A Hyperlink-directed Data Dissemination Scheme Suite[*]

Sooyeon Kim[*], Jihyuk Choi[†], Yanghee Choi[*]
[*] School of Computer Science and Engineering, Seoul National University
[†] Electronics and Telecommunications Research Institute

Abstract

In data dissemination to a large client population using a broadcast medium, data items are inevitably serialized over the medium and thus a scheduling strategy that determines the data transfer sequences should be developed. The resulted data provision intended to satisfy multiple clients at the same time insists that client local storage should be utilized as data cache to compensate the gap between the broadcast schedule and a single user's behavior. This paper suggests that the application characteristics should be examined and exploited when designing data dissemination schemes; Dominating data broadcast applications depend on hyperlinks for movement between information pages, which definitely restricts the sequence of data requests. Inspired by this observation, we propose a scheduling mechanism that deals with users of different interests in a fair manner but does not sacrifice the response time, and a cache management that significantly lowers the response time by raising cache hit ratio.

Keywords: *Data Dissemination, Scheduling, Cache Management, Hyperlink-based Applications*

1. Introduction

Advances in embedded system design and networking technologies have encouraged embedded systems such as cell phones, set-top boxes (STB), and personal digital assistants (PDA) to be ready for information retrieval through network interfaces. Most of such information retrieval applications rely on data push via a broadcast medium, where information pages are sent from the server to the clients without any explicit requests from the clients. This push-based data dissemination is justified by the following properties of current systems:

[*] This work was supported in part by the Brain Korea 21 project of Ministry of Education and in part by the National Research Laboratory project of Ministry of Science and Technology, 2002, Korea.

- Scarcity of system resources: Server capacity/population might not be sufficient to serve every user request individually. Network bandwidth might be expensive or limited by some regulations (e.g., governmental regulations on frequency usage) and consequently scarce to support on-demand data provision.
- Similarity in user interests: Audience who are watching the same television program or drivers whose destinations are located close together might have common interests.
- Shared medium access: Networking via cable, satellite, and other wireless medium access embraces broadcast capability in nature.

Even the broadcast feature, however, cannot prevent the data items from being serialized over a transport medium, and thus a scheduling algorithm that determines what and when to broadcast should be developed. We claim that the scheduling scheme should consider following criteria:

- Fairness: The scheduling algorithm should not discriminate against or in favor of clients, because their behaviors are different from or in accord with the server's estimation reflected on the broadcast schedule.
- Efficiency: It is desirable that the scheduling algorithm contributes to the response time reduction, in both terms of mean and variance, as pointed out by [1].

The idea to satisfy multiple users with a single data feed implies that the system would not promptly respond to every individual data request, which insists that clients should utilize available local storage to keep data that might be requested in the near future, to avoid substantial and possibly fluctuating latency. In such cases the local storage can be regarded as cache [2], in a sense that the latency incurred by local storage access is expected to be smaller than that introduced by waiting for the desired data to appear on the broadcast channel. However, we argue that the cache management for data dissemination applications should be distinguished from traditional caching approaches due to following reasons:

- In data dissemination applications, it is expected that users do not repeat requests to the same data items. Therefore, response time gain is hardly achieved by caching previously accessed data items. Instead, data prefetching in advance of user requests is preferable.
- A large portion of data items are dynamically generated in accord with one-time events or short-lived situations such as currently on-air television programs and traffic jams caused by car accidents, which imposes that future access patterns might not be speculated from the past ones.
- The use of shared broadcast medium alleviates the concern suggested in [3] that prefetching could increase network load.

With considerations like above, we propose a scheduling algorithm and a cache management scheme throughout the remainder of this paper. Section 2 explains the underlying environmental assumptions. Section 3 and 4 introduce the scheduling algorithm and the cache management we propose. Then some experimental results are shown in Section 5. Section 6 presents related work, and finally Section 7 concludes this paper.

2. Environmental Model

We consider an environment where a large number of *data items* are cyclically disseminated on a broadcast channel by the server, as also assumed by other researchers [4, 5, 6] and practically incorporated into some authoritative data service protocols for digital television broadcast [7]. A data item is presumed to be a set of data piles that might be simultaneously requested by an application; For instance, in a hypertext markup language (HTML) based application, an item refers to the whole set of files (e.g., an HTML file, JPEG files for still images, etc.) required to display an HTML page flawlessly, and in that sense an item is also termed a *page*. Data items are heterogeneous in terms of size.

