *hidden layer*(s): there may be several such layers; they act as intermediary processing ensembles.

Artificial neurons within the same layer usually have the same transfer function, and obey the same *learning rule* (an algorithm according to which the weights change). Typically, learning rules are derived from the *Hebb's Rule* which reads as follows: if two neurons are simultaneously active, the weight between them is increased by a quantity $k \cdot f_i \cdot f_j$. Examples of learning rules derived from Hebb's Rule are the *Delta Rule*, *Kohonen Rule*, etc.. Learning rules are based on *learning methods* such as:

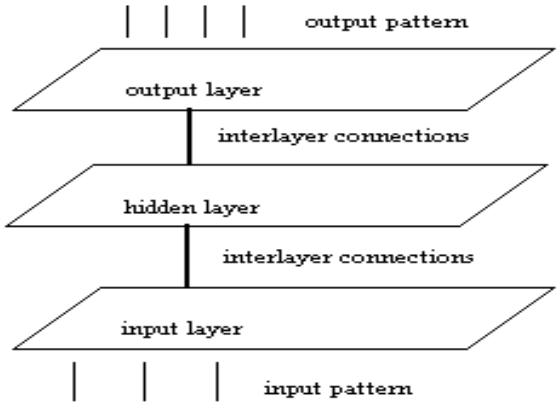


Fig. 2. Typical ANN layered architecture: input, hidden and output layers. The input of the architecture accepts the input pattern, whereas ANN's output is the output pattern. Hidden layers perform intermediary processing

(i) *supervised learning*: both the input pattern and the solution are known; the actual output pattern, produced during the operation of the network, is compared to the solution, and the weights are changed accordingly;

(ii) *unsupervised learning*: the network develops its own classification rules; (iii) *reinforcement learning*: the net is rewarded if the actual output pattern is accepted.

Globally, in a layered *ANN* architecture, there is an activation flow between layers:

(a) *feed-forward propagation*: the activation flows from the input layer towards the output layer;

(b) *feed-backward propagation*: the activation flows from the output layer towards the input layer;

(c) *interactive propagation*: the activation flows both for- and backward;

(d) *equilibrium*: the network relaxes.

Information Retrieval Using Multi-Layered Artificial Neural Networks

In [36], a typical application of ANNs to IR is described: ANNs are used to model a probabilistic IR model using single terms as document components. A 3-layer network (*Q*, *T*, *D*) is considered, *Q* and *D* denote the layer of queries Q_k and documents D_i, respectively. *T* denotes the layer of index terms t_j which are connected bi-directionally with documents and queries. (Fig. 3)

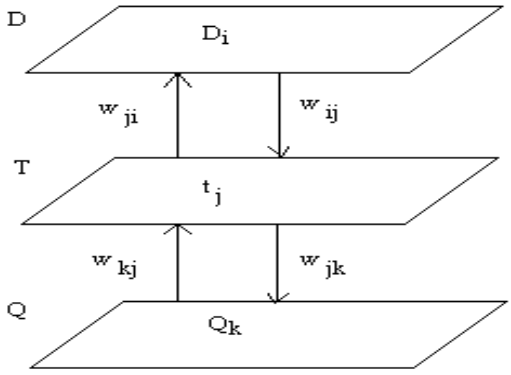


Fig. 3. A 3-layer ANN architecture for IR. There is a query, a document and a terms layer (Q,D,T). There are weighted connections between terms on the one hand, and documents and queries on the other hand

Intra-layer connections are disallowed. The net is considered to be initially blank. As each document neuron is created and fed into the network, associated index term neurons are also created. From the document neuron D_i to the index term neuron t_j a new connection is created and

its strengths w_{ij} is set to $w_{ij} = f_{ij}/L_i$, where f_{ij} denotes the number of occurrences of term t_j in document D_i, and L_i denotes the length (e.g. total number of index terms) of that document. From a term neuron t_j to a document neuron D_i a connection is created if that term is new, and its strengths w_{ji} is set to a special form of inverse document frequency as follows:

$$w_{ji} = \log\frac{p}{1-p} + \log\left(\frac{1-s_{ji}}{s_{ji}}\right) \qquad (2)$$

where p is a small positive constant (the same for all terms), and s_{ji} is the proportion F_j/N, where F_j represents the document collection number of occurrences of the jth term, and N is the total number of terms in the document collection.

Queries Q_k are treated as if they were documents. Let us consider a query neuron Q_k, and clamp its activity to 1. Any term neuron t_j affected by this query will receive as input $1 \cdot w_{kj}$, and outputs the same value. From each of the terms affected by the query some documents D_i (those having terms in common with the query) are affected receiving input $I_i = \sum_j w_{ji} w_{kj}$, and output the same value I_i. The documents so affected turn back the activation to term neurons which receive input $I_j = \sum_i I_i w_{ij}$, and which output the same value I_j to query neurons. These output values can then be used for ranking purposes.

EXAMPLE. Let us consider the document (i=1, p=0.5): D_1=Bayes' Principle: The principle that, in estimating a parameter, one should initially assume that each possible value has equal probability (a uniform prior distribution). Consider the following terms (j=2): t_1=Bayes' Principle, t_2=probability. The connection strengths are as follows: w_{ij} for i=1, j=1, 2: w_{11}=0.5, w_{12}= 0.5; w_{ji} for j=1,2, i=1: w_{11}=0, w_{21}=0. Let the query be Q_k=Bayes' Principle (k=1). The corresponding weights are: w_{kj}, for k=1, j=1, 2: w_{11}=1, w_{12}=0; w_{jk}, for j=1, 2, k=1: w_{11}= $-\infty$, w_{21}=0. Activation spreading: from Q_k an activation with value $1 \cdot w_{kj} = 1 \cdot w_{11} = 1$ is started, it is spread over to D_1 having value $w_{kj} w_{ji} = w_{11} w_{11} = 1$. From D_1, the activation is turned back to the term it has started from, having values 0.5.□

The above model only operates with inter-layer interactions, and does not take into account intra-layer interactions such as, for example, possible relationships between documents or terms. The model was further developed in [38]. The methods for computing term associations can be divided into two categories. One can estimate term relationships directly from term co-occurrence frequencies. On the other hand, one can infer term associations from relevance information through feedback. In the first approach, semantic relationships are derived from the characteristics of term distribution in a document collection. These methods are based on the hypothesis that term co-

occurrence statistics provide useful information about the relationships between terms. That is, if two or more terms co-occur in many documents, these terms would be more likely semantically related. In [54], a method for computing term associations by using a three-layer feed-forward neural network is presented. Term associations are modeled by weighted links connecting different neurons. Each document is represented by a node. Likewise, each query is also represented by a node. Document vectors are input to the network. The nodes in the input layer represent the document terms. These term nodes are connected to the document nodes with individual weights. The nodes in the hidden layer represent query terms. These nodes are connected to the document term nodes. a_{ij} is a weight between the ith document term node and the jth query term node. A value q_j, which represents the importance of term s_j in query q, is used as a scaling factor. The output layer consists of just one node, which pools the input from all the query terms. In this neural network configuration, the weights of the input and output layers are fixed. This makes the task of training the network much easier, because only the weights of the hidden layer are adjustable. The training method was tested using the ADINUL collection, the maximum document length is 95, and the maximum query length is 27, where length is the number of terms in a vector, This means that there are many terms with zero weights in the document and query vectors. It is therefore possible to reduce the dimension of the term-association matrix by eliminating the zero weighted terms. Those terms which are absent in all the query vectors were eliminated. The remaining terms were used to describe the document vectors. In this way, the dimension of the term-association matrix was reduced to 200×200 from an initial dimension of 1217×1217.

Text categorization, also known as automatic indexing, is the process of algorithmically analyzing an electronic document to assign a set of categories (or index terms) that succinctly describe the content of the document. This assignment can be used for classification, filtering, or retrieval purposes. Text categorization can be characterized as a supervised learning problem. We have a set of example documents that have been correctly categorized (usually by human indexers). This set is then used to train a classifier based on a learning algorithm. The trained classifier is then used to categorize the target set. Ruiz and Srinivasan [47] present the design and evaluation of a text categorization method based on a multi-layer ANN. This model uses a ‘divide and conquer’ principle to define smaller categorization problems based on a predefined hierarchical structure. The final classifier is a hierarchical array of neural networks. The method was evaluated using the UMLS Metathesaurus as the underlying hierarchical structure, and the OHSUMED test set of MEDLINE records. The method is scalable to large test collections and vocabularies because it divides the problem into smaller tasks that can be solved in shorter time.

