Dissemination-based Data Delivery Using Broadcast Disks

A Case for Data Dessemination Model

- Technological advances admittedly reduced the cost for data transfer in recent years.
- Newer and bolder applications have also come up.
- Applications requiring simultaneous accesses by million clients.
- WWW – World Wide Wait?
- The reason is we use a fundamentally unsuitable protocol such as request-response (HTTP) kind for data dissemination approach to implement these applications.
- Multiple requests in huge numbers generated by users simultaneously.
- The situation is compounded further by users not getting response continue to send (poll) more requests for same data.
- Repeated requests cost in terms of n/w bandwidth and the server cycles

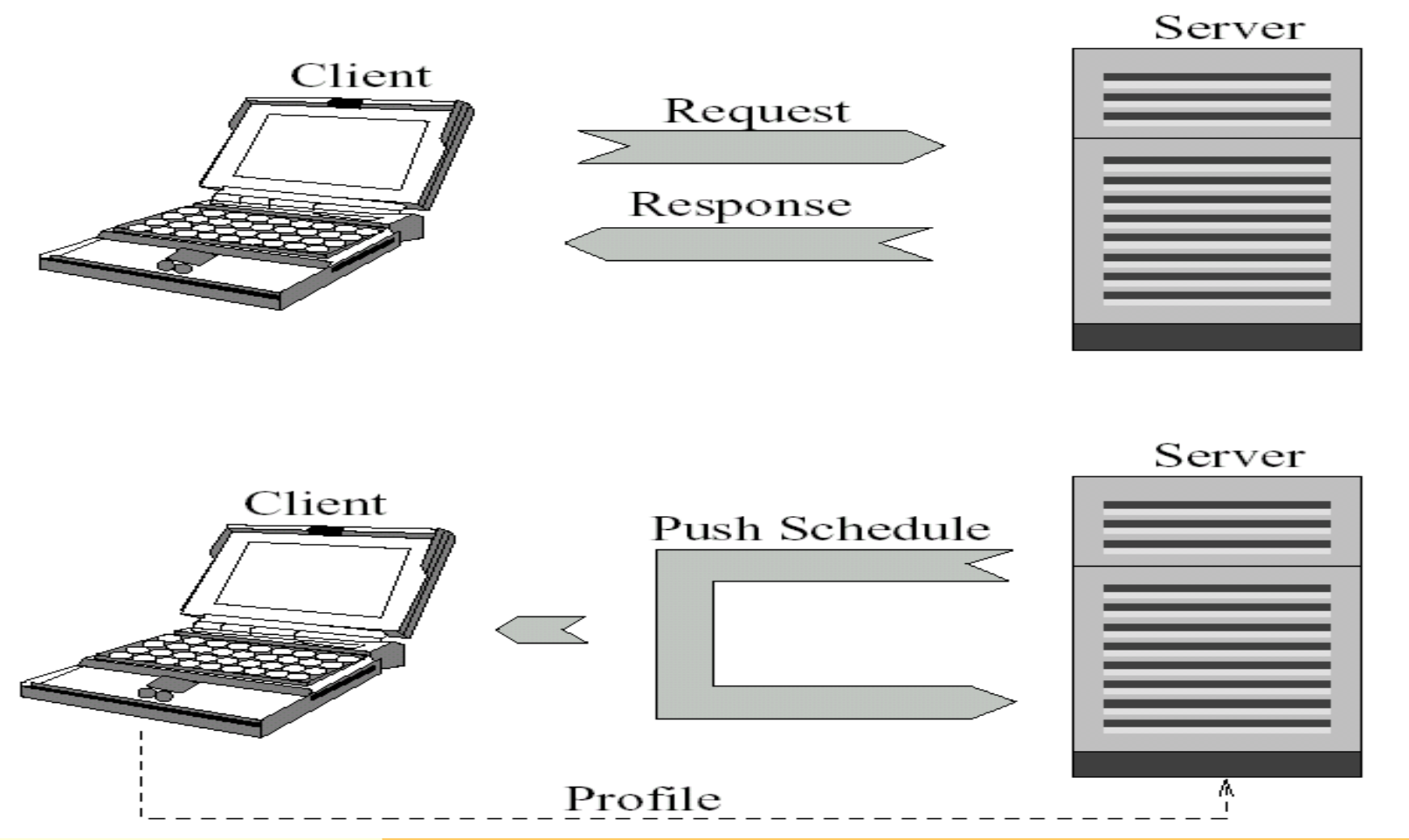
Communication Asymmetry

- Dissemination of Data to large number of clients
  - Stock Prices, news, sports, software distribution
- Access by Millions of Users
- Data from server to clients is much larger than from clients to server
  - Network Asymmetry – larger bandwidth from server to clients
  - Slow uplink
  - More clients, less server
  - Updates and new information
- HTTP and FTP limit the number of connections

