The Web as a graph

Ravi Kumar* Prabhakar Raghavan Sridhar Rajagopalan D. Sivakumar
Andrew Tomkins Eli Upfal†

ABSTRACT

The pages and hyperlinks of the World-Wide Web may be viewed as nodes and edges in a directed graph. This graph has about a billion nodes today, several billion links, and appears to grow exponentially with time. There are many reasons—mathematical, sociological, and commercial—for studying the evolution of this graph. We first review a set of algorithms that operate on the Web graph, addressing problems from Web search, automatic community discovery, and classification. We then recall a number of measurements and properties of the Web graph. Noting that traditional random graph models do not explain these observations, we propose a new family of random graph models.

1. OVERVIEW

The World-Wide Web has spawned a sharing and dissemination of information on an unprecedented scale. Hundreds of millions—soon to be billions—of individuals are creating, annotating, and exploiting hyperlinked content in a distributed fashion. These individuals come from a variety of backgrounds and have a variety of motives for creating the content. The hyperlinks of the Web give it additional structure; the network of these links is a rich source of latent information.

The subject of this paper is the directed graph induced by the hyperlinks between Web pages; we refer to this as the *Web graph*. In this graph, nodes represent static html pages and directed edges represent hyperlinks. Recent estimates [5] suggest that there are over a billion nodes in the Web graph; this quantity is growing by a few percent a month. The average node has roughly seven hyperlinks (directed edges) to other pages, so that the graph contains several billion hyperlinks in all.

*IBM Almaden Research Center K53/B1, 650 Harry Road, San Jose CA 95120.
†Computer Science Department, Brown University, Providence, RI.

The network of links in this graph has already led to improved Web search [7, 10, 23, 32] and more accurate topic-classification algorithms [14], and has inspired algorithms for enumerating emergent cyber-communities [25]. The hyperlinks further represent a fertile source of sociological information. Beyond the intrinsic interest of the topology of the Web graph, measurements of the graph and the behavior of users as they traverse the graph are of growing commercial interest.

1.1 Guided tour of this paper

In Section 2 we review several algorithms that have been applied to the Web graph: Kleinberg's HITS method [23] and extensions, the enumeration of certain bipartite cliques [25], and classification algorithms utilizing hyperlinks [14].

In Section 3 we summarize a number of measurements on large portions of the Web graph. We show that the in- and out-degrees of nodes follow inverse polynomial distributions [15, 20, 29, 33], and we study the distributions of more complex structures. We present measurements about connected component sizes, and from this data we draw conclusions about the high-level structure of the web. Finally, we present some data regarding the diameter of the web.

In Section 4 we show that the measurements described above are incompatible with traditional random graph models such as $\mathcal{G}_{n,p}$ [8]. ($\mathcal{G}_{n,p}$ is a model for n-node graphs in which an edge connecting any pair of nodes occurs independently with probability p.) We describe a class of new random graph models, and give evidence that at least some of our observations about the Web (for instance, the degree distributions) can be established in these models. A notable aspect of these models is that they embody some version of a *copying* process: a node v links to other nodes by picking a random (other) node u in the graph, and copying some links from it (i.e., we add edges from v to the nodes that u points to). One consequence is that the mathematical analysis of these graph models promises to be far harder than in traditional graph models in which the edges emanating from a node are drawn independently. We conclude in Section 5 with a number of directions for further work.

1.2 Related work

The structure of the Web graph has been exploited to enhance the quality of Web search [6, 7, 10, 11, 23, 32]. In the setting of supervised classification [14], the topics of pages

pointed to by a Web page v can be used to improve the accuracy of determining the (unknown) topic of v.

Power law distributions seem to characterize Web citation frequency; interestingly, similar distributions have been observed for citations in the academic literature, an observation originally due to Lotka [29]. Gilbert [20] presents a probabilistic model supporting Lotka's law. His model is similar in spirit to ours, though different in details and application. The field of bibliometrics [17, 19] is concerned with citation analysis; some of these insights have been applied to the Web as well [28].

Many authors have advanced a view of the Web as a semi-structured database. In particular, LORE [1] and WebSQL [30] use graph-theoretic and relational views of the Web respectively. These views support structured query interfaces to the Web (Lorel [1] and WebSQL [30]) that are inspired by SQL. An advantage of this approach is that many interesting queries can be expressed as simple expressions in the very powerful SQL syntax. The disadvantage of this generality is the associated computational cost, which can be prohibitive until we have query optimizers for Web graph queries similar to those available for relational data. Other examples include W3QS [24], WebQuery [10], Weblog [27], and ParaSite/Squeal [32].

Mendelzon and Wood [31] argue that the traditional SQL query interface to databases is inadequate in its power to specify several interesting structural queries in the Web graph. They propose G^+, a language with greater expressibility than SQL for graph queries.

2. ALGORITHMS

Consider classical database and information retrieval problems such as text search, data mining and classification. The input to these problems is usually a collection of documents. The Web, with its additional structure as a graph, allows the enhancement of existing techniques with graph-theoretic ones. We illustrate this through graph-based solutions to the following problems: topic search, topic enumeration, classification, and crawling.

Given a set of web pages V, let E be the set of directed edges (hyperlinks) among these pages. The pair (V, E) naturally forms an unweighted digraph. For pages $p, q \in V$, we denote a hyperlink from p to q by $p \to q$.

2.1 Topic search

The topic search problem is: Given a search topic (in the form of a query), output high-quality pages for the topic query.