Formal Theory of Connectionist Web Retrieval

Firstly, three major techniques — PageRank, HITS, I^2R — used in Web retrieval tasks are briefly recalled as they were originally proposed (parts 3.1, 3.2, 3.3). They are based on different paradigms:

a) PageRank: normalized importance (the importance of a Web page depends on the importance of Web pages linked to it);
b) HITS: mutual re-enforcement (the importance of hub and authority pages mutually depend on each other);
c) I^2R: winner-take-all (the importance of a Web page is given by its activity level within a reverberatice circle).

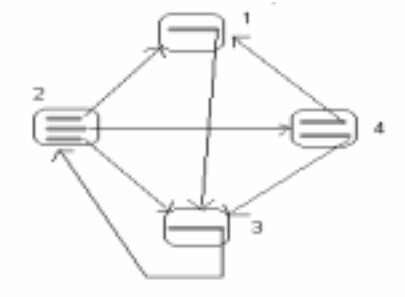
The aim of the present section is to show that the above three techniques, albeit they stem originally from different paradigms, can be integrated into one unified formal view. The conceptual and notational framework used in our approach is given by ANNs and the generic network equation. It will be shown (in parts 3.4, 3.5, 3.6) that the PageRank, HITS and I^2R methods can be formally obtained from the generic equation as different particular cases by making certain assumptions reflecting the corresponding underlying paradigm.

PageRank

In the PageRank method, a Web page's importance is determined by the importance of Web pages linking to it. Brin and Page [6], extending the Garfield static citation principle [24], define the PageRank value R_i of a Web page W_i using the following equation:

$$R_i = \sum_{W_j \in \mathrm{B}_i} \frac{R_j}{L_j} \tag{3}$$

where L_j denotes the number of outgoing links from the page W_j, B_i denotes the set of pages W_j pointing to W_i. Eq. (3) is a homogenous and simultaneous system of linear equations, which always has trivial solution (the null vector), but which has nontrivial solutions too if and only if its determinant is equal to zero. Let $G = (V,A)$ denote the directed graph of the Web, where the set $V = \{W_1,W_i,\ldots,W_N\}$ of vertices denotes the set of Web pages. The set A of arcs consists of the directed links (given by URLs) between pages. Let $M = (m_{ij})_{N\times N}$ denote a square matrix attached to the graph G such that $m_{ij} = 1/L_j$ if there is a link from W_j to W_i, and 0 otherwise (Fig. 4).



Matrix M					PageRank
0	1/3	0	1/2		0.325
0	0	1	0		0.651
1	1/3	0	1/2		0.651
0	1/3	0	0		0.217

Fig 4. A small Web graph *G* with four pages: 1, 2, 3 and 4. The horizontal bars within each page symbolise URLs to other pages as shown by the arrows. The elements of the matrix *M* are also shown, they were computed as follows: $m_{ij}=1/L_j$ (see text). The PageRank values, i.e., the eigenvector corresponding to the eigenvalue 1, were computed using the Mathcad command 'eigenvec(*M*,1)'.

Because the elements of the matrix *M* are the coefficients of the right hand side of eq. (3), this can be re-written in a matrix form as $M\times R=R$, where *R* denotes the vector of PageRank values, i.e., $R = [R_1,\ldots,R_i,\ldots,R_N]^{\mathsf{T}}$. If the graph *G* is strongly connected, the column sums of the matrix *M* are equal to 1. Because the matrix *M* has only zeroes in the main diagonal, the matrix $M–I$ has zero column sums (*I* denotes the unity matrix). Let *D* denote its determinant, i.e., $D = |M–I|$. If every element of, say, the first line of *D* is doubled we get a new determinant *D*', and we have $D' = 2D$. We add now, in *D*', every other line to the first line. Because the column sums in *D* are null, it follows that $D' = 2D = D$, from which we have $D = 0$. The matrix $M–I$ is exactly the matrix of eq. (3), which thus has nontrivial solutions too. The determinant $|M–I|$ being equal to 0 is equivalent to saying that the number 1 is an eigenvalue of the matrix *M*. The PageRank values are computed in practice using some numeric approximation procedure to calculate the eigenvector *R* corresponding to the eigenvalue 1.

Another equation to compute PageRank values in practice is the following:

$$R_i = (1-d) + d \cdot \sum_{W_j \in \mathrm{B}_i} \frac{R_j}{L_j} \qquad (4)$$

where $0 < d < 1$ is a damping factor (e.g., set to 0.85 in practice).

Authorities and Hubs

A method for computing hubs and authorities is suggested in [33]. Two types of Web pages are defined first: hubs and authorities. They obey a mutually reinforcing relationship, i.e., a Web page is referred to as an authority if it is pointed to by many hub pages, and a hub if it points to many authoritative pages (Fig. 5.a)). Given a page p, an authority weight $x^{<p>}$ and a hub weight $y^{<p>}$ is assigned to it. If p points to pages with large x-values, then it receives large y-values, and if p is pointed to by pages with large y-values, then it should receive a large x-value.

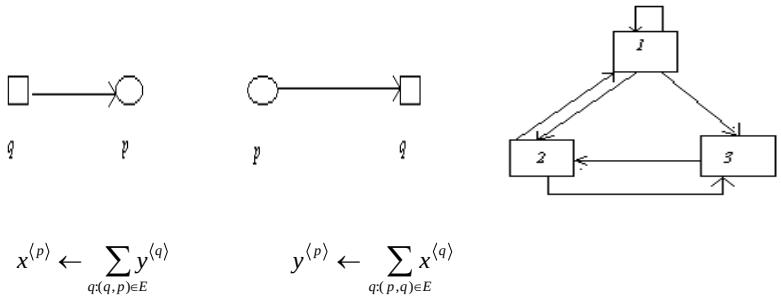


Fig 5. a) Illustration of operations for computing hubs and authorities.
b) Mini-Web (example).

The following iterative operations can be defined:

$$x^{\langle p\rangle} \leftarrow \sum_{q:(q,p)\in E} y^{\langle q\rangle}$$
$$y^{\langle p\rangle} \leftarrow \sum_{q:(p,q)\in E} x^{\langle q\rangle} \qquad (5)$$

where E denotes the set of arcs of the Web graph. To find 'equilibrium' values for the weights of n linked Web pages, a sufficient number of iterations are repeated starting from the initial values $x_0 = (1,\ldots,1)$ and $y_0 = (1,\ldots,1)$ for both x and y. It can be shown that x is the principal eigenvector of $M^{\mathsf{T}}M$, and y is the principal eigenvector of MM^{T}.

EXAMPLE. Let $M = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ denote the adjacency matrix of the mini-Web graph of Fig. 5.b). Perform the operations $x_{i+1} = M^{\mathsf{T}} y_i$ and $y_{i+1} = M x_i$ (after every such iteration length normalize both vectors) until the vectors x and y do not change significantly (convergence). In this example, after four steps the following values are obtained: x = (0.628; 0.628; 0.46) and y = (0.789; 0.211; 0.577).□

Interaction Information Retrieval

In the Interaction Information Retrieval (I^2R) method [18] each Web page o_i is viewed as an artificial neuron, and is associated an n_i-tuple of weights corresponding to its identifiers (e.g., obtained after stemming and stoplisting) t_{ik}, $k = 1,\ldots,n_i$. Given now another page o_j. If identifier t_{jp}, $p = 1,\ldots,n_j$, occurs f_{ijp} times in o_i then there is a link from o_i to o_j, and this has the following weight:

$$w_{ijp} = \frac{f_{ijp}}{n_i} \tag{6}$$

If identifier t_{ik} occurs f_{ikj} times in o_j, and df_{ik} denotes the number of pages in which t_{ik} occurs, then there is a link from o_i to o_j, and this has the following weight:

$$w_{ikj} = f_{ikj} \cdot \log \frac{2N}{df_{ik}} \tag{7}$$

The total input to o_j is as follows:

$$\sum_{k=1}^{n_i} w_{ikj} + \sum_{p=1}^{n_j} w_{ijp} \tag{8}$$

The other two connections — in the opposite direction — have the same meaning as above: w_{jik} corresponds to w_{ijp}, while w_{jpi} corresponds to w_{ikj}. The query Q is considered to be a page, too. The process of retrieval is as follows (Fig. 6 shows and example.). A spreading of activation takes place according to a WTA strategy. The activation is initiated at the query $Q = o_j$, and spreads over along the strongest total connection thus passing on to another page, and so on. After a finite number of steps the spreading of activation reaches a page which has already been a winner earlier thus giving

rise to a loop (referred to as a reverberative circle) This is analogous to a local memory recalled by the query. Those objects are said to be retrieved which belong to the same reverberative circle.