Data Delivery Models

- The problem of data dissemination we discuss here is with respect to a client-server system.
- Client is consumer, server is producer. Server is the repository of info.
- Server has to deliver data which clients requires.
- There are broadly two data delivery models for transfer of data.
  - Client initiated data delivery: Client sends explicit request to server. Responsibility of data transfer rests with client. Also called Pull-based data delivery model.
  - Server initiated data delivery: Responsibility of transferring data lies with the server without any explicit request from client. Data is transferred in anticipation of future access. Push based data delivery model. Analogous to TV transmission.

Pull and Push Models



Drawbacks of Pull Based Model

- Typically similiar to RPC protocol.
- Biggest drawback is client must have a back channel to request for data.
  - First of all number of applications only provide asymmetric environment which provide unidirection communication from server to client. Client may be disconnected.
  - Availability and use of backchannel for the Client is restricted because of security reasons or battery problem.
  - Back channel congestion is a bottleneck: number of clients, who can access the channel?, etc.

Cont.

- Saturation of server – if request rate $>$ service rate it will occur eventually. Though it can be controlled in traditional application.
- Thus in summary scale is single most factor for all drawbacks
- Clients must have a priori knowledge about what to ask for.

Advantages of Push Based Model

- All drawbacks of Pull model can be considered as favourable for case of Push based model.
- A server centric approach to transfer of data.
- Efficiency: Data transfer needed only when updates or new data is created. Client is not expected to know when new data gets generated.
- Scalability: Translates into greater scalability. Client need not poll.
- Lower bandwidth utilization: No need for back channel. Data transfer when updates received or data created. So both upstream and downstream utilization of bandwidth are less.
- Push based data transfer is called data dissemination.

Explanation of Push Taxonomy

- Both Pull and Push can be periodic or aperiodic.
- Periodic push can be for a set of data updated or created in regular interval. Such as Stock prices.
  - Useful in situation where clients not available always.
  - Client request processing load is high.
  - There are several clients for what is being pushed.
- Aperiodic push is a kind publish and subscribe. There is no periodic interval for sending out the data.
  - It assumes that the clients are always listening.
  - What is sent out is not crucial.
  - Uses downstream channel more effectively.

Explanation for Pull Taxonomy

- Aperiodic pull is found in traditional system or request/response data delivery.
- Periodic pull arises when client uses polling to obtain data. It uses a regular schedule to request for data. Useful in application such as remote sensing.

Poin-to-Point and 1-to-N

- In a P-to-P communication data sent from source to a single client.
- P-to-P is not suitable for scaling up.
- In 1-to-N communication data is sent to a number of clients.
- 1-to-N can be either multicast or broadcast.
- Usually, multicasting is implemented by sending data to a router which maintains the list of recipients and forwards the data to those.
- So clients interests are known a priori as opposed to broadcast where clients are unidentified.
- In 1-to-N broadcast clients receive data for which they may not be interested.

Push Based Data Delivery Models

- A push based data delivery is one in which server transmits the data to satisfy the needs of a population of clients. The sequence of data transmitted is called a *push program* or *schedule*. If the transmission is over a broadcast channel the sequence is called a *boradcast program*.
- Periodic: Transmission can be periodic. The schedule is repeated. Designed on the basis of anticipated polling patterns of clients.
- Aperiodic: Transmission is when data updates are received or created.
- Point-to-point: data delivered on downlink channel of client. Equivalent to unicast. Note that unicast refer to a communication pattern rather than physical communication link. So in a non point-to-point broadcast medium unicast refers to case where message is meant for a single client though it is visible to all.
- One-to-many: Can be broadcast or multicast.