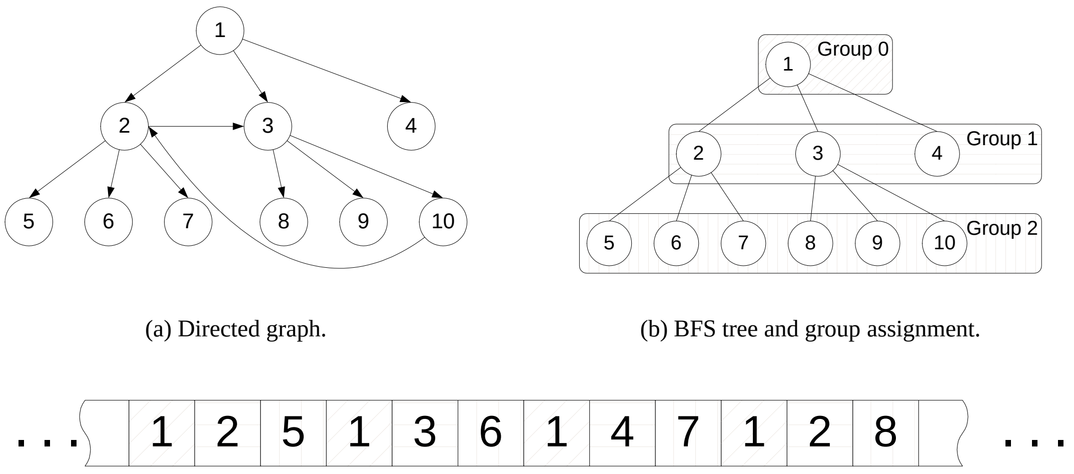
There are a large number of clients that listen to the broadcast channel but do not have any transmission capabilities to send data requests back to the servers. Some client systems have local storage (also referred to as *cache*) that can be used to prefetch broadcast data preceding user requests, but others does not. Having a request for a data item, clients firstly check if their own cache has a copy for the requested item; If yes, they immediately deliver the copy and in such cases the latency is assumed to be ignorable; Otherwise (including no-cache cases), they should wait until the desired item appears on the broadcast channel.

We pay attention to the fact that predominant data dissemination applications in television broadcasts and personal mobile communications are based on hyperlinks[1]. Moreover, in most cases, hyperlinks are the only way to access data items, that is, direct accesses to items by specifying item identifiers are not supported. Such a hyperlink-based application can be abstracted as a directed graph $G(V, E)$ with a distinguished source vertex s which represents the first page (or *start page*) that is automatically displayed to present the top menu for further browsing when a user invokes the application[2]. We call such a graph the *access flow graph* meaning that the graph represents the restricted sequences of user access.

Since we assume that there are no ways to access nodes other than hyperlinks, at an arbitrary time there are only a few *eligible nodes* among all the nodes in the access flow graph, which are directly connected via hyperlinks from the current page. The reference probability of the eligible nodes, which summarizes the user behavior model, follows the Zipf distribution [8] that is frequently referred to as a realistic model for Web user behavior [9, 10] and adopted by other researchers in the field of data broadcast [4, 5, 11]. Suppose that the number of eligible nodes is denoted by M and that the eligible nodes are numbered $1, 2, ..., M$, the access probability of eligible node i

[1]Typical mobile phone applications depend on menu systems for branching between pages, where several menu listings are presented per page and a user is supposed to select one of them to browse another page. Such menu listings can be regarded as hyperlinks.

[2]To simplify graph representation, we assume that hyperlinks to the start page (home) and the previous page (back) are omitted in $G(V, E)$, while they are reflected in the user behaviors of our simulation.



(a) Directed graph.

(b) BFS tree and group assignment.

(c) Broadcast schedule.

Figure 1. Breadth-first search based group assignment and scheduling.

following the Zipf distribution with an *access skew coefficient* θ is calculated by

$$p_i = \frac{(1/i)^\theta}{\sum_{i=1}^{M}(1/i)^\theta}. \tag{1}$$

Equation 1 states that the access probability of eligible node i is proportional to $(1/i)^\theta$, and that accordingly, access patterns become increasingly skewed as the absolute value of θ increases.