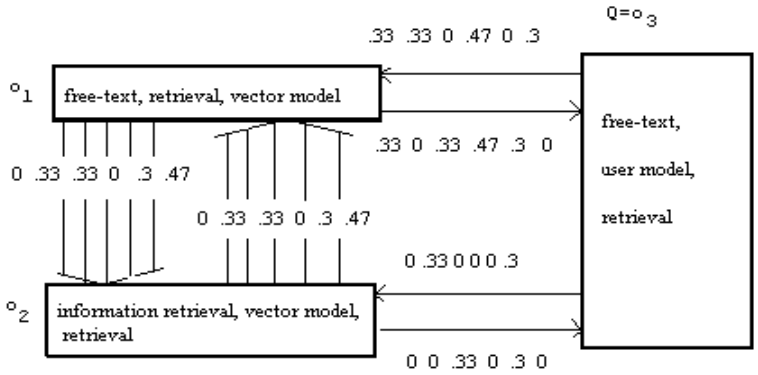


Fig. 6. Retrieval in Interaction Information Retrieval (I^2R). All links having the same direction between Q and o_1, and Q and o_3 are shown as one single arrow to simplify the drawing. The activation starts at *Q*, and spreads over to o_1 (total weight = 0.33+0.33+0.47+0.3= 1.43) from which to o_2, and then back to o_1. o_1 and o_2 form a reverberative circle, and hence o_1 and o_2 will be retrieved in response to Q

Interaction Information Retrieval: Particular Case of the Generic Equation

Let (Fig. 7):

(i) $\Delta = \{O_1, O_2,\ldots,O_i,\ldots,O_N\}$ denote a set of Web pages under focus, each page O_i is assigned an artificial neuron $\aleph_i$, $i = 1,\ldots,N$; thus we may consider $\Delta = \{\aleph_1, \aleph_2,\ldots,\aleph_i,\ldots,\aleph_N\}$,

(ii) $\Phi_i = \{\aleph_k \mid k = 1,\ldots,n_i\}$ denote the set of artificial neurons that are being influenced (i.e., synapsed, pointed to by) by $\aleph_i$, $\Phi_i \subseteq \Delta$,

(iii) $B_i = \{\aleph_j \mid j = 1,\ldots,m_i\}$ denote the set of artificial neurons that influence (i.e., synapse to, point to) $\aleph_i$, $B_i \subseteq \Delta$.

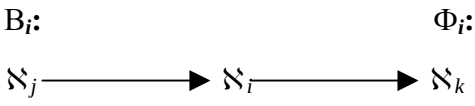


Fig. 7. Objects and links as viewed in I^2R. $\aleph_1, \aleph_2,\ldots,\aleph_i,\ldots,\aleph_N$ form an artificial neural network, $\Phi_i=\{\aleph_k \mid k=1,\ldots,n_i\}$ denotes the set of artificial neurons that are being influenced by $\aleph_i$, $B_i=\{\aleph_j \mid j=1,\ldots,m_i\}$ denote the set of artificial neurons that influence$\aleph_i$

The I^2R technique can be formally derived from the generic eq. (1) as follows.

Because the objects to be searched are IR objects, e.g, pages, no external input (i.e., from outside the Web) can be assumed, so we take $I_i(t) = 0$. One way to define f_j is to conceive the influence of an object j upon another object i as being determined by the strengths of the connections which convey this influence, i.e., weights w_{ij} of the links between them. Eq. (1) thus reduces to the following equation:

$$\frac{dz_i(t)}{dt} = -z_i(t) + \sum_{\aleph_j \in \mathrm{B}_i} w_{ij} \tag{9}$$

In order to simplify the writing, let us introduce the following notation: $\sum_{\aleph_j \in \mathrm{B}_i} w_{ij} = \Sigma^{(i)}$. It is known from the theory of differential equations [41] that the solution of eq. (9) has the following general form

$$z_i(t) = C \cdot e^{-t} + \Sigma^{(i)} \tag{10}$$

where C is a constant depending on the initial condition. When the I^2R network operates for retrieval activation spreading is taking place according to WTA strategy. At any time step t_u, u = 0,1,…, exactly one neuron $k \in \{1,\ldots,N\}$, i.e., the winner, is active, all the other neurons $s \in \{1,\ldots,k-1,k+1,\ldots,N\}$, $s \neq k$, are deactivated, i.e., $z_s(t_u) = 0$. Taking into account this initial condition the activity level of any non-winner neuron s is given by the following function:

$$z_s(t) = (1 - e^{t_u - t})\Sigma^{(s)} \tag{11}$$

If time t is let to increase the activity level $z_s(t)$ tends to stabilize on the total input value $\Sigma^{(s)}$ of that neuron s:

$$\lim_{t \to \infty} z_s(t) = \Sigma^{(s)} \tag{12}$$

At the next time step t_{u+1}, of these neurons s the winner will be that neuron p whose activity level z_p exceeds the activity level z_s of any other neuron s, i.e., $z_p \geq z_s$. This re-writes as follows:

$$(1 - e^{t_u - t})\Sigma^{(p)} \geq (1 - e^{t_u - t})\Sigma^{(s)} \tag{13}$$

Because $t > t_u$ we have $e^{t_u - t} < 1$, and so $(1 - e^{t_u - t})$ is strictly positive. Hence, the winner-condition $z_p \geq z_s$ becomes equivalent to $\Sigma^{(p)} \geq \Sigma^{(s)}$. In other words, the neuron with the highest total input (given by eq. (8)) will be the winner. Fig. 8 shows example — and typical — plots of activity levels $z_s(t)$ for four neurons, it can be nicely seen how the activity levels reach asymptotically their limit which is equal to the corresponding total input values 1, 5, 3, 6.

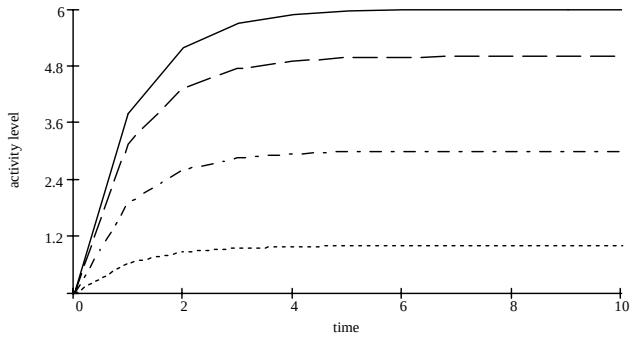


Fig. 8. Typical plots of activity levels for four neurons in I^{2}R. It can be nicely seen how the activity levels reach asymptotically their limit which is equal to the corresponding total input. The highest will be the winner

PageRank: Particular Case of the Generic Equation

The importance levels of Web pages are viewed as the activity levels of associated artificial neurons. Once computed, they remain constant while being used in the retrieval process. Thus, the activity level $z(t)$ may be viewed as a particular case, namely constant in time. Then eq. (1) has a null in its left hand side (the derivative of a constant is zero), and is thus asking for finding the equilibrium as a solution; it becomes (see Fig. 7 for notations):

$$0 = I_i - z_i + \sum_{\aleph_j \in \mathrm{B}_i} f_j(z_j, w_{ij}, z_i) \qquad (14)$$

No external (i.e., from outside the Web) inputs to Web pages are assumed in PageRank, hence we take $I_i(t) = 0$. Taking into account the principle of PageRank, the function f_j does not depend on z_i, but it depends on z_j (the citation level of a Web page depends on the citation levels of the pages linking to it). The function f_j is taken as the dot product of the vector of activity levels and corresponding weights of the pages pointing to it, i.e., $f_j = z_j w_{ij}$. Thus, eq. (14) re-writes as follows:

$$0 = -z_i + \sum_{\aleph_j \in \mathrm{B}_i} z_j w_{ij} \tag{15}$$

Let w_{ij} be defined as a weight meaning the frequency of URL_i (i.e., page j contains the URL of page i) in page j, i.e., $w_{ij} = 1/n_j$, where n_j denotes the number of URLs occurring in page j, and multiple occurrences of an URL are considered as single occurrences, then eq. (15) becomes:

$$z_i = \sum_{\aleph_j \in \mathrm{B}_i} \frac{z_j}{n_j} \tag{16}$$

which is the same as PageRank's eq. (3). If an external input I_i is assumed in eq. (10) and defined as $I_i = 1-d$, where $0 < d < 1$, further the function f_j is defined as $f_j = d \cdot z_j \cdot w_{ij}$, then eq. (4) of PageRank can be obtained:

$$z_i = 1 - d + d \cdot \sum_{\aleph_j \in \mathrm{B}_i} \frac{z_j}{n_j} \tag{17}$$

Hubs and Authorities: Particular Case of the Generic Equation

If the activity level $z(t)$ of eq. (1) is conceived as corresponding to an authority or hub weight x or y, respectively, of eq. (5), and is viewed as being constant in time (when being used; similar to PageRank), then eq. (1) has a null in its left hand side (the derivative of a constant is zero), and $z(t)$ does not depend on time, i.e., $z(t) = z$. Thus, eq. (1) becomes (see Fig. 7 notations):

$$0 = I_i - z_i + \sum_{\aleph_j \in \mathrm{B}_i} f_j(z_j, w_{ij}, z_i) \tag{18}$$

No external inputs to Web pages are assumed, hence we take $I_i = 0$. It can also be seen that the authority of a Web page is influenced only by the authorities of the pages pointing to it. Thus, the function f_j under the summation of eq. (18) becomes $f_j = z_j$. Writing now eq. (18) also for all neurons k pointed to by neurons i, the following two particular cases of the generic equation are obtained:

$$\begin{aligned} 0 &= -z_i + \sum_{\aleph_j \in \mathrm{B}_i} z_j \\ 0 &= -z_k + \sum_{\aleph_k \in \Phi_i} z_i \end{aligned} \tag{19}$$

which are identical with eq. (5).