Classification of Delivery Mechanisms

- *Request/response*: traditional scheme such as RPC uses aperiodic pull over P-to-P connection. If 1-to-N connection is used then for the client who requested the data it appears as P-to-P, while others can snoop on the request and get data they have not explicitly requested.
- *Polling*: In some application such as remote sensing or control applications, a system may periodically send request to other sites to obtain changed values. If the information is sent over a P-to-P connection it is a pull based approach and considered as Polling. But if it is i sent over 1-to-N link then other clients can snoop.

- *Request/response*: traditional scheme such as RPC uses aperiodic
- *Publish/Subscribe*: It is push based dissemination technique. Data flow is initiated by the server and aperiodic. Typically 1-to-N.
- *Broadcast disks*: It is a periodic push. Clients wait until the item appears. So in a sense it is like accessing of a storage device who average latency is half the interval at the searched item repeats. Periodic push can use either P-to-P or 1-to-N link. But 1-to-N is more likely.

Broadcast Disk

- The periodic push is essentially akin to a secondary storage device access as far as the client is concerned.
- If every item in a broadcast schedule appears only once then the worst case wait for the client is same as one broadcast period.
- It rarely is the case that all items are accessed equally frequently.
- It tends to be skewed to a few hot spots. So it makes sense to capture the pattern of access in a broadcast program.
- Broadcast disk is a paradigm for organizing the structure of a periodic broadcast program.

Broadcast Disk

- Proposes a mechanism called Broadcast Disks to provide database access to mobile clients.
- Server continuously and repeatedly broadcasts data to a mobile client as it goes by.
- Multiple disks of different sizes are superimposed on the broadcast medium.
- Exploits the client storage resources for caching data.

Topics of Discussion

- Server Broadcast Programs
- Client Cache Management
- Prefetching
- Read/Write Case

Server Broadcast Programs

- Data server must construct a broadcast “program” to meet the needs of the client population.
- Server would take the union of required items and broadcast the resulting set cyclically.
- Single additional layer in a client’s memory hierarchy - flat broadcast.
- In a flat broadcast the expected wait for an item on the broadcast is the same for all items.

Flat Periodic Broadcast

- Flat broadcast program of the figure shown can be logically considered as a disk with the speed of one spin per broadcast period.
- Flat program is a degenerate case of generic Broadcast Disk model.
- Single rotation of a single disk per broadcast period.

Server Broadcast Programs

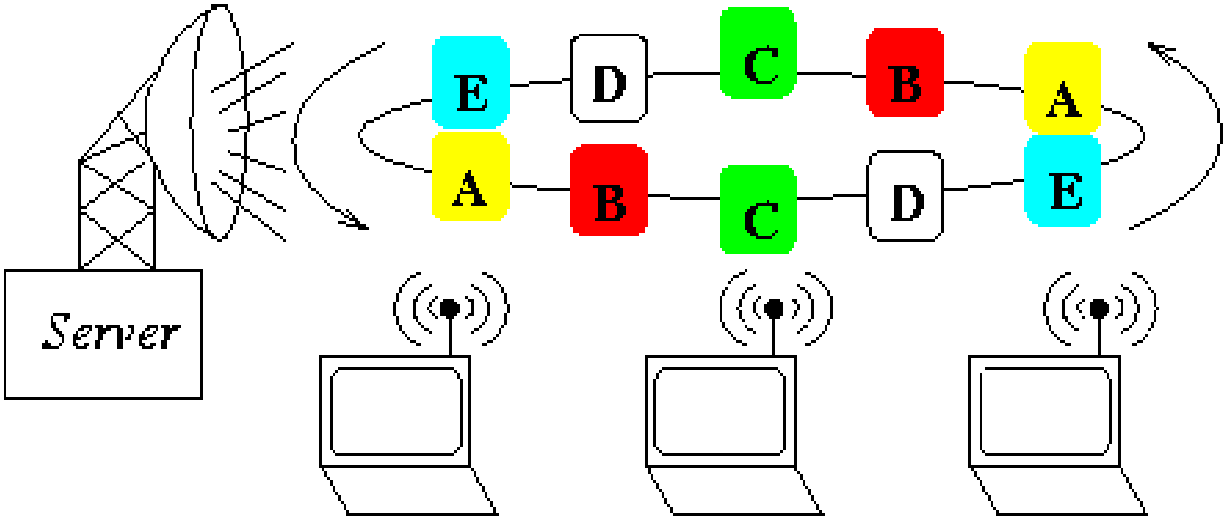


Figure 1: A Flat Broadcast Program

Server Broadcast Programs

- Broadcast Disks are an alternative to flat broadcasts.
- Broadcast is structured as multiple disks of varying sizes, each spinning at different rates.