3. Scheduling

We claim that, when a schedule cannot satisfy everybody, ironically the best is to satisfy nobody. Instead of speculating user access patterns, the proposed scheduling scheme investigates the application characteristics and estimates which type of data items would have more access requests. Suppose that page retrieval sequences are abstracted by the access flow graph $G(V, E)$ as previously mentioned in Section 2, there is no option for a user to browse a page at a distance (fewest number of edges) of $n + 1 (n = 0, 1, ...)$ from the first page, without visiting a page at a distance of n. Therefore, we argue that pages in the vicinity of the start page get more access requests, and according to previous suggestions [4, 5], our scheme intends to schedule those of smaller distance from the start page with higher broadcast frequency.

Given the access flow graph $G(V, E)$, the proposed scheduling algorithm performs the following steps in series:

1. Do breadth-first search (BFS) [12] to $G(V, E)$ to compute the distance from the start page s to all pages.

```
1 if a copy of the newly arrived page already resides in the cache
2   then no change to the cache
3   else if the cache has room for the newly arrived
4         then fetch the newly arrived
5   else if the max distance of in-cache pages from the current page is
6         greater than that of the new one
7         then while cache does not have sufficient room for the new one
8                    replace the max-distance page in the cache
9              fetch the new page
```

Figure 2. Connectivity-aware prefetching.

2. Every page whose distance from s is d $(d = 0, 1, ...)$ is assigned to group d. The number of groups $N = max(d)$.

3. Items are scheduled in a two-dimensional round robin manner. Group $(i + 1)\ mod\ N$ $(i = 0, 1, ...)$ is eligible to schedule a page after group i schedule a page, and pages in an eligible group are also scheduled in a round-robin manner.

Figure 1 presents an example. When the application structure is summarized as a graph shown in Figure 1(a), Figure 1(b) is the resulted BFS tree and the group assignment after a breadth-first search, and finally the broadcast schedule would be like Figure 1(c). We call this scheduling scheme *breadth-first search based group assignment (BGA)*.

4. Cache Management

The proposed cache management called *connectivity-aware prefetching (CAP)* tries to keep the pages that can be reached by fewer number of branching, running the algorithm described in Figure 2 every time a data item is newly received. Although this prefetching technique requires clients to find the distance from the current page for every new page arrival, it can be easily resolved by a server-side support that appropriately determines page identifiers as described in the next paragraph.