The following is a recurrent phenomenon on the Web: For any particular topic, there tend to be a set of "authoritative" pages focused on the topic, and a set of "hub" pages, each containing links to useful, relevant pages on the topic. This observation motivated the development of the following *HITS* search algorithm [23] and its subsequent variants.

HITS and related algorithms. Given a set of pages V and the interconnections E between them, HITS ranks the pages in V by their quality as hubs, and as authorities. The notions of good hubs and authorities are captured by numerical values whose definitions and update rules are given below.

Each page $p \in V$ has a pair of non-negative weights $\langle x_p, y_p \rangle$ where x_p is the *authority weight* and y_p is the *hub weight*. Before the start of the algorithm, all the x- and y-values are set to 1.

The authority and hub weights are updated as follows. If a page is pointed to by many good hubs, we would like to increase its authority weight; thus for a page p, the value of x_p is updated to be to be the sum of y_q over all pages q that link to p:

$$x_p = \sum_{q \mid q \to p} y_q. \tag{1}$$

In a strictly dual fashion, for a page p, its hub weight is updated via

$$y_p = \sum_{q \mid p \to q} x_q. \tag{2}$$

The algorithm repeats these steps a number of times, at the end of which it generates rankings of the pages by their hub- and authority-scores.

There is a more compact way to write these updates that sheds more light on the mathematical process. Let us number the pages $\{1, 2, \ldots, n\}$ and define their *adjacency matrix* A to be the $n \times n$ matrix whose (i, j)-th entry is equal to 1 if and only if $i \to j$, and is 0 otherwise. Let us also write the set of x-values as a vector $x = (x_1, x_2, \ldots, x_n)^T$, and similarly define $y = (y_1, y_2, \ldots, y_n)^T$. Then the update rule for x can be written as $x \leftarrow A^T y$ and the update rule for y can be written as $y \leftarrow Ax$. Unwinding these one step further, we have

$$x \leftarrow A^T y \leftarrow A^T A x = (A^T A) x \tag{3}$$

and

$$y \leftarrow Ax \leftarrow AA^T y = (AA^T) y. \tag{4}$$

Thus the vector x after multiple iterations is precisely the result of applying *power iteration* to $A^T A$—we multiply our initial iterate by larger and larger powers of $A^T A$—and a standard result in linear algebra tells us that this sequence of iterates, when normalized, converges to the principal eigenvector of $A^T A$. Similarly, the sequence of values for the normalized vector y converges to the principal eigenvector of AA^T. (See the book by Golub and Van Loan [21] for background on eigenvectors and power iteration.)

In fact, power iteration will converge to the principal eigenvector for any "non-degenerate" choice of initial vector—in our case, for the vector of all 1's. Thus, the final x- and y-values are independent of their initial values. This says that the hub and authority weights computed are truly an *intrinsic* feature of the collection of linked pages, not an artifact of the choice of initial weights or the tuning of arbitrary parameters. Intuitively, the pages with large weights represent a very "dense" pattern of linkage, from pages of large hub weight to pages of large authority weight. This

type of structure—a densely linked *community* of thematically related hubs and authorities—will be the motivation underlying Section 2.2 below.

Finally, notice that only the *relative* values of these weights matter not their actual magnitudes. In practice, the relative ordering of hub/authority scores becomes stabile with far fewer iterations than needed to stabilize the actual magnitudes. Typically five iterations of the algorithm is enough to achieve this stability.

In subsequent work [6, 11, 13], the HITS algorithm has been generalized by modifying the entries of A so that they are no longer boolean. These modifications take into account the content of the pages in the base set, the internet domains in which they reside, and so on. Nevertheless, most of these modifications retain the basic power iteration process and the interpretation of hub and authority scores as components of a principal eigenvector, as above.

Implementation. The actual implementation of HITS algorithms (or its variants) consists of a *sampling* step, which constructs a subgraph of several thousand Web pages likely to be rich in relevant authorities and hubs for the particular query topic. To construct this subgraph, the algorithm first uses keyword queries to collect a *root set* of, say, 200 pages from a traditional index-based search engine. This set does not necessarily contain authoritative pages; however, since many of these pages are presumably relevant to the search topic, one can expect some to contain links to good authorities, and others to be linked to by good hubs. The root set is therefore expanded into a *base set* by including all pages that are linked to by pages in the root set, and all pages that link to a page in the root set (up to a designated size cut-off). This follows the intuition that the prominence of authoritative pages is typically due to the endorsements of many relevant pages that are not, in themselves, prominent. We restrict our attention to this base set for the remainder of the algorithm; this set often contains roughly 1000–3000 pages, and hidden among these are a large number of pages that one would subjectively view as authoritative for the search topic.

The sampling step performs one important modification to the subgraph induced by the base set. Links between two pages on the same Web site very often serve a purely navigational function, and typically do not represent conferral of authority. It therefore deletes all such links from the subgraph induced by the base set, and the HITS algorithm is applied to this modified subgraph. As described earlier, the hub and authority values are then used to determine the best pages for the given topic.

2.2 Topic enumeration

The topic enumeration problem is: Given a snapshot of the web, output all *communities* (defined below) in the snapshot.