Power Law in The World Wide Web

The experimental discovery by Faloutsos et al. [7] that the degree distribution for Web pages and Internet nodes follows a power law with a fairly robust degree exponent value was a basic milestone towards the emergence of a new science of the Web. The formulation of the principle of preferential attachment [5] triggered research into and stimulated ideas towards trying to explain, using generative models [13], why the Web link topology evolves according to a power law. Pennock et al. [15] as well as Adamic and Huberman [3] showed that this principle is not necessarily valid in the real Web; modified principles were proposed to better explain the development of a power law for degree distribution in the real Web.

Kahng et al [11] investigated the question of why the degree exponent exhibits a fairly robust behaviour, just above 2. Using a directed network model in which the number of vertices grows geometrically with time, and the number of edges evolves according to a multiplicative process, they established the distribution of in- and out-degrees in such networks. They arrived at the result that if the degree of vertex grows at a higher pace than the edges then the in-degree distribution is independent of the ‘details’ of the network.

The usual method, that of linear regression, used to construct the Power Law is not based on any ‘internal’ property of the Web network—it is a mere reflection of some deeper structure. The generative models proposed thus far are incomplete. They only model growth, and fail to take into account that nodes and links are also destroyed (not just added). It is not known how the processes of growth and extinction go on in the Web, how they relate to each other to give birth to what we observe as a power law.

In the present writing, based on the robustness property of the degree exponent, a different approach is proposed: it is conjectured that, at the present scale of the Web, at the heart of this robustness property lies the Golden Section. The Golden Section is one of the most ancient and overdone yet evergreen topics in mathematics. It is also far-reaching in several other fields, e.g., art, architecture, biology, music, physics. There is a common agreement that it always relates — subjectively — to a notion of ’beauty’ of the field. For example, it is believed that rectangles whose

width-to-height ratio is the Golden Section are the most pleasing to the human eye, or that the timing of musical pieces is considered to be most pleasing to human ears when in Golden Section. In this paper, it is beleived that the evolution of the Web link topology may have an intrinsic property that is reflected in the Golden Section, and this is the expression of an inner beauty of the Web.

After a brief overview of several degree exponent values obtained experimentally, statistical evidence is given to conjecture that the degree exponent value varies around a Golden Section-based value. Using number theoretic results, this conjecture is then used, on the one hand, to propose a method, referred to as F-L method, for the construction of the Power Law for the real Web portion under focus, and, on the other hand, to give a theoretical underpinning for the application of high degree walks in crawling and searching in peer-to-peer networks. Also, formal relationships between the Golden Section and the LCD as well as Bollobás models are shown.

Power Law

If the probability P that a discrete random variable V assumes values equal to or greater than some value v is given by

$$P(V \geq v) = \left(\frac{m}{v}\right)^k, \quad m > 0, k > 0, v \geq m, \tag{1}$$

we say that V follows Pareto's Law [1, 10]. It follows from (1) that:

$$P(V < v) = 1 - \left(\frac{m}{v}\right)^k, \tag{2}$$

which is the distribution function $F(v)$ of V; it is differentiable with respect to v, the derivative is continuous (absolutely continuous). V has density function $f(v) = F'(v) = m^k \cdot v^{-(k+1)}$. The function $f(v)$ is referred to as a Power Law [18], and it is usually written in the following general form:

$$f(v) = C \cdot v^{-\alpha}, \tag{3}$$

where C is a — problem-dependent — constant, α is referred to as the degree exponent. For visualisation purposes, the Power Law is represented in a log-log plot as a straight line obtained by taking the logarithm of (3):

$$\log f(v) = \log C - \alpha \times \log v\ . \tag{4}$$

$\log v$ is represented on the abscissa, $\log f(x)$ on the ordinata, $-\alpha$ is the slope, $\log C$ is the intercept. Given two sequences of values $X = (x_1,..., x_n)$ and $Y = (y_1, ..., y_n)$. If the correlation coefficient $r(X,$

Y) suggests a fairly strong correlation — i.e., it is close to |1| — between *X* and *Y* at a log scale, then a regression line can be drawn to exhibit a relationship between *X* and *Y*; using the slope and the intercept of the regression line the corresponding Power Law can be written.

Degree Exponent Values for the World Wide Web

Faloutsos et al. [7] arrived at the result that, using data provided by the National Laboratory for Applied Networks Research between the end of 1997 and end of 1998, the tail of the frequency distribution of an out-degree — i.e., the number of Internet nodes and Web pages with a given out-degree — is proportional to a Power Law. Their observation was that the values of the exponent seemed to be almost constant: 2.15; 2.16; 2.2; 2.48.

Barabási et al. [5] — using 325,729 HTML pages involving 1,469,680 links from the *nd.edu* domain — confirmed the earlier results obtained for the values of the degree exponent. They obtained the value 2.45 for out-degree, and 2.1 for in-degree.

In [6], two experiments are described using two web crawls, one in May and another one in October 1999, provided by Altavista, involving 200 million pages and 1.5 billion links. The results arrived at were the same in both experiments: the values of the degree exponent were estimated to be 2.1, 2.54, 2.09, 2.67, 2.72 for out-links distribution.

The values obtained earlier for the degree exponent were also confirmed by Pennock et al. [15], who found — using 100,000 Web pages selected at random from one billion URLs of Inktomi Corporation Webmap; they binned the frequencies using histograms — that the exponent for out-degree was 2.72, whereas 2.1 for in-degrees. Similar exponent values were obtained for the in-degree distribution for category specific homepages: 2.05 for companies and newspapers, 2.63 for universities, 2.66 for scientists, and 2.05 for newspapers.

Shiode and Batty [16] assessed the Power Law for Web country domain names in- and out-link distribution as of 1999. Their results for the Power Law exponent were the following values: 2.91, 1.6, 2.98, 1.46, 2.18, 2.

Adamic and Huberman [3] report on an experiment involving 260,000 sites, each representing a separate domain name. The degree exponent was estimated to be 1.94.

In [12], it is reported that a copy of the 1997 Web from Alexa (a company that archives the state of the Web) was used to estimate the degree exponent of the Power Law. The data consisted of about 1 Terabyte of data representing the content of over 200 million web pages. It was found that the degree exponent was 2.38.

In [4], it is reported that the value of 2.3 was found for the degree exponent, in [9] the values 2.1 and 2.38 are reported, while in [14] 2.1 and 2.7.

Friedman et al. [8], using a crawl on the *.hu* domain, assessed the power law for 11,359,640 pages and 95,713,140 links, and found the following values for exponent: 2.29 for in-degree, and 2.78 for out-degree.

Experiment 1. Using the "Barabási-data"[1], we repeated the fitting of a Power Law curve to out-degree distribution. Fig. 1 shows our results. (Computational details are given in the Appendix.)

Experiment 2. We generated the in-links frequency distribution for country domain names[2] as of January, 2004 (Fig. 2). The domain names *.gov, .org, .net, .edu, .us, .com, .mil, .um, .vi* were all considered as representing the USA, the domain names *.ac, .uk, .gb* as representing the UK, and *.fr, .fx* for France. The number of inlinks for every country domain name was identified using Altavista search engine's Webmasters option during 19-22 January, 2004. For example, the UK got a total of 30,701,157 in-links, the USA got 271,019,148; Albania got 2,041,573, Belgium got 3,386,900 in-links. The in-links were binned into 1,000 equally spaced intervals, the correlation coefficient was found to be –0.99 (at a log scale). The value for the Power Law exponent was found to be equal to $\alpha = 1.18$ using Mathcad's *linfit* linear regression command, the approximation error was equal to 14,509.