Skewed Periodic Broadcast

- The model is like data organized into a multiple number of disks where each disk spins with a different speed.
- The data items are organized into the disks depending on their respective frequencies of accesses.
- That is the data items most frequently accessed are placed in the fastest disk.
- Data items least frequently accessed are placed in the slowest disk.

- There are two different style of periodic broadcast
  - *Static* or *Dynamic*: Where sequence of data items broadcast remains same for every broadcast period is called static. Otherwise it is dynamic. In case of dynamic periodic broadcast the period of broadcast would vary.
  - *Regular* or *irregular*: A period is regular if the inter arrival time between two consecutive broadcast of same data item is the same. Otherwise it is irregular.

Properties of Broadcast Programs

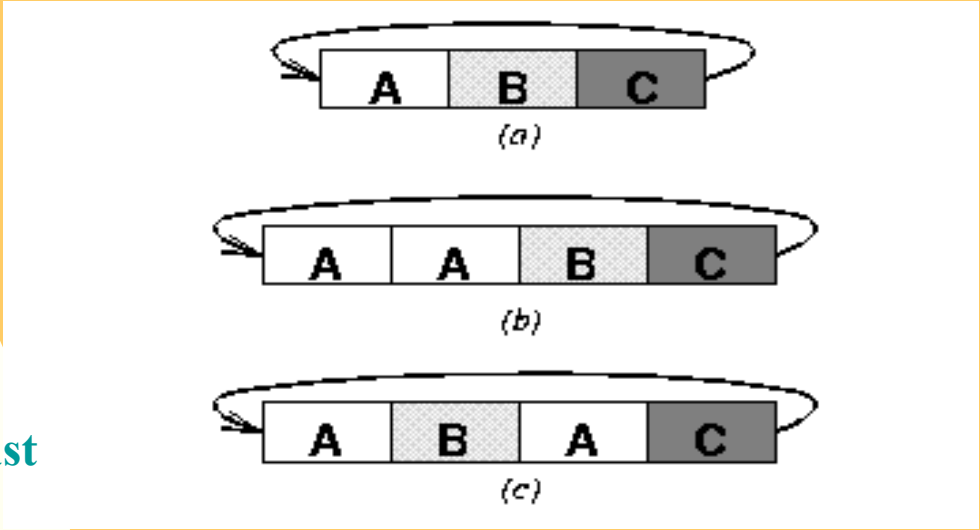
- Theoretically, broadcast program attempts to generate a banwidth allocation scheme.
  - Given all clients access probabilities, it determines the optimal percentage of bandwidth that should be allocated to each item.
  - In absence of cache, the optimal bandwidth for an item is proportional to square root of its access probability.
  - The broadcast program can then be generated randomly according to those bandwidth allocations, such that average inter-arrival time between two instances of same item matches the needs of the client population.
  - But random broadcast will not be optimal in terms of minimizing expected delay due to the variance in the inter-arrival time.

Server Broadcast Programs

Flat Broadcast

Skewed Broadcast

Multi-disk Broadcast



Example

- The performance characteristic of third (a multi-disk) program is identical to the case that A is stored on a single-page disk spinning twice as fast as the 2-page disk storing B and C.
- In case of the third program wait for accessing page, say A, is either 0 page or 1 pages, assuming that the requirement coincides with broadcast of a page.
- Therefore, average wait is 0.5 page for A.
- For page B or C the average wait is 1.5 pages.
- Assuming proability of access for each to be $1/3$, the total wait $(1/3)(0.5+1.5+1.5)=7/6$.
- In reality the requirement for a page coinciding with page broadcast is low.
- So adding $1/2$ page to total wait, we have the delay as $5/3 = 1.67$ pages.

Example

- The table shows the delay for page requests corresponding to client access probability distribution and the three broadcast programs.

