# For Further Information

**http://www.cs.vt.edu/~chitra/www.html**

***Contents:***

- **workload trace files**
- **trace manipulation tools**
- **tcpdump filter with http decode to write Common Log Format**
- **simulation (with source code)**

# Work In Progress

1. **Reconciling our simulation results with others (Univ. Saskatchewan, Korea)**

2. **Extended study with non-educational workloads**

3. **New policy: latency-based (e.g., European users want North American documents to remain in cache)**

4. **Experimental observation of modified Harvest cache that uses SIZE, LFU, latency policies (looking for sites that want to run it...)**

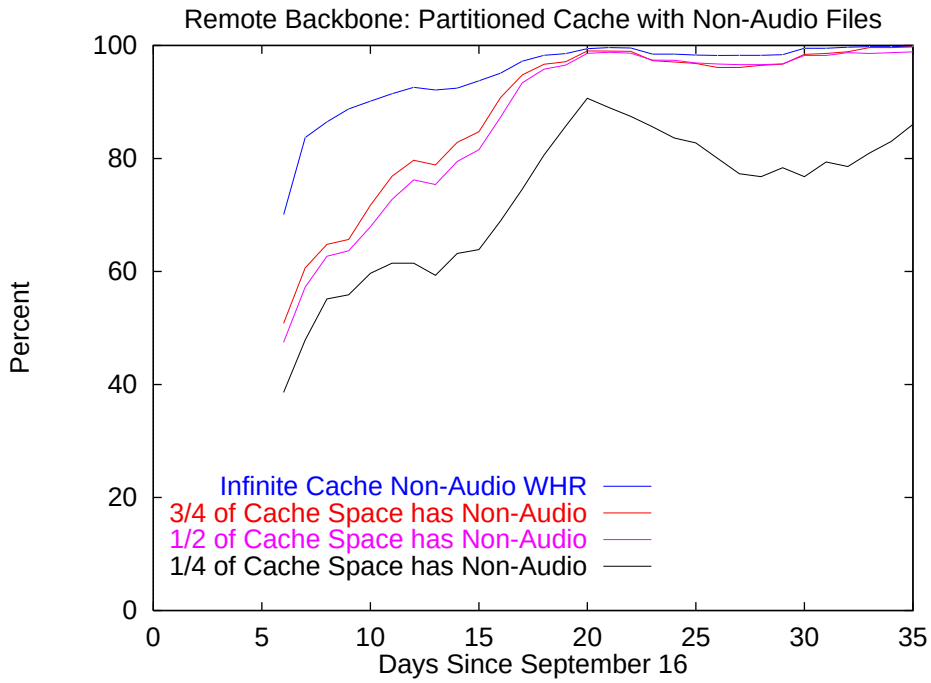
5. **Help for dynamic documents that can't be cached:**