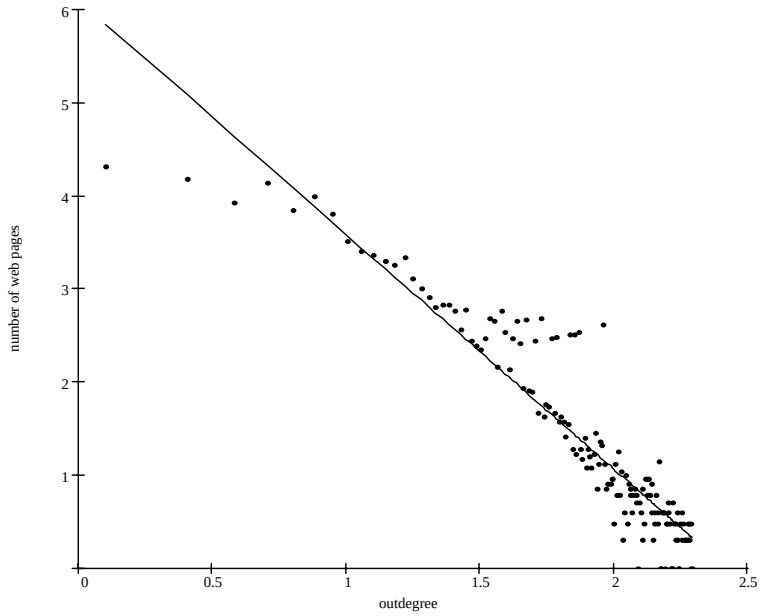


Fig. 1. World Wide Web Power Law. The frequency (i.e., number of Web pages) of the outdegrees of Web pages plotted at a log-log scale. The points represent real values, the straight line represents the regression line fitted to the real values. The correlation coefficient is equal to $r = -0.94$, the Power Law exponent is equal to $\alpha = 2.5$

[1] Provided at http://www.nd.edu/~networks/database/index.html; downloaded January 2, 2004
[2] Taken from http://www.webopedia.com/quick_ref/topleveldomains.

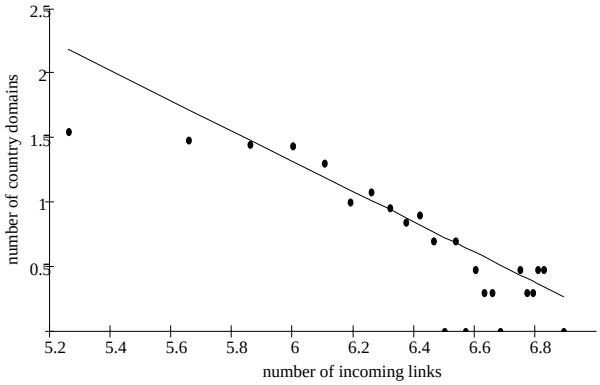


Fig. 2. Log-log plot of the Power Law for the in-links of country domain names as of January, 2004. The correlation between the number of in-links and the corresponding number of country domain names was found to be –0.99, whereas the value of the power law exponent was 1.18

The estimated values obtained experimentally thus far for the exponent of the Power Law for degree distribution in then Web Power Law are summarised in Table 1.

Table 1. Estimated values obtained experimentally thus far for the exponent of the Power Law for degree distribution in the World Wide Web

Source (experiment)	**Degree exponent value**
Faloutsos et al. (1999)	2.15; 2.16; 2.2; 2.48
Barabási et al. (2000)	2.1; 2.45
Broder et al (2001)	2.1; 2.72; 2.09; 2.67; 2.54
Pennock et al. (2002)	2.1; 2.72, 2.05; 2.05; 2.63; 2.66
Kumar et al. (1998)	2.38
Adamic, Huberman (2000)	1.94
Shiode, Batty (2000)	2.91; 1.6; 2.98; 1.46; 2.18; 2
Albert (2000)	2.3
Gil et al. (2004)	2.1; 2.38
Pandarungan (2002)	2.1; 2.7
Friedman et al.	2.29; 2.78

(2003)	
Experiment 1 (see text)	2.5
Experiment 2 (see text)	1.18

Statistics of the Experimentally Obtained Degree Exponent Values

Let us conceive the different degree exponent values obtained experimentally (Table 1) as being a sample drawn from a population [17] consisting of degree exponent values (the population may consist, for example, of the degree exponent values obtained using the data of all Web crawlers, all domain names, etc.). Our sample has size $N = 34$. The mean M of the sample is equal to

$$M = \frac{1}{N}\sum_{i=1}^{N}\alpha_i = 2.284 \qquad (5)$$

The standard deviation s of the sample is equal to

$$s = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\alpha_i - M)^2} = 0.392 \qquad (6)$$

Because all the degree exponent values α_i lie in the open interval (2; 3), the mean μ, whether sample or population ('true') mean, should also lie in this same interval. We may ask ourselves the question of whether there exists positive integer numbers p such that the hypothesis: "$\mu = \sqrt{p}$" be supported. Candidate values for p are 4, 5, 6, 7, 8. Using the

$$z\text{-score}(\mu) = \left|\frac{M - \mu}{s / \sqrt{N - 1}}\right| \qquad (7)$$

the following values are obtained:

z-score($\sqrt{4}$) = 4.163, z-score($\sqrt{5}$) = 0.7,
z-score($\sqrt{6}$) = 2.43, z-score($\sqrt{7}$) = 5.309, z-score($\sqrt{8}$) = 7.988.

The 95% confidence interval for the score is $-2.035 < z\text{-score}(\mu) < 2.035$ ($t_{0.975} = 2.035$, $N - 1 = 33$ degrees of freedom), only z-score($\sqrt{5}$) lies within this interval. Thus, the 95% confidence interval for the mean $\mu = \sqrt{5} = 2.236$ is as follows:

$$2.145 < \mu < 2.422 \qquad (8)$$

We may hence say that there is statistical support to assume that the sample comes from a population with mean $\mu = \sqrt{5}$. Thus, the Power Law for the degree distribution in the World Wide Web may be written in the following form:

$$f(x) \approx C \cdot x^{-\sqrt{5}} \qquad (9)$$

i.e., the number $f(x)$ of Web nodes having degree x is proportional to $x^{-\sqrt{5}}$.

Golden Section, Fibonacci and Lucas Numbers

In this part, those properties of the Golden Section, the Fibonacci and the Lucas numbers are recalled which are of interest to us in connecting them with the Web Power Law.

Golden Section

The Golden Section (*aka* Golden Ratio, Golden Mean, Divine Proportion) is denoted by φ, and defined as the smallest root of the equation:

$x^2 - x - 1 = 0; \quad \varphi = (\sqrt{5} - 1)/2 \approx 0.61803398875$

The other root is $\Phi = (\sqrt{5} + 1)/2 \approx 1.61803398875$. The following relationships hold:

$\sqrt{5} = 2\varphi + 1, \quad \varphi \cdot \Phi = 1$

A straightforward connection between the degree exponent as defined in (9) and the Golden Section is as follows:

$\mu = \sqrt{5} = 2\varphi + 1$

Fibonacci Numbers

The *Fibonacci numbers* are defined as $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$, $n \geq 2$, i.e., 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ... The ratio of the consecutive numbers (i.e., 5/8 = 0.625; 8/13 = 0.615; 13/21 = 0.619;…) has limit equal to the Golden Section, namely:

$$\lim_{n \to \infty} \frac{F_n}{F_{n+1}} = \varphi \qquad (10)$$

The Golden Section and the Fibonacci numbers are related by Binet's formula:

$$F_n = \frac{1}{\sqrt{5}} \left(\Phi^n - (-\varphi)^n \right) \qquad (11)$$

from which it follows that:

$$(-1)^n \cdot \varphi^{2n} + F_n \cdot (2\varphi + 1) \cdot \varphi^n = 1, \quad n = 0, 1, 2, \ldots \qquad (12)$$

Lucas Numbers

If the recurrence relation $L_n = L_{n-1} + L_{n-2}$, $n \geq 2$, is initialised with the numbers $L_0 = 2$, $L_1 = 1$, then one obtains the *Lucas numbers* L_n: 2, 1, 3, 4, 7, 11, 18, 29, 47, 76,… . It can be shown, e.g., using induction on n, that the Fibonacci and Lucas numbers are bound by the following relationship:

$$L_n = F_{n-1} + F_{n+1}, \quad n \geq 1 \qquad (13)$$

Constructing the Web Power Law using Fibonacci and Lucas numbers

In this part, we propose a method based on the Golden Section, Fibonacci and Lucas numbers to construct the Web Power Law for a web portion under focus. Also, experimental evidence will be given to demonstrate the application of the method in practice.