Access Probability			Expected Delay		
A	B	C	Flat	Skewed	Multi-Disk
0.333	0.333	0.333	1.50	1.75	1.67
0.5	0.25	0.25	1.50	1.63	1.50
0.7	0.125	0.125	1.50	1.44	1.25
0.9	0.05	0.05	1.50	1.33	1.10
1.0	0.0	0.0	1.50	1.25	1.00

Analysis

- For uniform page access probabilities flat disk is the best.
- For skewed access probabilities non-flat programs are better.
- Obviously, higher the access probability more is the bandwidth requirement.
- When A is accessed with probability 0.5, according to square root formula the optimal bandwidth allocation is given by $\sqrt{0.5}/(\sqrt{0.5} + \sqrt{0.25} + \sqrt{0.25})$ which is 41%.
- The flat program gives only 33% (8% less in) bandwidth.
- Whereas The multi-disk program (A appears twice in the schedule) allocates 50% (9% excess in) bandwidth.

Server Broadcast Programs

- For uniform access probabilities a flat disk has the best expected performance.
  - Increasing broadcast rate of one data item decreases broadcast rate of others (continuous broadcast)
- For increasingly skewed access probabilities, non-flat disk programs perform better.
- Multi-disk programs perform better than the skewed programs.
  - If the inter-arrival time of a page (broadcast rate) is fixed then the expected delay for a request arriving at random time is 1/2 of the gap between successive broadcast of the page. In contrast, if there is variance in the inter-arrival rate then the gaps between broadcast will be of different lengths. In this case the probability of a request arriving during a large gap is greater.

Server Broadcast Programs

Generating a multi-disk broadcast

- Number of disks (*num_disks*) determine the number of different frequencies with which pages will be broadcast.

- For each disk, the number of pages and the relative frequency of broadcast (*rel_freq(i)*) are specified.

Broadcast Program Generation

(For simplicity, assume that data items are "pages", that is, they are of a uniform, fixed length.)

1. *Order the pages from hottest (most popular) to coldest.*
2. *Partition the list of pages into multiple range*, where each range contains pages with similar access probabilities. These ranges are referred to as *disk*.
3. *Choose the relative frequency of broadcast for each of the disks.* The only restriction on the relative frequencies is that they must be integers. For example given two disks, disk 1 could be broadcast three times for every two times that disk 2 is broadcast, thus, *rel_freq(1)*=3, and *rel_freq(2)*=2.

4. *Split each disk into a number of smaller units.* These units are called *chunks* (C_{ij} refers to the j^{th} chunk in disk i). First, calculate *max_chunks* as the Least Common Multiple (LCM) of the relative frequencies. Then, split each disk i into *number_chunks(i)* = *max_chunks / rel_freq(i)* chunks. In the previous example, *number_chunks(1)* would be 2, while *number_chunks(2)* would be 3.
5. *Create the broadcast program* by interleaving the chunks of each disk in the following manner:

```
01 for i = 0 to max_chunks − 1
02     for j = 1 to num_disks
03         k = i mod num_chunks(j)
04         Broadcast chunk C_{j,k}
05     endfor
06 endfor
```

Example

- Assume a list of pages partitioned into 3 disks
- Pages in disk 1 are to be broadcast twice as frequently as pages in Disk 2 and four times as frequently as pages in disk 3
- Rel-freq(1) = 4, rel-freq(2) = 2