   **Delta-based encoding in HTTP**

# Conclusions

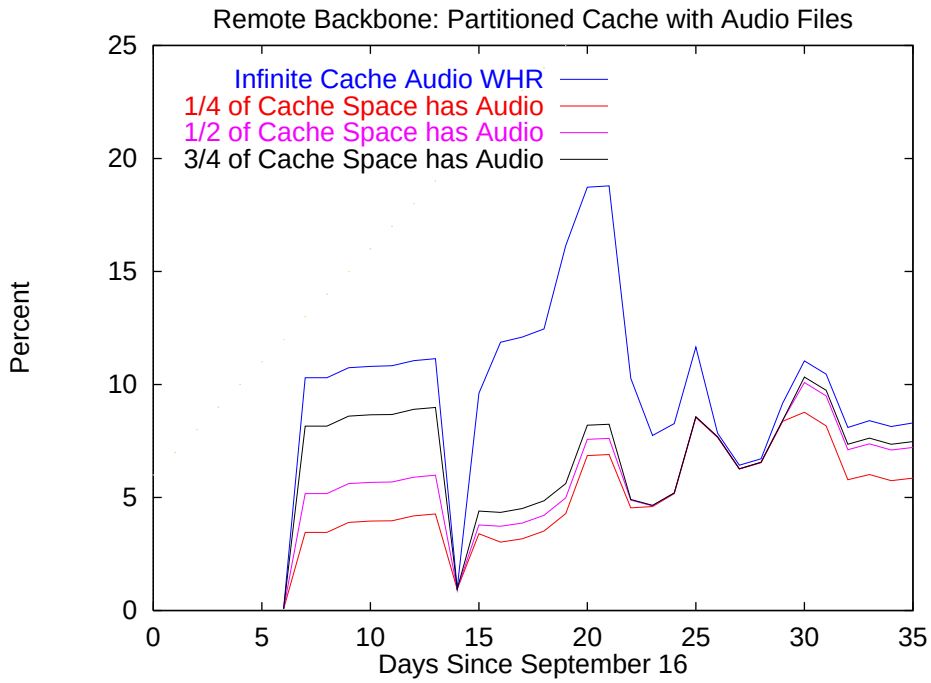
- **HR, WHR of university workloads highly influenced by academic calendar**
- **To reduce server load (HR), use SIZE as primary key.**
- **Secondary key was immaterial with SIZE as primary key.**
- **To reduce network volume (WHR), best policy is inconclusive, but worst is size.**
- **Hyper-G's key ordering is backwards for reducing server load (HR).**
- **Size is a natural policy to use in two level caches. HR, WHR are low if primary cache large.**
- **Partitioning cache by media type was effective in one server with popular audio Web site.**

# Non-audio partition




**Partition size of 1/4 reduces WHR, but larger partitions have no penalty after about day 20.**

# Workload BR, audio partition




**In log run, partitioning cache doesn't affect WHR.**

**Partition size (1/4, 1/2, 3/4) doesn't make much difference after day 14.**

# Exp. 4: Partitioning Cache by Media

**Simulate**

- **cache size = 10% of max needed**
- **two partitions: audio and non-audio**

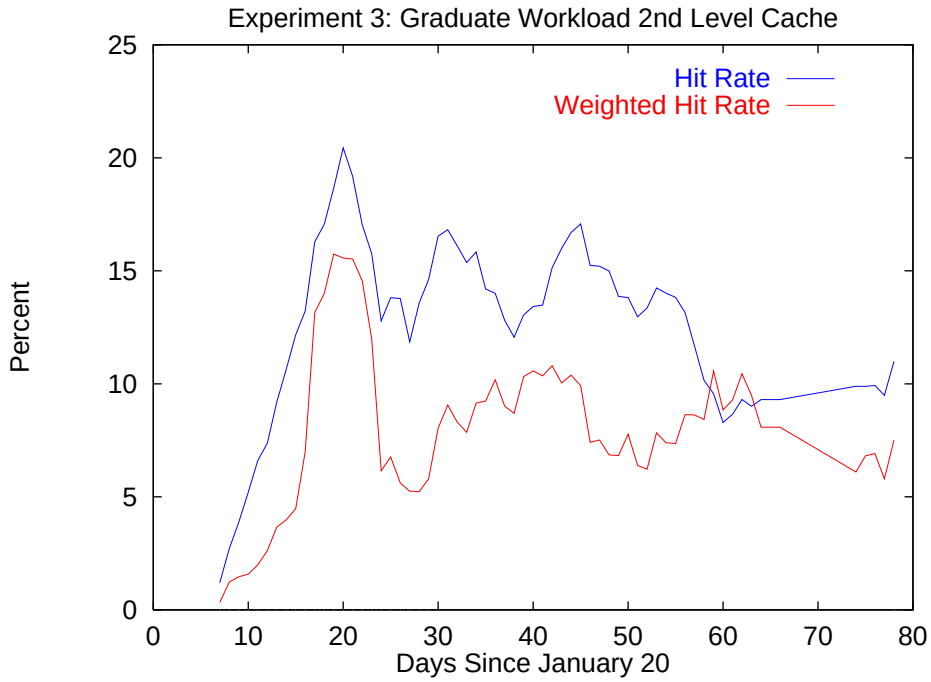
# Exp. 3: Two Level Caching

**Simulate**

- **primary cache size = 10%, 50% of max needed for no replacement**
- **secondary cache size = infinite**
- **best primary key for HR: SIZE**