Recall that a complete bipartite clique $K_{i,j}$ is a graph in which every one of i nodes has an edge directed to each of j nodes (in the following treatment it is simplest to think of the first i nodes as being distinct from the second j; in fact this is not essential to our algorithms). We further define a *bipartite core* $C_{i,j}$ to be a graph on $i+j$ nodes that contains at least one $K_{i,j}$ as a subgraph. The intuition motivating this notion is the following: on any sufficiently well represented topic on the Web, there will (for some appropriate values of i and j) be a bipartite core in the Web graph. Figure 1 illustrates an instance of a $C_{4,3}$ in which the four nodes on the left have hyperlinks to the home pages of three major commercial aircraft manufacturers. Such a subgraph of the

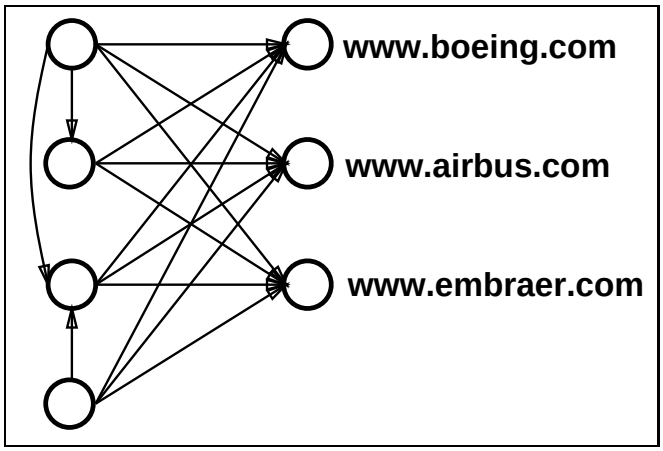

Figure 1: A bipartite core.

Web graph would be suggestive of a "cyber-community" of aficionados of commercial aircraft manufacturers who create hub-like pages like the four on the left side of Figure 1. These pages co-cite the authoritative pages on the right. Loosely speaking, such a community emerges in the Web graph when many (hub) pages link to many of the same (authority) pages. In most cases, the hub pages in such communities may not co-cite *all* the authoritative pages for that topic. Nevertheless, the following weaker hypothesis is compelling: every such community will contain a bipartite core $C_{i,j}$ for non-trivial values of i and j. Turning this around, we could attempt to identify a large fraction of cyber-communities by enumerating all the bipartite cores in the Web for, say $i = j = 3$; we call this process *trawling*. Why these choices of i and j? Might it not be that for such small values of i and j, we discover a number of coincidental co-citations, which do not truly correspond to communities?

In fact, in our experiment [25] we enumerated $C_{i,j}$'s for values of i ranging from 3 to 9, for j ranging from 3 to 20. The results suggest that (i) the Web graph has several hundred thousand such cores, and (ii) it appears that only a minuscule fraction of these are coincidences—the vast majority do in fact correspond to communities with a definite topic focus. Below we give a short description of this experiment, followed by some of the principal findings.

Trawling algorithms. From an algorithmic perspective, the naive "search" algorithm for enumeration which considers every set of $i+j$ pages suffers from two fatal problems. First, the size of the search space is far too large—using the naive algorithm to enumerate all bipartite cores with two Web pages pointing to three pages would require examining approximately 10^{40} possibilities on a graph with 10^8 nodes.

A theoretical question (open as far as we know): does the work on fixed-parameter intractability [16] imply that we cannot—in the worst case—improve on naive enumeration for bipartite cores? Such a result would argue that algorithms that are provably efficient on the Web graph must exploit some feature that distinguishes it from the "bad" inputs for fixed-parameter intractability. Second, and more practically, the algorithm requires random access to edges in the graph, which implies that a large fraction of the graph must effectively reside in main memory to avoid the overhead of seeking a disk on every edge access.

We call our methodology the *elimination-generation paradigm* [26]. An algorithm in this paradigm performs a number of sequential passes over the Web graph, stored as a binary relation. During each pass, the algorithm writes a modified version of the dataset to disk for the next pass. It also collects some metadata which resides in main memory and serves as state during the next pass. Passes over the data are interleaved with sort operations, which change the order in which the data is scanned, and constitute the bulk of the processing cost. We view the sort operations as alternately ordering directed edges by source and by destination, allowing us alternately to consider out-edges and in-edges at each node. During each pass over the data, we interleave *elimination* operations and *generation* operations, which we now detail.

Elimination. There are often easy necessary (though not sufficient) conditions that have to be satisfied in order for a node to participate in a subgraph of interest to us. Consider the example of $C_{4,4}$'s. Any node with in-degree 3 or smaller cannot participate on the right side of a $C_{4,4}$. Thus, edges that are directed into such nodes can be pruned from the graph. Likewise, nodes with out-degree 3 or smaller cannot participate on the left side of a $C_{4,4}$. We refer to these necessary conditions as *elimination filters*.

Generation. Generation is a counterpoint to elimination. Nodes that barely qualify for potential membership in an interesting subgraph can easily be verified either to belong in such a subgraph or not. Consider again the example of a $C_{4,4}$. Let u be a node of in-degree exactly 4. Then, u can belong to a $C_{4,4}$ if and only if the 4 nodes that point to it have a neighborhood intersection of size at least 4. It is possible to test this property relatively cheaply, even if we allow the in-degree to be slightly more than 4. We define a *generation filter* to be a procedure that identifies barely-qualifying nodes, and for all such nodes, either outputs a core or proves that such a core cannot exist. If the test embodied in the generation filter is successful, we have identified a core. Further, regardless of the outcome, the node can be pruned since all potential interesting cores containing it have already been enumerated.

Note that if edges appear in an arbitrary order, it is not clear that the elimination filter can be easily applied. If, however, the edges are sorted by source (resp. destination), it is clear that the out-link (resp. in-link) filter can be applied in a single scan. Details of how this can be implemented with few passes over the data (most of which is resident on disk, and must be streamed through main memory for processing) may be found in [25].