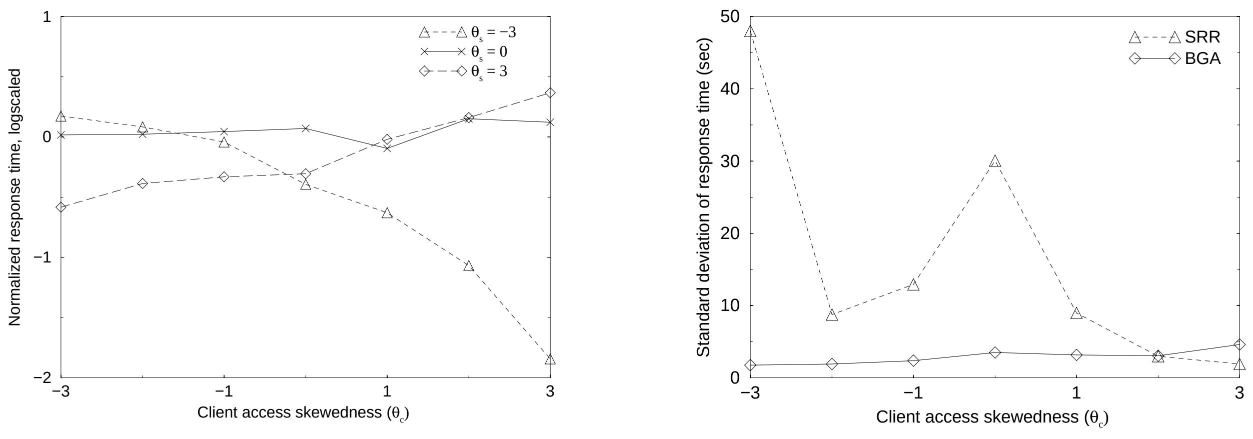
We modify the BFS in traditional graph theory [12] such that it performs page identifier assignment while searching the access flow tree graph. Note that the scheduling proposed in Section 3 includes a BFS phase, which means that this identifier assignment does not increased the algorithmic complexity. Figure 3 presents the modified breadth-first search in detail. With such an identifier assignment, the distance of a page whose identifier is q from the current page whose identifier is p can be computed by a function $dist(q, p)$ presented in Figure 4. Eventually the identifier assignment enables straightforward distance computations to the clients, with neither a priori knowledge on the global topology (access flow graph) nor analysis of hyperlinks embedded in each page.

```
BFS $(G, s)$
1  **for** each vertex $u \in V[G] - \{s\}$
2      **do** $color[u] \leftarrow$ WHITE
3         $newcid[u] \leftarrow 0$
4         $nthid[u] \leftarrow 0$
5         $d[u] \leftarrow \infty$
6         $\pi[u] \leftarrow$ NIL
7         **for** each $1 \leq i \leq$ MAXNOIDS
8             $id[u][i] \leftarrow -1$
9  $color[s] \leftarrow$ GRAY
10 $newcid[s] \leftarrow 0$
11 $nthid[s] \leftarrow 1$
12 $id[s][1] \leftarrow 0$
13 $d[s] \leftarrow 0$
14 $\pi[s] \leftarrow$ NIL
15 $Q \leftarrow \{s\}$
16 **while** $Q \neq 0$
17     **do** $u \leftarrow head[Q]$
18        **for** each $v \in Adj[u]$
19            **do** $newcid[u] \leftarrow newcid[u] + 1$
20               $nthid[v] \leftarrow nthid[v] + 1$
21               $id[v][nthid[v]] \leftarrow id[u][1] * 10 + newcid[u]$
22               **if** $color[v] = $ WHITE
23                  **then** $d[v] \leftarrow d[u] + 1$
24                       $\pi[v] \leftarrow u$
25                       ENQUEUE $(Q, v)$
26        DEQUEUE($Q$)
27        $color[u] \leftarrow$ BLACK
```

Figure 3. Modified breadth-first search for page identifier assignment.

```
dist $(p, q)$
1 **if** $p = 0$
2   **then** return $(\lfloor log_{10} q \rfloor + 1)$
3   **else** $l = \lfloor log_{10} p \rfloor$
4        $m = \lfloor log_{10} q \rfloor$
5        **while** $l \geq 0$ and $m \geq 0$
6            **if** $\lfloor p/10^l \rfloor \neq \lfloor q/10^m \rfloor$ **then** break
7            $l \leftarrow l - 1$
8            $m \leftarrow m - 1$
9        return $min(l + m + 2, \lfloor log_{10} q \rfloor + 2)$
```

Figure 4. Distance between two pages.



(a) Response time. (b) Standard deviation ($\theta_\mathbf{s} = \mathbf{3}$).

Figure 5. Response time and standard deviation of response time versus client access skews.

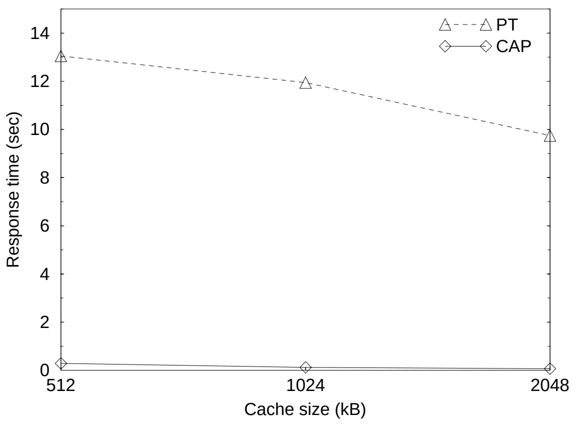
5. Results

We present some experimental results for the proposed scheduling algorithm and cache management. Throughout this section, data item sizes follow a normal distribution $N(86.5 \text{ kB}, 10^2)$ derived from our previous work that implemented data dissemination applications for interactive televisions. Access flow graphs are randomly generated with two parameters: the *maximum distance from the start page H* (i.e., the height of the BFS tree) and the *maximum number of outgoing edges C* (i.e., the maximum number of children in BFS tree). Reading time per page (i.e., the amount of time between two consecutive data item requests) follows Inverse Gaussian distribution with mean 30 seconds, median 7, standard deviation 80, which approximates to the heavy tailed distribution shown in a Web characteristic study [10].