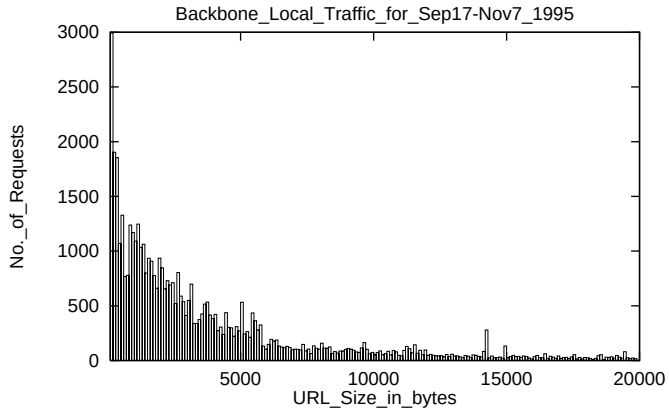
---

## ***Workload G (10% of max needed):***

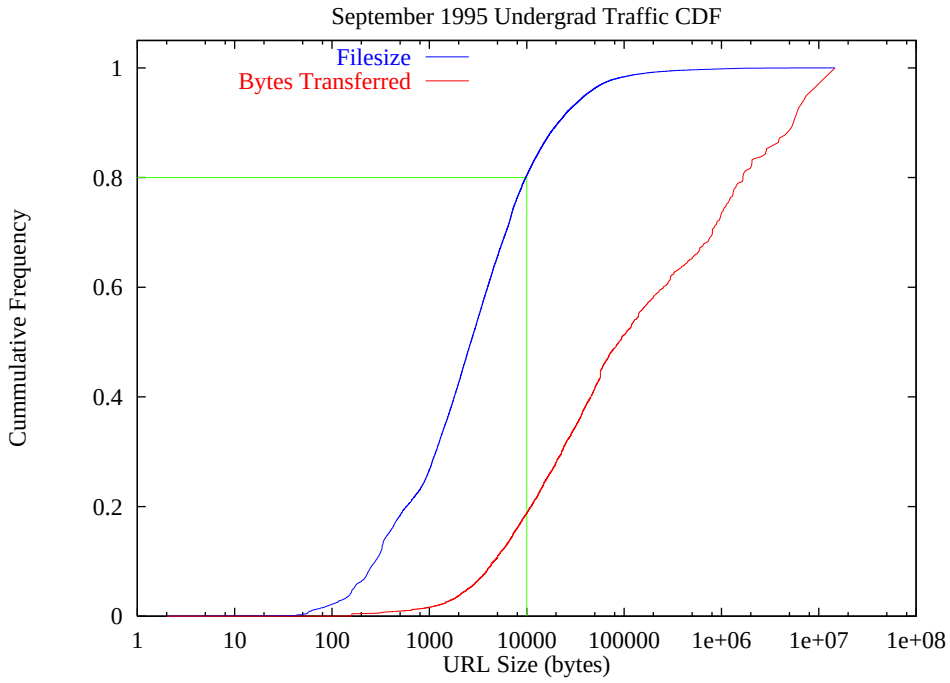


# Why Size?

## ***Distribution of doc sizes requested***




## ***Cumulative distribution of bytes requested and transfered***

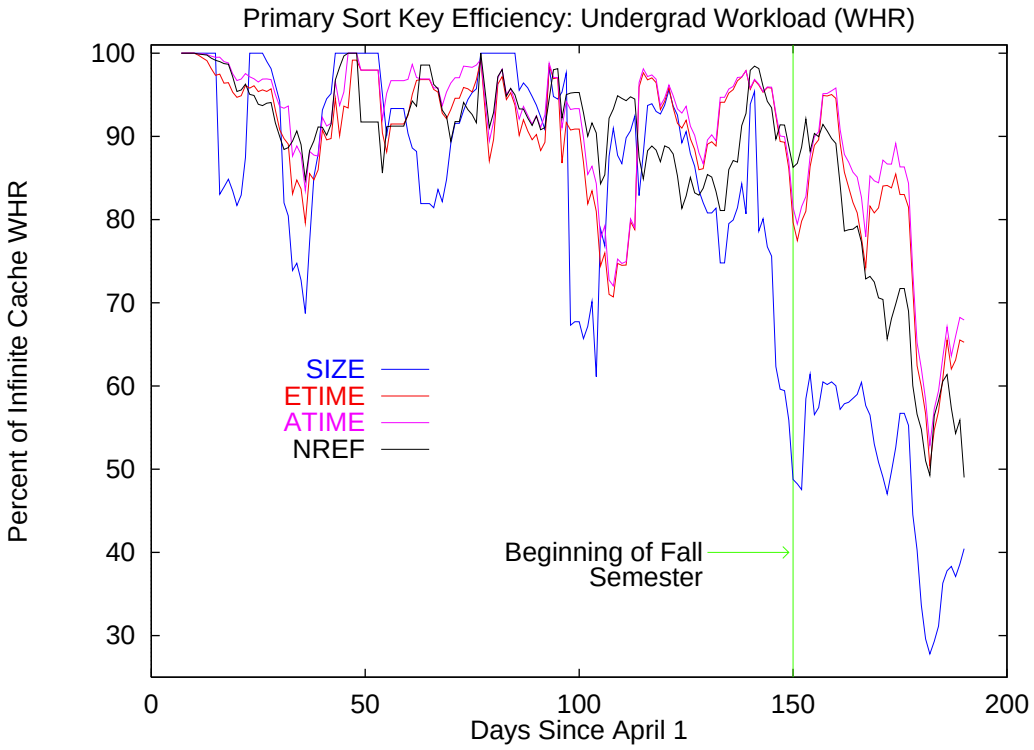


# Why Size?

- **Most accesses are for smaller documents**