After an elimination/generation pass, the remaining nodes have fewer neighbors than before in the residual graph, which may present new opportunities during the next pass. We can continue to iterate until we do not make significant progress. Depending on the filters, one of two things could happen: (i) we repeatedly remove nodes from the graph until nothing is left, or (ii) after several passes, the benefits of elimination/generation "tail off" as fewer and fewer nodes are eliminated at each phase. In our trawling experiments, the latter phenomenon dominates. However, the number of edges has dropped substantially so a relatively naive post-processing step suffices to output any remaining cores.

Why should such algorithms run fast? We make a number of observations about their behavior:

(i) The in/out-degree of every node never increases during an elimination/generation phase. During each generation test, we either eliminate a node u from further consideration (by developing a proof that it can belong to no core), or we output a subgraph that contains u. Thus, the total work in generation is linear in the size of the Web graph plus the number of cores enumerated, assuming that each generation test runs in constant time.

(ii) In practice, elimination phases rapidly eliminate most nodes in the Web graph. A complete mathematical analysis of iterated elimination is beyond the scope of this paper, and requires a detailed understanding of the kinds of random graph models we propose in Section 4.

2.3 Classification

The supervised classification problem is: Given a set of predefined categories, build a system that (learns from a set of examples and) assigns a given document to one of the categories.

Classification is a hard problem in general and is seems no easier for the Web. The hypertext pages found on the Web pose new problems, however, rarely addressed in the literature on categorizing documents based only on their text. On the Web, for example, pages tend to be short and of widely varying authorship style. Hyperlinks, on the other hand, contain more reliable semantic clues that are lost by a purely term-based categorizer. The challenge is to exploit this information-rich but still noisy link information. Experimentally, it is known that a naive use of terms in the anchortext of links pointing to a document can even *degrade* accuracy.

An approach to this problem is embodied in a classification system[14] which uses robust statistical models (including the Markov Random Field (MRF)) and a *relaxation labeling* technique for better categorization by exploiting link information in a small neighborhood around documents. The intuition is that pages on the same or related topics tend to be linked more frequently than those on unrelated topics and the classification algorithm captures this relationship using a precise statistical model (the MRF) whose parameters are set by the learning process.

The Hyperclass algorithm. The basic idea in this algo-

rithm, called *Hyperclass*, is described below: If p is a page, then instead of considering just p for classification, the algorithm considers the neighborhood $?_p$ around p; here $q \in ?_p$ if and only if $q \to p$ or $p \to q$, although the authors of [14] also consider more general neighborhood functions. The algorithm begins by assigning class labels to all $p \in V$ based purely on the terms in p. Then, for each $p \in V$, its class label is updated based on the terms in p, terms in pages in $?_p$, and the (partial) classification labels of pages in $?_p$. This update is done via robust statistical methods. The iteration, called *relaxation labeling*, is continued until near-convergence.

It has been shown experimentally [14] that hyperlinks used in conjunction with text, if the categories of the linked (to or from) pages are known, can lead to dramatic improvements in categorization accuracy. Even if none of the categories of the linked pages is known, significant improvements can be obtained using relaxation labeling, wherein the category labels of the linked pages and of the page to be categorized are iteratively adjusted until the most probable configuration of class labels is found. Experiments with Hyperclass [14] using pre-classified samples from the US Patent Database (`www.ibm.com/patents`) and Yahoo! cut the patent error rate by half and the Yahoo! (web documents) error rate by two-thirds. The Hyperclass algorithm has been applied in conjunction with the HITS algorithm to construct a *focused crawler*, designed to fetch all the pages on the Web on a given topic with minimal use of resources[12].

3. MEASUREMENTS

In this section we survey empirical observations drawn from a number of recent measurement experiments on the web. The degree distribution results of Section 3.1 are drawn from Kumar et al.[25], Albert et al.[2], and Broder et al.[9]. The enumeration of bipartite graphs in Section 3.2 is from Kumar et al.[25]. Finally, the connected component analysis of Section 3.3 and the diameter analysis of Section 3.4 are from Broder et al.[9].

3.1 Degree distributions

We begin by considering the in-degree and out-degree of nodes in the Web graph. Early work of Kumar et al.[25] contained the first observation that in-degrees follow a power law: The fraction of web pages with in-degree i is proportional to $1/i^x$ for some $x > 1$.

Subsequently, work of Albert et al.[2] and Broder et al.[9] confirmed this result at a variety of scales and times ranging from pages within the Notre Dame University web site, to pages in a 200 million node crawl of the web at large. In all these experiments, the value of the exponent x in the power law is a remarkably consistent 2.1. The results from the largest study, that of Broder et al.[9], are reproduced herein.

Figure 2 is a log-log plot (with the x-axis negated) of the in-degree distribution. The value $x = 2.1$ is derived from the slope of the line providing the best fit to the data in the figure. Figure 3 shows the same results for out-degrees. The best fit line gives a power law with $x = 2.72$, although it is clear from the plot that some concavity exists for smaller out-degrees. The average out-degree is about 7.2.

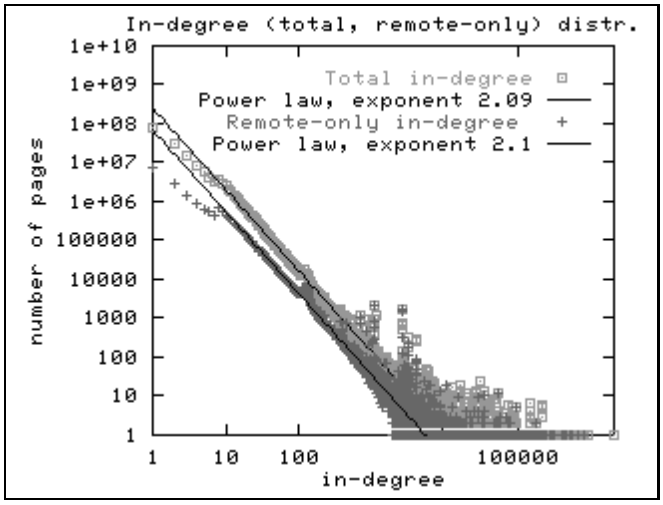

Figure 2: In-degree distribution.