Fibonacci-Lucas (F-L) method

Taking into account the relationship (10), for sufficiently large values of n, we can write:

$$\sqrt{5} = 2\varphi + 1 \approx 2 \cdot \frac{F(n-1)}{F(n)} + 1 = \frac{F(n-1) + F(n-1) + F(n)}{F(n)}. \qquad (14)$$

which — given the recursive definition of the Fibonacci numbers, and taking into account the relationship between The Fibonacci and Lucas numbers — becomes

$$\frac{F(n-2) + F(n-3) + F(n-1) + F(n)}{F(n)} = \frac{L(n-1) + L(n-2)}{F(n)} = \frac{L(n)}{F(n)} \qquad (15)$$

Thus, the Web Power Law (9) re-writes in a form in which the exponent is expressed using both Fibonacci and Lucas numbers as follows:

$$f(x) \approx C \cdot x^{-\frac{L(n)}{F(n)}}. \qquad (16)$$

Taking the logarithm of the relationship (16), one can write the following:

$$\log f(x) \approx \log C - \frac{L(n)}{F(n)} \log x \qquad \log f(x) + \frac{L(n)}{F(n)} \log x \approx \log C \qquad (17)$$

$$F(n)\cdot\log f(x) + L(n)\cdot\log x \approx F(n)\cdot\log C \qquad f(x)^{F(n)}\cdot x^{L(n)} \approx C^{F(n)}$$

For real Web data, $f(x)$ is not a computed value but the actual frequency, while the Power Law exponent is slightly different from $L(n)/F(n)$. Let X_k denote the actual page degrees and Y_k denote the corresponding actual frequency (k = 1, 2, …, M). Then, the relationship (17) becomes:

$$F(n)\cdot\log Y_k + L(n)\cdot\log X_k \approx F(n)\cdot\log C \qquad (\mathbf{18})$$

Because the relationship (18) should hold for every k = 1, 2, …, M, the mean of the left-hand side taken over all k should equal $F(n)\cdot\log C$ (of course, with an inherent approximation error):

$$\frac{1}{M}\sum_{k=1}^{M} (F(n)\cdot\log Y_k + L(n)\cdot\log X_k) \approx \frac{1}{M}\sum_{k=1}^{M} F(n)\cdot\log C = \qquad (\mathbf{19})$$

$$= \frac{1}{M}\cdot M\cdot F(n)\cdot\log C = F(n)\cdot\log C$$

This property makes it possible to propose the following method for constructing a specific Power Law for given real Web data.

Fibonacci-Lucas (F-L) Method for constructing the Web Power Law for degree distribution

Step 1. Establish the number of degrees (e.g., out-links) X_k (in ascending order) and their corresponding frequencies Y_k, , k = 1, 2, …, M, for the Web or Internet nodes under focus.

Step 2. Choose some n, e.g., n = 8, 9 or 19, and compute the corresponding Fibonacci number $F(n)$ and Lucas number $L(n)$ using, for example, Binet's formula (11) and formula (12) respectively (or other formulas available).

Step 3. Compute the left-hand side of the relationship (18), i.e.,

$S_k = F(n)\cdot\log Y_k + L(n)\cdot\log X_k\,, \quad k = 1, 2, \ldots, M$

(**20**)

Step 4. Compute the mean μ of S_k over all *k*, i.e.,

$$\mu = \frac{1}{M}\sum_{k=1}^{M} S_k \qquad (\mathbf{21})$$

Step 5. Apply a correction equal to the standard deviation of S_k to compensate for the approximation errors:

$$\mu' = \mu + \text{stdev}(\mu) \qquad (\mathbf{22})$$

Step 6. Compute the approximate value for the constant *C* as follows (using the relationship (17)):

$$C = 10^{\frac{\mu'}{F(n)}} \qquad (\mathbf{23})$$

Step 7. Write the specific Power Law for the real Web or Internet portion under focus as follows:

$$f(x) \approx C\cdot x^{-\frac{L(n)}{F(n)}} \qquad (\mathbf{24})$$

where, as already seen, *x* denotes degree and *f*(*x*) denotes frequency.

Experimental evidence in support of the F-L method

We give now experimental evidence to support the applicability of the F-L method proposed above in practice.

Using the data of the Appendix, we applied the F-L method using the following pairs of *F*(*n*) and *L*(*n*):

F(8) = 21, *L*(8) = 47
F(19) = 4181, *L*(19) = 9349
F(22) = 17711, *L*(22) = 39603
F(29) = 514229, *L*(29) = 1149851.

For example, when *F*(8) = 21 and *L*(8) = 47, S_k assumes the values 115, 120, 122, 119, etc.. The means μ from Step 4 corresponding to the pairs *F*(*n*) and *L*(*n*) are as follows (rounded to integer values): 112; 22,332; 94,603; 274,6752, respectively, whereas the corrected means μ' of the Step 5 are as follows: 128; 25,576; 108,342; 3,145,661 (rounded to integer values). The application of the Steps 6 and 7 yielded the following Power Laws for the four *F*(*n*) and *L*(*n*) pairs respectively:

$f(x) = 10^{\frac{128}{21}}\cdot x^{-2.23} = 1{,}324{,}3171\cdot x^{-2.23}$

$$f(x) = 10^{\frac{25576}{4181}} \cdot x^{-2.23} = 1{,}309{,}903 \cdot x^{-2.23}$$

$$f(x) = 10^{\frac{108342}{17711}} \cdot x^{-2.23} = 1{,}309{,}904 \cdot x^{-2.23}$$

$$f(x) = 10^{\frac{3145661}{514229}} \cdot x^{-2.23} = 1{,}309{,}904 \cdot x^{-2.23}$$

It can be seen that the values obtained for the constant are fairly stable and compare well with that obtained in the experiment of the Appendix using linear regression: $10^{6.1043} = 1{,}271{,}452$.

High Degree Seeking Walk

In a high degree seeking algorithm (HDS) an arbitrary node is chosen first, then a node with a degree higher than the current node; once the highest degree node has been found, a node of approximately second highest degree will be chosen, and so on. In a peer-to-peer (P2P) system, like GNUTELLA (which obeys a power law), a query is iteratively sent to all the nodes in a neighborhood of the current node until a matching is found. This broadcasting is costly in terms of bandwidth. If every node keeps adequate information (e.g., file names) about its first and second neighbors, then HDS can be implemented. Because storage is likely to remain less expensive than bandwidth, and since network saturation is a weakness of P2P, HDS can be an efficient alternative to usual searching. Adamic et al. show [2] that the expected degree $E(\alpha, n)$ of the richest neighbor of a node having degree n is given by

$$E(\alpha, n) = \frac{n(\alpha - 2)}{(1 - N^{2/\alpha - 1})^n} \sum_{x=0}^{\lfloor N^{1\alpha} \rfloor} x(1+x)^{1-\alpha}(1-(x+1)^{2-\alpha})^{n-1} \qquad (25)$$

where N denotes the number of nodes in the graph, α is the power law exponent. Fig. 3 shows simulation results for the ratio $E(\alpha, n)/n$. It can be seen that for the power law exponent between 2 and 2.3, the chance to find a richer neighbor is higher than the degree of the node itself within a relatively large interval of degree values, which means that HDS can be applied non-trivially. In Web search engines and retrieval, crawlers implement different strategies, e.g., breadth-first-search, to crawl the Web graph. However, it is becoming increasingly difficult to cope with scalability limitations. One possible way is given by an HDS-based crawling strategy which exploits the power law property of link distribution. From eq. (9) we have that $2 < \alpha < 2.3$, which may be viewed as a theoretical justification for the application of HDS to GNUTELLA or to crawling.

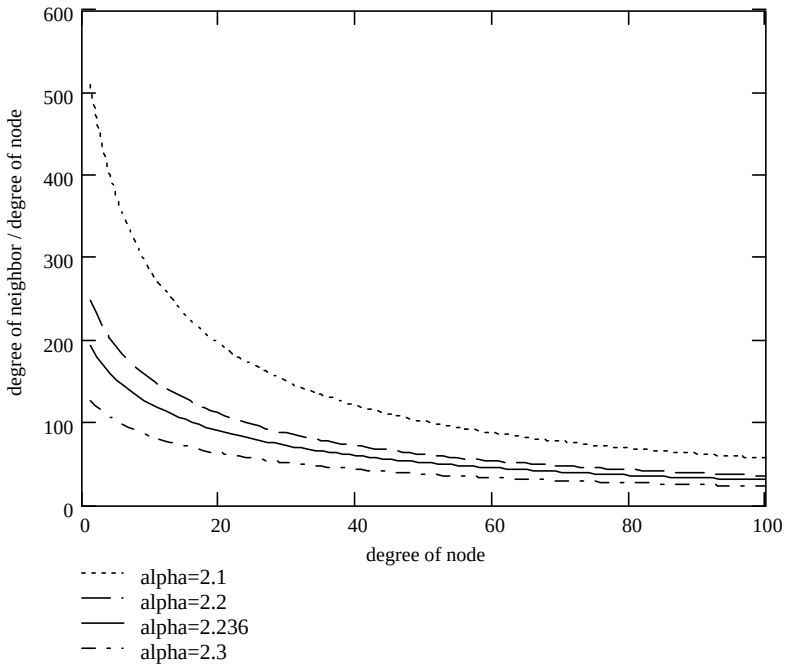


Fig. 3. Simulation of the ratio of the expected degree of the richest neighbor of a node with degree *n* for different values of the power law exponent alpha. The total number of nodes is equal to 100,000,000; and α = alpha = $2\varphi + 1 = \sqrt{5}$