- **A few loarge documents take the space of many small documents**
- **Concentration of large interreference times**

# Exp. 2: Weighted Hit Rate

## ***Workload U:***




**Size is *worse* than other keys!**

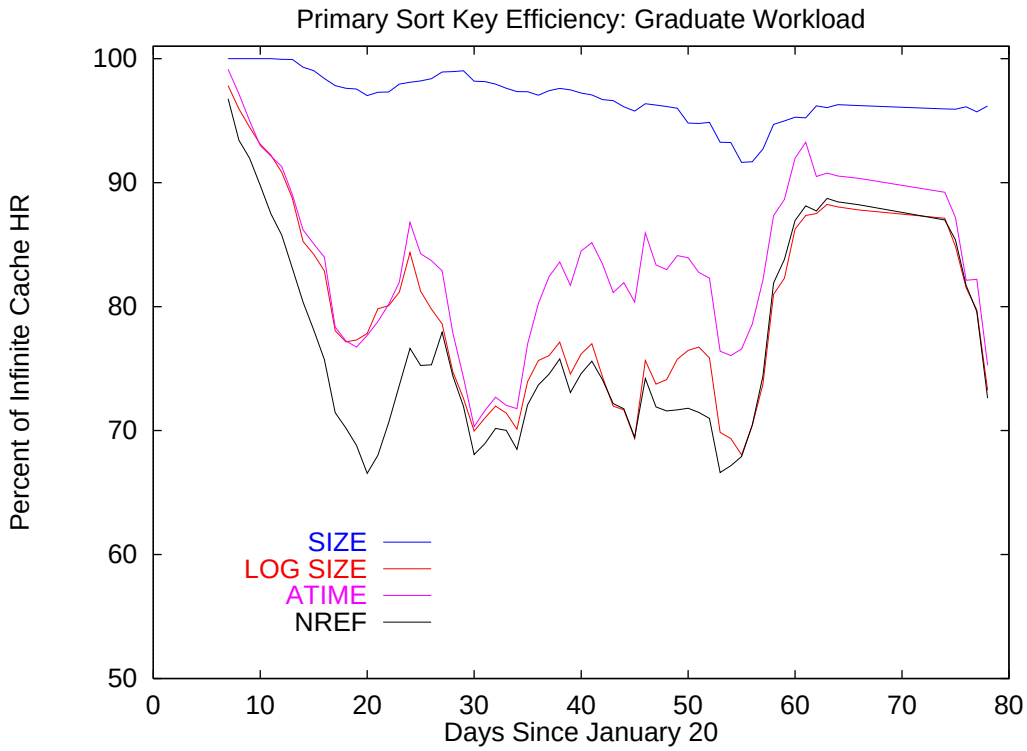
# Weighted Hit Rate

- **Results on best primary key are inconclusive**
- **Most references are from small files, but most bytes are from large files**