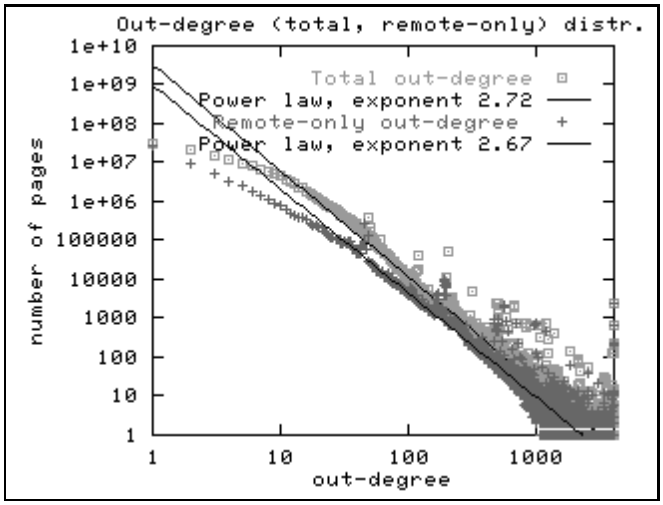

Figure 3: Out-degree distribution.

i	j	Cores	Diverse cores
3	3	89565	38887
3	5	70168	30299
3	7	60614	26800
3	9	53567	24595
4	3	29769	11410
4	5	21598	12626
4	7	17754	10703
4	9	15258	9566
5	3	11438	7015
5	5	8062	4927
5	7	6626	4071
5	9	5684	3547
6	3	4854	2757
6	5	3196	1795
6	7	2549	1425
6	9	2141	1206

Table 1: Number of cores enumerated during the pruning phase of trawling.

3.2 Enumeration of bipartite cores

We turn next to the enumeration of cores $C_{i,j}$. The trawling experiment of Kumar et al.[25] unearthed well over a hundred thousand cores for values of i, j in the range of 3–4. Table 1 gives the number of cores output during the elimination/generation phases; the results of the table show actual number of cores, as well well as the number of *diverse* cores in which each hub page comes from a distinct website. The post-processing step resulted in a smaller number of additional cores, but the cores output during post-processing overlap with one another so the exact counts are not comparable—we do not include them here.

3.3 Connected components

We now report on a connected component analysis from Broder et al.[9]. This analysis includes a number of results about directed and undirected connected components, and also builds an aggregate picture of the structure of the web at large. The results are drawn from an Altavista crawl from May 1999, and have been replicated for a larger crawl from October 1999. All numbers below are from the May crawl, which contains over 200 million pages and 1.5 billion links. We begin with the results for component counts.

A *weakly-connected component* is a set of pages each of which is reachable from any other if hyperlinks may be followed either forwards or backwards. The largest weakly-connected component in this crawl has 186 million nodes, so more than 90% of the crawl lies within this component.

Similarly, a *strongly-connected component* is a set of pages such that for all pairs of pages (u, v) in the set, there exists a directed path from u to v. In web terminology, this means that a surfer can follow hyperlinks to surf from u to v. The largest strongly-connected component has roughly 56 million nodes. The second-largest strongly-connected component has size around 50 thousand, three orders of magnitude smaller. The study also notes that the number of components, either weak or strong, of a given size also follow a power law distribution.

By performing breadth-first searches from a number of random starting nodes following hyperlinks forwards, and then separately backwards, Broder et al. were able to elicit the map of the web depicted in Figure 4. They refer to this picture as a *bowtie*, and describe the different regions of the bowtie as follows:

The "knot" of the bowtie, called the *SCC*, represents the single giant strongly-connected component of size around 56 million. The "left side" of the bowtie represents about 44 million pages called *IN*, defined to be all pages not in the SCC, but from which a path exists to some node of the SCC. Since a path to some node of the SCC implies a path to every node of the SCC, a surfer beginning at some page of IN can reach any page of the SCC. The set IN can be thought of as "new pages" that link to interesting destinations on the web, but which have not yet been discovered by the core of the web and are therefore not reachable from the SCC. (If a page of IN became reachable from the SCC, it would become part of the SCC.)

Similarly, another large set of approximately 44 million pages make up the "right side" of the bowtie. This set is called *OUT*, and has the property that any page of OUT can be reached from any page of the SCC by following hyperlinks, but no page of the SCC can be reached from a page of OUT by following hyperlinks. A surfer beginning at one of these pages will quickly get stuck and be unable to explore further. One may think of these pages as corporate internets which are well-known, but whose links point only internally.

Thus, in this model it is always possible to surf the bowtie from left to right, but not the other way: from the pages of IN a surfer can reach SCC, and can then continue on to the pages of OUT, but motion in the other direction is not possible by clicking on links.

Finally, there is a fourth region called the *TENDRILS*, consisting of pages that do not link to the knot, and which are not reachable from the knot. These pages may be thought of as possessing the disadvantages of IN and OUT: the web has not yet discovered these pages, and these pages do not contain interesting links back to better-known regions of the web. Although these pages do not fit the image of traditional web pages, they nonetheless make up a significant fraction of the web—once again, there are approximately 44 million such pages.