Formal Relationships Between the Golden Section, Degree and Probabilities in Generative Models

Based on results presented by Bollobás[3], we can establish the following formal relationships between the Golden Section and the degree distributions and link probabilities in generative models. In the LCD model, it is shown that in a graph with *n* vertices and *m* edges the fraction $F = \#d/n$ of vertices having degree *d* is bounded as follows:

$(1-\varepsilon)\alpha \le F \le (1+\varepsilon)\alpha,$

where $\alpha = 2m(m+1)/[(d+m)(d+m+1)(d+m+2)]$. Based on (9) and part 4.1, we take $F = d^{-(2\varphi+1)}$, and thus we obtain (taking the logarithm of both sides and appropriately re-arranging) the following relationships between the Golden Section and degree:

$1/[(1-\varepsilon)\alpha d] \le d^{2\varphi} \le 1/[(1+\varepsilon)\alpha d].$

In the Bollobás model, the number $x_i(t)$ of nodes having in-degree *i* at step *t* is given by:

$x_i(t) = Ci^{-(1+1/c)}, \quad c = (\alpha+\beta)/[1+\delta(\alpha+\gamma)], \ \gamma \in \mathbb{R}_+$

[3] Bollobás, B. (2003). Mathematical results on scale-free networks. http://stat-www.berkeley.edu/users/aldous/Networks (downloaded December, 2004)

where α is the probability to add a new vertex together with an edge from it to an old vertex, β is the probability to add an edge between two existing vertices, and γ is the probability to add a new vertex and an edge from an old vertex to it (obviously $\alpha + \beta + \gamma = 1$). Based on (9) and part 4.1, we take $1 + 1/c = 2\varphi+1$, thus we obtain:

$$\varphi = [1 + \delta(\alpha + \gamma)] / [2(\alpha + \beta)].$$

Adamic, L. A. (2003). Zipf, Power-laws, and Pareto – a ranking tutorial. http://ginger.hpl.hp.com/shl/papers/ranking/ranking.html (visited: December 18, 2003)

Adamic, L., Lukose, R.M., Puniyami, A.R., Huberman, B.A. (2001). Search in power-law networks. *Physical Review*, The American Physical Society, vol. 64, pp: 046135(8)

Adamic, L.A., and Huberman, B.A. (2000). Power-Law Distribution of the World Wide Web. *Science*, vol. 287, March, p.2115a

Albert, R. (2000). PhD Dissertation, University of Notre Dame (visited December, 2003).

Barabási, A-L. Albert, R., and Jeong, H. (2000). Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A*, 281, pp: 69-77.

Broder, A., Kumar, R., et.al. (2000). Graph structure in the Web. *Computer Networks*, vol. 33, pp: 309-320.

Faloutsos, M., Faloutsos, P., and Faloutsos, Ch. (1999). On Power-Law Relationship of the Internet Topology. *Proceeedings of the ACM SIGCOMM*, Cambridge, MA, pp: 251-262.

Friedman, E., Uher, M., Windhager, E. (2003). A magyar Web. (The Hungarian Web). *Híradástechnika*, vol. 58, no. 3., pp: 25-31 (in Hungarian)

Gil, R, Garcia, R, Delgado, J. (2004). Are we able to charaterize Semantic Web behaviour? http://dmag.upf.es/livingsw (downloaded December, 2004)

Guilmi, C., Gaffeo, E., and Gallegati, M (2003). Power Law Scaling in the World Income Distribution. *Economics Bulletin*, vol. 15, no. 6, pp: 1-7.

Kahng, B., Park, Y., and Jeong, H. (2002). Robustness of the in-degree exponent for the World Wide Web. *Physical Review*, vol. 66, 046107 1-6.

Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. (1998). Trawling the web for emerging cyber-communities. *Proceedings of the WWW8 Conference*. http://www8.org/w8-papers.

Menczel, F. (2002). Growing and navigating the small world Web by local content. *Proceedings of the PNAS*, vol. 99, no. 22, pp: 14014-14019.

Pandurangan, G., Raghavan, P. (2002). Using PageRank to charaterize Web Structure. *Lecture Notes In Computer Science*, vol. 2387, Springer Verlag, pp: 330-339

Pennock, D.M., Flake, G.W., Lawrence, S., Glover, E., and Giles, L. (2002). Winners don't take all: characterizing the competition for links on the Web. *Proceedings of the New York Academy of Sciences*, April 16, vol. 99, no. 8, pp: 5207-5211.

Shiode, N., and Batty, M. (2000). Power Law Distribution in Real and Virtual Worlds. http://www.isoc.org/inet2000/cdproceeedings/2a/2a_2.htm (downloaded on December 18, 2003.

Spiegel, M.R. (1961). *Theory and Problems of Statistics*. McGraw Hill.

Zipf, G. K. (1949). *Human Behaviour and Principle of Least Effort*. Addison Wesley, Cambridge, Massachussetts.

Appendix

Experiment 1. Using the "Barabási-data"[4], we repeated the fitting of a Power Law curve to out-degree distribution. The data was provided as a zipped file; after unzipping it the result was a text

[4] Provided at http://www.nd.edu/~networks/database/index.html; downloaded January 2, 2004

file which contained two numbers in each line: the leftmost number was the sequence number of Web pages (0; 1; 2; …; 325,729), the other number was the sequence number of the Web page pointed to by the page represented by the leftmost number. A noteworthy observation is that the exponent of the Web Power Law is slowly increasing from 1 with the number of pages (from a few hundred up to several ten thousand pages), and is starting to stabilise around the value $\alpha = 2.5$ if the number of Web pages involved is fairly high, above 100,000. Thus, for example, for 30,000 pages, the correlation — at a log scale — r between out-degree and frequency was only $r = -0.892$, and the fitting of a Power Law curve $C \cdot x^{-\alpha}$ using Mathcad's in-built curve fitting command *genfit* resulted in $\alpha = 0.867$ with an approximation error of the sum of the absolute values of differences of 3.7×10^6 at 10^{-4} convergence error, whereas using linear regression yielded $\alpha = 1.47$ with an approximation error of 1,589,104 at 10^{-4} convergence error. Fig. 1 shows our results (see text) for a number of 256,062 Web pages — involving 1,139,426 links — selected at random from the provided 325,729 pages. After processing this file the X data consisted of the out-degrees of Web pages, whereas the Y data consisted of the corresponding frequencies. For example, there were 2,206 pages having out-degree 13, and the out-degree 14 had its frequency equal to 1,311. The empirical correlation coefficient — taking log scale data — r between out-degree and frequency was $r = -0.94$. The linear regression yielded the following values: $\alpha = 2.5$ for the exponent; and $C = 10^{6.1043}$ for the constant. The computation was performed using Matchcad's in-built *line* command; the numeric computation used in this command as well as the fact that we used 69,667 pages less may account for the difference of 0.05 in the exponent value compared to the value reported in [4]. Because of the strong correlation (see above) and power-like behaviour, and also due to inherently present numeric approximation errors, we believe that the difference of 0.05 is not an important one.