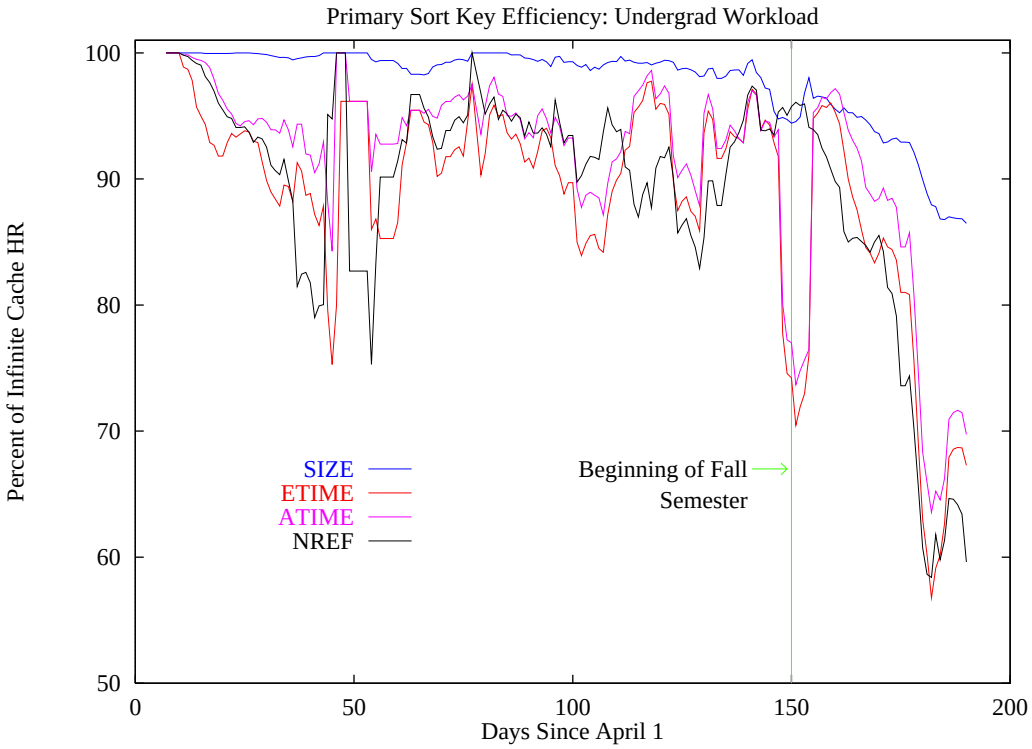
# Exp 2: Second Workload

(Cache Size = 10% of max needed)

**Size is better by a bigger margin...**



# Experiment 2: Primary Key Comparison
## (Cache Size = 10% of max needed)



# Exp 2: Removal Policy Comparison

**Simulate**

- **cache size = 10%, 50% of max needed for no replacement**
- **all primary keys**
- **certain primary/secondary combinations**