3.4 Measures of diameter

The bowtie of Figure 4, in conjunction with a deeper analysis of the pages outside the SCC, reveals an unexpected property of web connectivity: for most pages u and v, there does not exist a path from u to v. More precisely, if u lies in IN $\cup$ SCC, and v lies in SCC $\cup$ OUT then a path exists, but if not then a path will almost certainly not exist. The probability that u lies in IN $\cup$ SCC is about 1/2, and the probability that v lies in SCC $\cup$ OUT is likewise about 1/2, so the probability that these two independent events hold simultaneously is about 1/4. Thus, for around 75% of pages u and v, no path exists.

Recent results of Albert et al.[2] predict that for most pairs of web pages u and v the directed distance (following hyper-

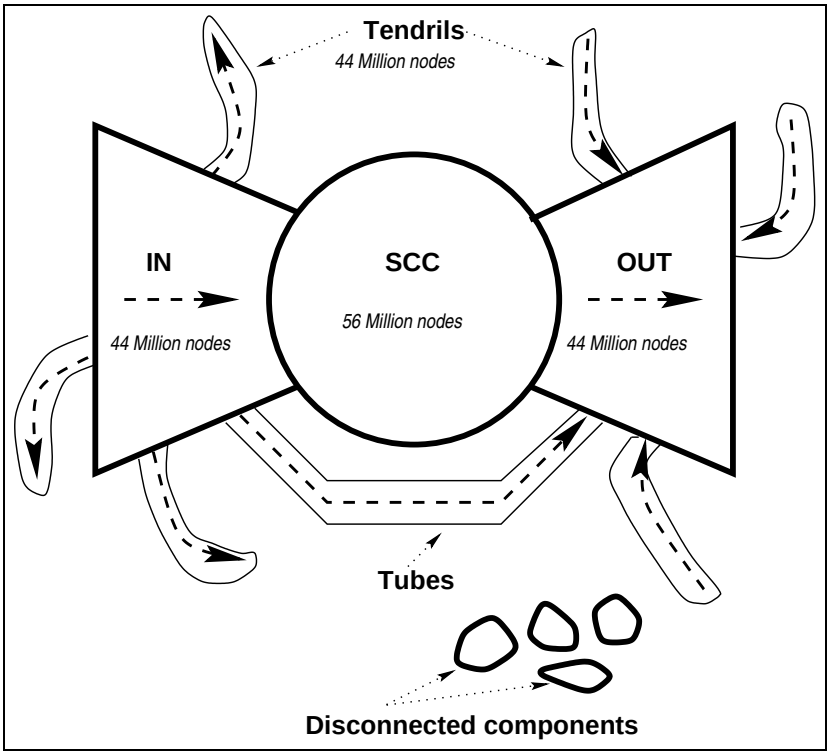

Figure 4: The web as a bowtie. SCC is a giant strongly connected component. IN consists of pages with paths to SCC, but no path from SCC. OUT consists of pages with paths from SCC, but no path to SCC. TENDRILS consists of pages that cannot surf to SCC, and which cannot be reached by surfing from SCC.

Edge type	In-links	Out-links	Undirected
Avg. Conn. Dist.	16.12	16.18	6.83

Table 2: Average distances on the web, conditioned on existence of a finite path.

links) from u to v is about 19, a so-called *small world phenomenon*. However, actual measurement reveals a different picture: most pairs of pages in fact have infinite directed distance.

To salvage the small world view, we can instead ask for the length of paths separating u and v during the 25% of the time in which there exists a path between them. Formally, we define the *average connected distance* under a certain definition of path (for instance, following hyperlinks, or following hyperlinks but in either direction) to be the average length of the path over all pairs for which the length is finite. The average connected distance is given in Table 2.

4. MODEL

In this section we lay the foundation for a class of plausible random graph models in which we can hope to establish many of our observations about the local structure of the Web graph. We begin by presenting the broad areas of application for Web graph models, to motivate the discussion.

(i) Web graph models can be analytical tools. Many problems we wish to address on the web are computationally difficult for general graphs; however, with an accurate model of the Web, specific algorithms could be rigorously shown to work for these problems within the model. They could also be simulated under the model to determine their scalability.

(ii) A good model can be an explanatory tool. If a simple model of content creation generates observed local and global structure, then it is not necessary to postulate more complex mechanisms for the evolution of these structures.

(iii) A good model can act as a predictive tool. The model can suggest unexpected properties of today's Web that we can then verify and exploit. Similarly, the model can also suggest properties we should expect to emerge in tomorrow's Web.

In Section 3 we presented a number of measurements on the Web graph. To motivate the need for new graph models tailored to the Web, we briefly point out the shortcomings of the traditional random graph model $\mathcal{G}_{n,p}$ [8]. (The $\mathcal{G}_{n,p}$ model describes a random graph on n nodes where every possible edge is included with probability p, independently of other edges.) First, note that $\mathcal{G}_{n,p}$ does not generate extremely popular pages such as Yahoo!. Instead, the tails of the degree distribution fall off exponentially, allowing us to conclude that it is unlikely to find a page with $\omega(\log n)$ in-links when np, the average number of out-links, is $O(1)$. More generally, the in-degree distribution in $\mathcal{G}_{n,p}$ graphs is binomial, not one that obeys a power law of the type reported in Section 3. Second, and more importantly, in $\mathcal{G}_{n,p}$ the links to or from a particular page are independent, while

on the web a page that has a link to `www.campmor.com` is much more likely to contain other links to camping-related material than a random page. A consequence of this is the fact that $\mathcal{G}_{n,p}$ does not explain well the number of cores $C_{i,j}$ (as reported in Section 3); for example, the expected number of cores $C_{i,j}$ in $\mathcal{G}_{n,p}$ with $np = 7.2$ (the average out-degree of the web graph) is $\binom{n}{i}\binom{n}{j}\left(\frac{7.2}{n}\right)^{ij}$, which is negligible for $ij > i + j$.