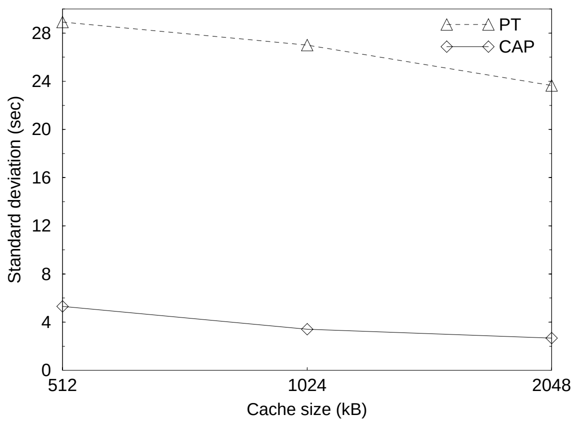
5.1 Scheduling

In order to evaluate the proposed scheduling scheme BGA, we built a simulation environment with two types of scheduling: BGA and an online scheduling algorithm (hereafter abbreviated to *SRR* after their theoretical basis *square-root rule*) proposed in [5], in which items of higher access probability and smaller size are scheduled with higher frequency.

We use two separate Zipf coefficients θ_s and θ_c to emulate orthogonal behaviors of the server and a client, each of which is varied in a range of $[-3, 3]$. Similar or equal θ_s and θ_c indicate a good estimation on the client's access pattern. The results to be shown in this section are statistically obtained from a simulation with 20 randomly generated access flow graphs ($H = 3$, $C = 9$) and 100 wandering-in-the-graph clients for each graph.



(a) Response time.



(b) Standard deviation.

Figure 6. Response time and standard deviation of response time versus cache size.

Figure 5(a) depicts the logscaled response time of BGA normalized by that of SRR versus client access skew coefficient (θ_c). More specifically, the y-values are determined by $log_{10} R_{BGA}/R_{SRR}$, where R_x denotes the expected response time (the average amount of elapsed time between a page request and the corresponding data service completed) with scheduling algorithm x. Since the y-values are plotted on a log scale, values less than zero represent the advantage of BGA. As expected, we observe that BGA is more effective when the access probabilities (described by θ_s) do not estimate well the real access skew of the client (characterized by θ_c), while being competitive even when the estimation fits well.

Figure 5(b) shows the standard deviations of response times for seven different access skew patterns of clients when the server access skew coefficient is three. For an intuitive presentation, here the response time is converted into seconds, with an assumption that the bandwidth of the broadcast channel is 2 Mbps. The results show that BGA is more stable than SRR, no matter how much a client's actual data request pattern differs from the server's estimation.

5.2 Cache Management

The proposed cache management scheme CAP is compared with a prefetching algorithm called *PT* introduced in [11]. Figure 6(a) and Figure 6(b) show the response time and the standard deviation of response time, for cache sizes 512, 1024, and 2048 kB and a randomly generated access flow graph parameterized by $H = 4$ and $C = 9$. The bandwidth of the broadcast channel is also assumed as 2 Mbps. The background of remarkable excellence is explained by Figure 7; the overall hit ratio is as high as 98.4% with a 512-kB cache which can hold only 2.6% of the whole set of broadcast data.

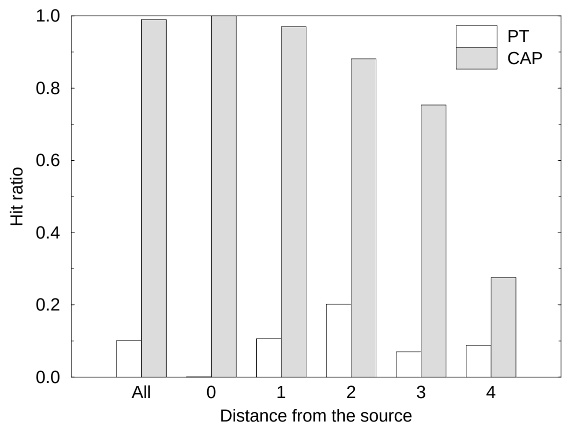


Figure 7. Cache hit ratio.