and rel-freq (3) = 1

Max-chunk = 4, num-chunks (1) = 1, num-chunks(2) = 2, num-chunks (3) = 4

Server Broadcast Programs

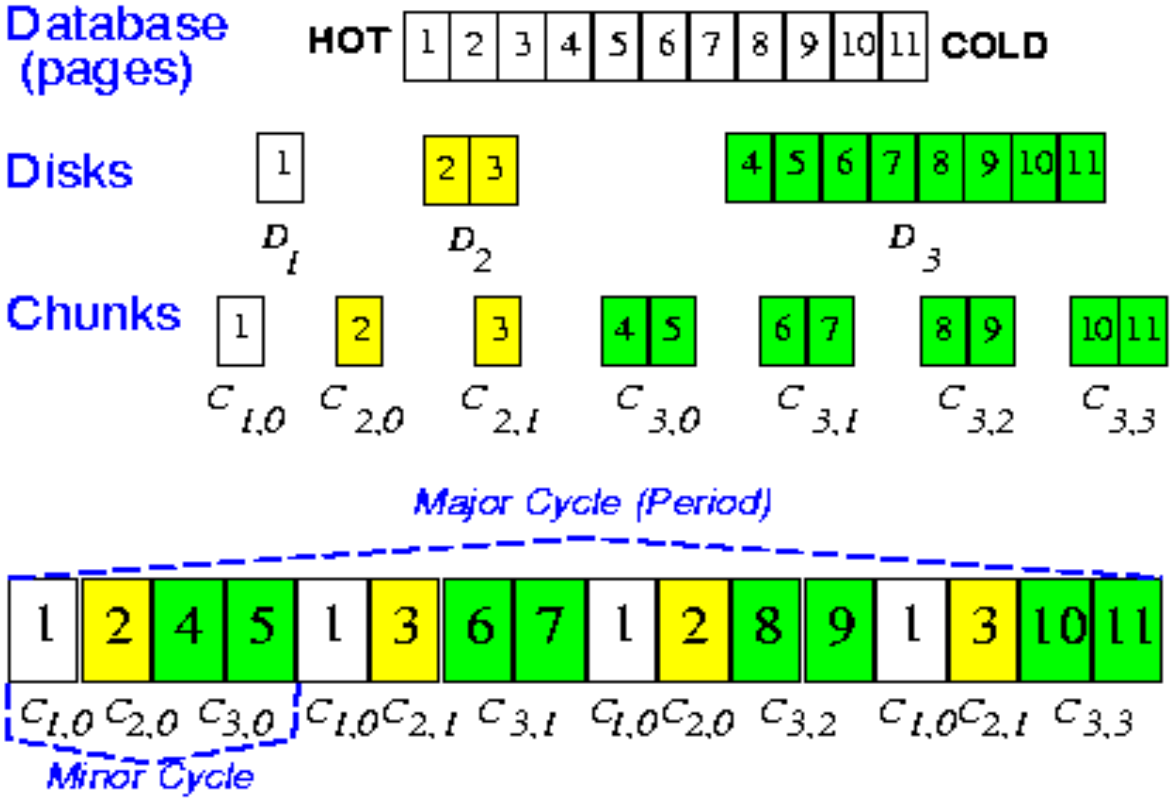


Figure 3: Deriving a Server Broadcast Program

Client Cache Management

- Improving the broadcast for one probability access distribution will hurt the performance of other clients with different access distributions.
- Therefore the client machines need to cache pages obtained from the broadcast.

Client Cache Management

- With traditional caching clients cache the data most likely to be accessed in the future.
- With Broadcast Disks, traditional caching may lead to poor performance if the server's broadcast is poorly matched to the clients access distribution.

Client Cache Management

- In the Broadcast Disk system, clients cache the pages for which the local probability of access is higher than the frequency of broadcast.
- This leads to the need for cost-based page replacement.

Client Cache Management

- One cost-based page replacement strategy replaces the page that has the lowest ratio between its probability of access (P) and its frequency of broadcast (X) - *PIX*

- PIX requires the following:

1 Perfect knowledge of access probabilities.

2 Comparison of PIX values for all cache resident pages at cache replacement time.

Client Cache Management

- Another page replacement strategy adds the frequency of broadcast to an LRU style policy. This policy is known as LIX.
- LIX maintains a separate list of cache-resident pages for each logical disk
- Each list is ordered based on an approximation of the access probability (L) for each page.
- A LIX value is computed by dividing L by X, the frequency of broadcast. The page with the lowest LIX value is replaced.

Prefetching

- An alternative approach to obtaining pages from the broadcast.
- Goal is to improve the response time of clients that access data from the broadcast.
- Methods of Prefetching:
  - Tag Team Caching
  - Prefetching Heuristic

Prefetching

- Tag Team Caching - Pages continually replace each other in the cache.
- For example two pages x and y, being broadcast, the client caches x as it arrives on the broadcast. Client drops x and caches y when y arrives on the broadcast.

Prefetching

- Simple Prefetching Heuristic
- Performs a calculation for each page that arrives on the broadcast based on the probability of access for the page (P) and the amount of time that will elapse before the page will come around again (T).
- If the PT value of the page being broadcast is higher than the page in cache with the lowest PT value, then the page in cache is replaced.

Read/Write Case

- With dynamic broadcast there are three different changes that have to be handled.

1. Changes to the value of the objects being broadcast.
2. Reorganization of the broadcast.
3. Changes to the contents of the broadcast.

Conclusion

- Broadcast Disks project investigates the use of data broadcast and client storage resources to provide improved performance, scalability and availability in networked applications with asymmetric capabilities.