Thus, a graph model for the web should manifest the following three properties:

(i) (The rich get richer) New links should point more often to pages with higher in-degree than to pages with lower in-degree.

(ii) (Correlated out-links) Given a link $h = (u, v)$, the destinations of other links out of u should reveal information about v, the destination of h.

(iii) (Correlated in-links) Given a link $h = (u, v)$, the sources of other links with destination v should reveal information about u, the source of h.

Having established some properties that the model should satisfy, we also enumerate some high-level goals in the design:

(i) It should be easy to describe and feel natural.

(ii) Graph structures should reflect measurements of the Web graph.

(iii) Topics should not be planted—they should emerge naturally. This is important because

1. It is extremely difficult to characterize the set of topics on the Web; thus it would be useful to draw statistical conclusions without such a characterization.

2. The set, and even the nature, of topics reflected in Web content is highly dynamic. Thus, any time-dependent model of topics would need to include this evolution over time—a daunting task.

4.1 Random copying

Our model is motivated by the following intuition:

- *some* authors will note an interesting (but hitherto unknown on the web) commonality between certain pages, and will link to pages exhibiting this commonality. The first person to create a resource list about fishing would be an example of such an author, as would the first person to create a resource list specifically about fly fishing.

- *most* authors, on the other hand, will be interested in certain already-represented topics, and will collect together links to pages about these topics.

In our model, an author interested in generating links to topics already represented on the web will do so by discovering existing resource lists about the topics, then linking to pages of particular interest within those resource lists. We refer to this authoring mechanism as *copying* because the new page contains a subset of links on some existing page.

However, we must present a few caveats. First, despite the term "copying," it is not necessary that the new author physically copy links. We assume simply that the new author will link to pages within the topic, and therefore to pages that are linked-to by some existing resource list. The mechanism is thus analytical, but not behavioral.

On a related note, this process is *not* designed as the basis of a user model; rather, it is a local link-creation procedure which in aggregate causes the emergence of web-like structure and properties. Topics are created as follows. First, a few users create disconnected pages about the topic without knowledge of one another—at this point, no "community" exists around the topic. Then, interested authors begin to link to pages within the topic, creating topical resource lists that will then help other interested parties to find the topic. Eventually, while the web as a whole will remain "globally sparse," a "locally dense" subgraph will emerge around the topic of interest.

We propose random copying as a simple yet effective mechanism for generating power law degree distributions and link correlations similar to those on the web, mirroring the first-order effects of actual web community dynamics. We now present a family of graph models based on random copying, and present some theoretical results for some simple models within this family.

4.2 A class of graph models

Traditional $\mathcal{G}_{n,p}$ graphs are static in the sense that the number of nodes is fixed at the beginning of the process, and does not change. The graphs in our model are evolving in that both nodes and edges appear over time; some of these nodes/edges may later disappear. As a means for presenting our models, we adopt the following terminology. A model is characterized by four stochastic processes responsible for creation and deletion of vertices and edges: $\mathcal{C}_v$, $\mathcal{C}_e$, $\mathcal{D}_v$, and $\mathcal{D}_e$. Each is a discrete-time process that may base its decisions on the time-step and the current graph.

As a simple example, consider casting $\mathcal{G}_{n,p}$ in our model. $\mathcal{C}_v$ creates n vertices at time 0, $\mathcal{C}_e$ creates each edge with uniform probability p, and both deletion processes are empty.

As a more realistic example that generates web-like graphs, let $\mathcal{C}_v$ at time t create a node with probability $\alpha_c(t)$ independent of the current graph, and let $\mathcal{D}_v$ remove some appropriately-chosen page and all incident edges with probability $\alpha_d(t)$. These probabilities would be chosen to mirror the growth rate of the web, and the half-life of pages respectively. The corresponding edge processes will incorporate random copying in order to generate web-like graphs.

An example $\mathcal{C}_e$ is the following. At each time-step, we choose to add edges to all the newly-arrived pages, and also to some existing pages via an update procedure, modeling the process of page modification on the web. For each chosen page, we randomly choose a number of edges k to add to that page. With some fixed probability β, we add k edges to destinations chosen uniformly at random. With the remaining

probability, we add k edges by copying: we choose a page v from some distribution and *copy* k randomly-chosen edges from v to the current page. If v contains fewer than k pages, we copy some edges from v and then choose another page to copy from, iterating until we have copied the requisite number of edges.

Similarly, a simple example of $\mathcal{D}_e$ might at time t choose with probability $\delta(t)$ to delete an edge chosen from some distribution.

We now turn our attention to a particular instantiation of this model that is theoretically tractable.

4.3 A simple model

We present with a simple special case of our family of models; this special case cleanly illustrates that the power law can be derived from a copying process. In this special case, nodes are never deleted. At each step we create a new node with a single edge emanating from it. Let u be a page chosen uniformly at random from the pages in existence before this step. With probability α, the only parameter of the model, the new edge points to u. With the remaining probability, the new edge points to the destination of u's (sole) out-link; the new node attains its edge by copying.