6. Related Work

Data dissemination to a fairly large number of clients has been extensively discussed in the literature [1, 4, 5, 6, 11, 13]. *Acharya et al.* introduce the multi-disk concept, where data items are divided into several groups of different broadcast frequencies reflecting the relative importance of the items, and the instances of each item are almost equally spaced by scheduling items of each group in a round-robin manner [4], as theoretically advocated by *Vaidya et al.* [5]. Regarding the response time, *Jiang et al.* point out that not only the mean but also the variance is of importance [1].

Although such prominent issues are dealt with by previous work, most of them have focused on estimating the access probability of data items. However, prospective applications would be associated with television broadcasts or personal mobile communications where it is expected that disseminated contents are created with a high degree of dynamics being affected by temporal and spatial factors surrounding users, and accordingly a statistically correct estimation on user behavior is hardly achieved. This work introduces a suite of scheduling and cache management, which exploits the targeted application characteristics that hyperlinks dominate user movement between data items and definitely restrict the sequence of data requests. Some work on teletext systems like [14] and studies on the relationship between user access patterns and hypertext [15] encourage us to investigate application characteristics. Also, the prefetching technique introduced in [11] inspires us to attempt to increase cache hit ratio by avoiding cold start cache misses [2].

7. Conclusion

In this paper we present a scheduling algorithm and a cache management technique for data dissemination to a quite large number of users. The proposed scheduling *breadth-first search based group assignment (BGA)* impartially deals with the users of different access patterns and guarantees less fluctuation in terms of response time. The cache management scheme *connectivity-aware prefetching (CAP)* exploits the application characteristics that the data access sequence is strictly restricted by hyperlinks, such that the items probably requested in the near future can be anticipated even when the access probabilities are not easily estimated.

References

[1] S. Jiang and N. H. Vaidya. Response time in data broadcast systems: Mean, variance, and trade-off. In *Proc. of Workshop on Satellite-based Information Services (WOSBIS)*, Dallas, TX, October 1998.

[2] J. L. Hennessy and D. A. Petterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, second edition, 1996.

[3] M. Crovella and P. Barford. The network effects of prefetching. In *Proc. of IEEE INFOCOM '98*, San Francisco, March 1998.

[4] S. Acharya, M. Franklin, and S. Zdonik. Dissemination-based data delivery using broadcast disk. *IEEE Personal Communications*, 2(6):50–60, December 1995.

[5] N. H. Vaidya and S. Hameed. Data broadcast in asymmetric wireless environment. In *Proc. of Workshop on Satellite-based Information Services (WOSBIS)*, Rye, NY, November 1996.

[6] C. Kenyon and N. Schabanel. The data broadcast problem with non-uniform transmission times. In *Proc. of Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 547–556, January 1999.

[7] Digital Video Broadcasting (DVB). *Digital Video Broadcasting (DVB); DVB specification for data broadcasting*. ETSI EN 301 192, June 1999.

[8] D. Knuth. *The Art of Computer Programming*, volume 2. Addison Wesley, 1981.

[9] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. Web caching and Zipf-like distributions: Evidence and implications. In *Proc. of IEEE INFOCOM '99*, New York, NY, March 1999.

[10] J. E. Pitkow. Summary of WWW characterization. In *Proc. of the Seventh International World Wide Web Conference*, Brisbane, Australia, April 1998.

[11] S. Acharya, M. Franklin, and S. Zdonik. Prefetching from a Broadcast Disk. In *Proc. of the International Conference on Data Engineering*, New Orleans, LA, February 1996.

[12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 1992.

[13] T. Imielinski, S. Viswanathan, and B. Badrinath. Energy efficient indexing on air. In *Proc. of ACM SIGMOD International Conference on Management of Data (SIGMOD)*, San Francisco, March 1994.

[14] M. H. Ammar. Response time in a teletext system: An individual user's perspective. *IEEE Transations on Communications*, 35(11):1159–1170, 1987.

[15] T. W. Yan, M. Jacobsen, H. Garcia-Molina, and U. Dayal. From user access patterns to dynamic hypertext linking. In *Proc. of the Fifth International World Wide Web Conference*, Paris, France, May 1996.