References

1. Arasu, A. (2002). PageRank Computation and the Structure of the Web: Experiments and Algorithms. *Proceedings of the World Wide Web 2002 Conference*, Honolulu, Hawaii, USA, 7-11 May, http://www2002.org/CDROM/poster (visited: 4 Nov 2002)
2. Bartell, B. T. (1994). Optimizing Ranking Functions: A Connectionist Approach to Adaptive Information Retrieval. *Ph.D. Thesis*, Department of Computer Science and Engineering, University of California, San Diego, 1994. http://www.cs.ucsd.edu/groups/guru/publications.html (visited: 10 May 2004)
3. Belew, R.K. (1987). A Connectionist Approach to Conceptual Information Retrieval. *Proceedings of the International Conference on Artificial Intelligence and Law* (pp. 116-126). Baltimore, ACM Press.
4. Belew, R.K. (1989). Adaptive information retrieval: Using a connectionist representation to retrieve and learn about documents. *Proceedings of the SIGIR 1989* (pp. 11-20). Cambridge, MA, ACM Press.
5. Bienner, F., Giuvarch, M. and Pinon, J.M. (1990). Browsing in hyperdocuments with the assistance of a neural network. *Proceedings of the European Conference on Hypertext* (pp. 288-297). Versailles, France.
6. Brin, S., and Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Proceedings of the 7th World Wide Web Conference*, Brisbane, Australia, 14-18 April, pp: 107-117
7. Chang, E. and Li, B. (2003). MEGA – The Maximizing Expected Generalization Algorithm for Learning Complex Query Concepts. *ACM Transactions on Information Systems*, 21(4), pp: 347–382.
8. Chen, H. (2003a). Introduction to the JASIST special topic section on Web retrieval and mining: a machine learning perspective. *Journal of the American Society for Information Science and Technology*, vol. 54, no. 7, pp: 621-624.
9. Chen, H. (2003b). Web retrieval and mining. *Decision Support Systems*, vol. 35, pp: 1-5.
10. Chen, H., Fan, H., Chau, M., Zeng, D. (2001). MetaSpider:Meta-Searching and Categorisation on the Web. *Journal of the American Society for Information Science and Technology*, vol. 52, no. 13, pp: 1134-1147.
11. Cheun, S. S. and Zakhor, A. (2001). Video Similarity Detection with Video Signature Clustering. *Proceedingsof the 8th IEEE International Conference on Image Processing*, vol. 1. pp: 649–652.
12. Cohen, P., and Kjeldson, R. (1987). Information retrieval by constrained spreading activation in semantic networks. *Information Processing and Management*, 23, 255-268.
13. Cordon, O., Herrera-Viedma, E. (2003). Editorial: Special issue on soft computing applications to intelligent information retrieval. *International Journal of Approximate Reasoning*, vol. 34, pp: 89-95.
14. Crestani, F., Lee, P. L. (2000). Searching the web by constrained spreading activation. *Information Processing and Management*, vol. 36, pp: 585-605.
15. Cunningham S.J., Holmes G., Littin J., Beale R., and Witten I.H. (1997). Applying connectionist models to information retrieval. In Amari, S. and Kasobov, N. (Eds.) *Brain-Like Computing and Intelligent Information Systems* (pp 435-457). Springer-Verlag.
16. De Wilde, Ph. (1996). *Neural Network Models*. Springer Verlag.
17. Ding, C., He, X., Husbands, P., Zha, H., Simon, H.D. (2002). PageRank, HITS, and a unified framework for link analysis. *Proceedings of the ACM SIGIR 2002*, Tampere, Finland, pp: 353-354.
18. Dominich, S. (1994). Interaction Information Retrieval. *Journal of Documentation*, 50(3), 197-212.
19. Dominich, S. (2001). *Mathematical Foundations of Information Retrieval*. Kluwer Academic Publishers, Dordrecht, Boston, London.
20. Dominich, S. (2004). Connectionist Interaction Information retrieval. *Information Processing and Management*, vol 39, no.2, pp: 167-194
21. Doszkocs, T., Reggia, J., and Lin, X. (1990). Connectionist models and information retrieval. *Annual Review of Information Science & Technology*, 25, 209-260.
22. Feldman, J.A., and Ballard, D.H. (1982). Connectionist models and their properties. *Cognitive Science*, vol. 6, pp: 205-254
23. Fuhr, N. and Buckley, C. (1991). A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3), 223-248.
24. Garfield, E. (1955). Citation indexes for science. *Science*, p. 108

25. Grossberg, S. (1976). Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biological Cybertnetics*, vol. 23, pp: 121-134
26. Haveliwala, T.H. (1999). *Efficient Computation of PageRank*. Stanford University, http://dbpubs. stanford.edu:8090/pub/1998-31 (visited: 27 Febr 2004)
27. Hopfield, J.J. (1984). Neurons with graded response have collective computational properties like those of two-states neurons. *Proceedings of the National Academy of Sciences*, vol. 81, pp: 3088-3092
28. Huang, Z., Chen, H., Zeng, D. (2004). Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. *ACM Transactions on Information Systems*, vol. 22, no. 1, pp: 116-142.
29. James, W. (1890). *Psychology (Briefer Course)*. New York: Holt, Chapter XVI, "Association", pp: 253-279
30. Johnson, A., and Fotouhi, F. (1996). Adaptive clustering of hypermedia documents. *Information Systems*, 21, 549-473.
31. Johnson, A., Fotouhi, F., and Goel, N. (1994). Adaptive clustering of scientific data. *Proceedings of the 13th IEEE International Phoenix Conference on Computers and Communication* (pp. 241-247). Tempe, Arizona.
32. Kim, S.J., and Lee, S.H. (2002). An Improved Computation of the PageRank Algorithm. In: Crestani, F., Girolamo, M., and van Rijsbergen, C.J. (eds.) *Proceedings of the European Colloquium on Information Retrieval*. Springer LNCS 2291, pp: 73-85
33. Kleiberg, J. M. (1999). Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, vol. 46, no. 5, pp: 604-632.
34. Kohonen, T. (1988). *Self-Organization and Associative Memory*. New York: Springer Verlag.
35. Kraft, D.H., Bordogna, P. and Pasi, G. (1998). Fuzzy Set Techniques in Information Retrieval. *In*: Didier, D. and Prade, H. (Eds.) *Handbook of Fuzzy Sets and Possibility Theory. Approximate Reasoning and Fuzzy Infomation Systems*, (Chp. 8). Kluwer Academic Publishers, AA Dordrecht, The Netherlands.
36. Kwok, K.L. (1989). A Neural Network for the Probabilistic Information Retrieval. In Belkin, N.J. and van Rijsbergen, C.J. (Eds.) *Proceedings of the 12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, Cambridge, MA, USA, pp: 21-29.
37. Kwok, K.L. (1990). Application of Neural Networks to Information Retrieval. In Caudill, M. (Ed.) *Proceedings of the International Joint Conference on Neural Networks*, Vol. II (pp. 623-626). Hilldale, NJ, Lawrance Erlbaum Associates, Inc.
38. Kwok, K.L. (1995). A network approach to probabilistic information retrieval. *ACM Transactions on Information Systems*, 13(3), 243-253.
39. Layaida, R., Boughanem, M. and Caron, A. (1994). Constructing an Information Retrieval System with Neural Networks. *Lecture Notes in Computer Science*, 856, Springer, pp: 561-570.
40. Lempel, R., Moran, S. (2001). SALSA: the stochastic approach for link-structure analysis. *ACM Transactions on Information Systems*, vol. 19, no. 2, pp: 131-160.
41. Martin, W. T., Reissner, E. (1961). *Elementary Differential Equations*. Addison-Wesley, Reading-Massachusetts, U.S.A.
42. Niki, K. (1997). Sel-organizing Information Retrieval System on the Web: Sir-Web. In Kasabov, N. et al. (Eds.) *Progress in Connectionist-based Information Systems. Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Information Systems*, vol. 2, Springer Verlag, Singapore, pp: 881-884.
43. Orponen, P. (1995). Computational Complexity of Neural Networks: A Survey. *Nordic Journal of Computing*, vol. 1, pp: 94-110.
44. Rose, D. E. (1994). *A symbolic and connectionist approach to legal information retrieval*. Hillsdale, NJ, Erlbaum.
45. Rose, D.E. and Belew, R.K. (1991). A connectionist and symbolic hybrid for improving legal research. *International Journal of Man-Machine Studies*, 35(1),1-33.
46. Roussinov, D.G., Chen, H. (2001). Information navigation on the Web by clustering and summarizing query results. *Information Processing and Management*, vol. 37, pp: 789-816.
47. Ruiz, M.E., Srinivasan, P. (1999). Hierarchical Neural Networks for Text Categorization. *Proceedings of the 22nd ACM SIGIR International Conference on Research and Development in Information Retrieval*, Berkeley, California, USA, pp:281-282.

48. Schlieder, T. (2002). Schema–Driven Evaluation of ApproXQL Queries. *Technical Report B02–01*, Freie Universität Berlin, January 2002. http://www.inf.fu-berlin.de/inst/ag-db/publications/2002/report-B-02-01.pdf (visited: 10 May 2004)
49. Sheikholeslami, G., Chang, W. and Zhang, A. (2002).SemQuery: Semantic Clustering and Querying on Heterogeneous Features for Visual Data. *IEEE Transactions on Knowledge and Data Engineering*, 14(5), pp: 988-1003.
50. Sima, J., Orponen, P. (2003). General-Purpose Computation with Neural Networks: A Survey of Complexity Theoretic Results. *Neural Computation*, vol. 15, pp: 2727-2778.
51. Van Rijsbergen, C.J. (2004). *The Geometry of IR*. Cambridge University Press.
52. Weiss, M.A. (1995). *Data Structures and Algorithm Analysis*. The Benjamin/Cummings Publishing Company, Inc., New York, Amsterdam.
53. Wermter S. (2000). Neural Network Agents for Learning Semantic Text Classification. *Information Retrieval*, 3(2), 87-103.
54. Wong, S.K.M., Cai, Y.J. (1993). Computation of Term Association by Neural Networks. *Proceedings of the 16th ACM SIGIR International Conference on Research and Development in Information Retrieval*, Pittsburgh, PA, USA, pp:107-115.
55. Yang, C.C., Yen, J., Chen, H. (2000). Intelligent internet searching agent based on hybrid simulated annealing. *Decision Support Systems*, vol. 28, pp: 269-277.