We now state a result that the in-degree distribution of nodes in this model follows a power law. More specifically, we show that the fraction of pages with in-degree i is asymptotic to $1/i^x$ for some $x > 0$. Let $p_{i,t}$ be the fraction of nodes at time t with in-degree i.

THEOREM 4.1.

$$(i) \quad \forall i, \quad \lim_{t\to\infty} E[p_{i,t}] \triangleq \hat{p}(i) \quad \text{exists} ,$$
$$(ii) \quad \lim_{i\to\infty} i^{\frac{2-\alpha}{1-\alpha}} \hat{p}(i) = \frac{1}{1+\alpha}.$$

5. CONCLUSION

Our work raises a number of areas for further work:

(i) How can we annotate and organize the communities discovered by the trawling process of Section 2.2?

(ii) Bipartite cores are not necessarily the only subgraph enumeration problems that are interesting in the setting of the Web graph. The subgraphs corresponding to Webrings (which look like bidirectional stars, in which there is a central page with links to and from a number of "spoke" pages), cliques, and directed trees are other interesting structures for enumeration. How does one devise general paradigms for such enumeration problems? Kumar et al.[26] describe initial approaches to this problem.

(iii) What are the properties and evolution of random graphs generated by specific versions of our models in Section 4? This would be the analog of the study of traditional random graph models such as $\mathcal{G}_{n,p}$.

(iv) How do we devise and analyze algorithms that are efficient on such graphs? Again, this study has an analog with traditional random graph models.

(v) What can we infer about the distributed sociological process of creating content on the Web?

(vi) What finer structure can we determine for the map of the Web graph (Figure 4) in terms of domain distributions, pages that tend to be indexed in search engines, and so on?

6. REFERENCES

[1] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel Query language for semistructured data. *Intl. J. on Digital Libraries,* 1(1):68–88, 1997.

[2] R. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the World Wide Web. *Nature* 401:130–131, 1999.

[3] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. *Proc. ACM Symp. on Theory of Computing*, 2000. To appear.

[4] G. O. Arocena, A. O. Mendelzon, and G. A. Mihaila. Applications of a Web query language. *Proc. 6th WWW Conf.*, 1997.

[5] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public Web search engines. *Proc. 7th WWW Conf.,* 1998.

[6] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. *Proc. ACM SIGIR*, 1998.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Proc. 7th WWW Conf.*, 1998.

[8] B. Bollobás. *Random Graphs.* Academic Press, 1985.

[9] A. Z. Broder, S. R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. *Proc. 9th WWW Conf.*, 2000. To appear.

[10] J. Carrière and R. Kazman. WebQuery: Searching and visualizing the Web through connectivity. *Proc. 6th WWW Conf.*, 1997.

[11] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan. Automatic resource compilation by analyzing hyperlink structure and associated text. *Proc. 7th WWW Conf.*, 1998.

[12] S. Chakrabarti, B. Dom, and M. van den Berg. Focused crawling: A new approach for topic-specific resource discovery. *Proc. 8th WWW Conf.*, 1999.

[13] S. Chakrabarti, B. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Experiments in topic distillation. *SIGIR workshop on Hypertext IR*, 1998.

[14] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext classification using hyperlinks. *Proc. ACM SIGMOD*, 1998.

[15] H. T. Davis. *The Analysis of Economic Time Series.* Principia Press, 1941.

[16] R. Downey and M. Fellows. Parametrized computational feasibility. In *Feasible Mathematics II*, P. Clote and J. Remmel, eds., Birkhauser, 1994.

[17] L. Egghe and R. Rousseau. *Introduction to Informetrics*. Elsevier, 1990.

[18] D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the World Wide Web: A survey. *SIGMOD Record*, 27(3): 59–74, 1998.

[19] E. Garfield. Citation analysis as a tool in journal evaluation. *Science*, 178:471–479, 1972.

[20] N. Gilbert. A simulation of the structure of academic science. *Sociological Research Online*, 2(2), 1997.

[21] G. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1989.

[22] M. M. Kessler. Bibliographic coupling between scientific papers. *American Documentation*, 14:10–25, 1963.

[23] J. Kleinberg. Authoritative sources in a hyperlinked environment. *J. of the ACM*, 1999, to appear. Also appears as IBM Research Report RJ 10076(91892) May 1997.

[24] D. Konopnicki and O. Shmueli. Information gathering on the World Wide Web: the W3QL query language and the W3QS system. *Trans. on Database Systems*, 1998.

[25] S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling emerging cyber-communities automatically. *Proc. 8th WWW Conf.*, 1999.

[26] S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. *Proc. VLDB*, 1999.

[27] L. V. S. Lakshmanan, F. Sadri, and I. N. Subramanian. A declarative approach to querying and restructuring the World Wide Web. *Post-ICDE Workshop on RIDE*, 1996.

[28] R. Larson. Bibliometrics of the World Wide Web: An exploratory analysis of the intellectual structure of cyberspace. *Ann. Meeting of the American Soc. Info. Sci.*, 1996.

[29] A. J. Lotka. The frequency distribution of scientific productivity. *J. of the Washington Acad. of Sci.*, 16:317, 1926.

[30] A. Mendelzon, G. Mihaila, and T. Milo. Querying the World Wide Web. *J. of Digital Libraries*, 1(1):68–88, 1997.

[31] A. Mendelzon and P. Wood. Finding regular simple paths in graph databases. *SIAM J. Comp.*, 24(6):1235–1258, 1995.

[32] E. Spertus. ParaSite: Mining structural information on the Web. *Proc. 6th WWW Conf.*, 1997.

[33] G. K. Zipf. Human behavior and the principle of least effort. *New York: Hafner*, 1949.