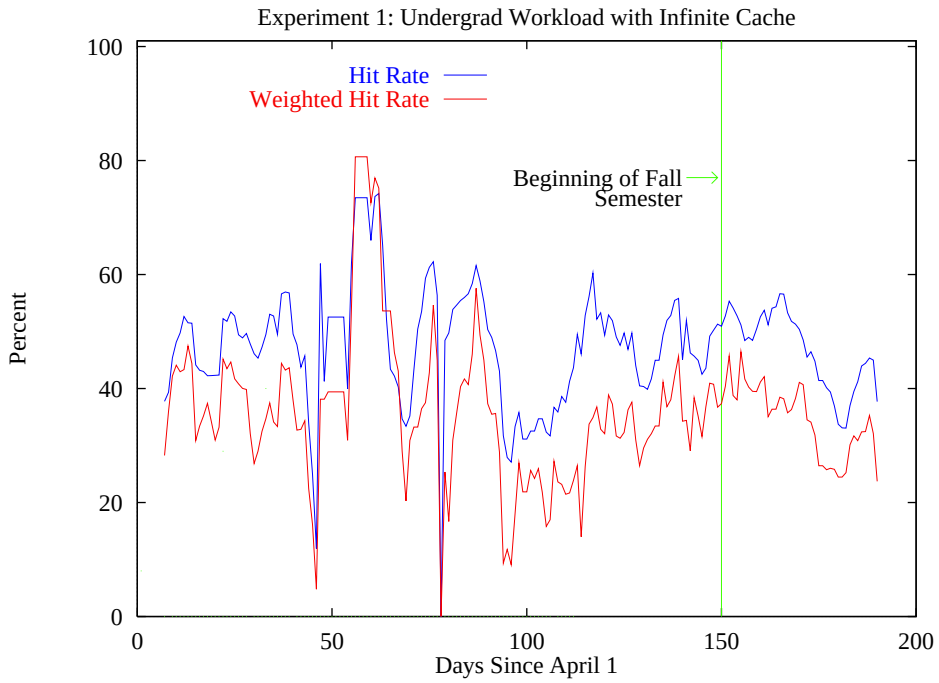
**Graph U (undergrad) --SHOWN--:**

- **SIZE superior primary key (with random secondary)**
- **Secondary key shows only marginal improvement when primary key has many ties**

**Other workloads:**

- **SIZE superior in all workloads**

# Exp. 1: Maximum Possible Hit Rate



# Exp 1: Max Theoretical HR, WHR

**Simulate infinite cache (plot 7 day moving average)**

**Workload U (undergrad) --SHOWN--:**

- **Seasonal variation (e.g., new students in fall access new URLs)**
- **Cumulative HR=44.9%, WHR=31.4%**

**Workload C (classroom):**

- **Did not show high hit rate as expected**
- **Increased HR near exams**

**Workload BR (remote clients on backbone):**

- **Hit rates over 90% due to proximity of proxy to servers**

# Simulation Assumptions

## 1. Valid Access:

- **a legal request**
- **document "passes" the cache**

**Simulate only requests with HTTP return code 200.**

## 2. Definition of *hit*:

**In reality, a "hit" is either**

- **proxy has doc, and doc estimated consistent**
- **proxy has doc, doc estimated *in*consistent, and CONDITIONAL-GET returns no doc**

**But 3 workloads traces lack last-modified times. Thus we use alternate definition:**

***Hit = match in URL and size***

## 3. When URL in common log file has size zero:

- **If URL appeared earlier with non-zero size, use last size in simulation**
- **Otherwise URL is probably a dynamic doc - don't cache in simulation**

# Experiment Overview

- **Trace-driven simulation**
- **Compare removal policies, viewed as sorting problems**
- **Answer:**

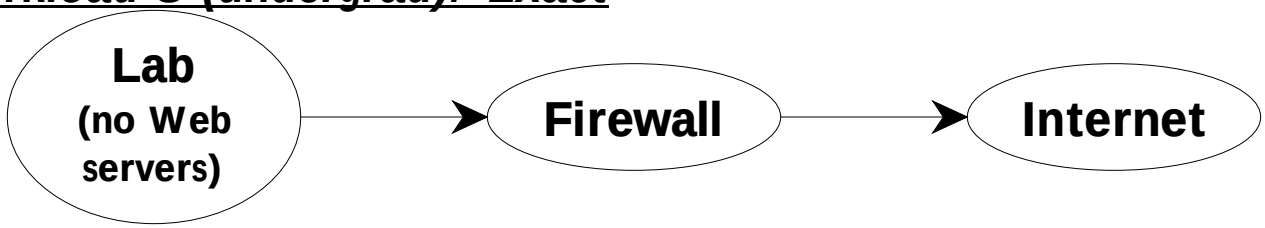
  1. **Maximum theoretical HR, WHR**
  2. **Best replacement policy**
  3. **Effectiveness of second level cache**
  4. **Effectiveness of partitioning cache by media type (Question raised by Kwan, McGrath, Reed, Nov. 95, *IEEE Computer*)**

# Workload Comparison

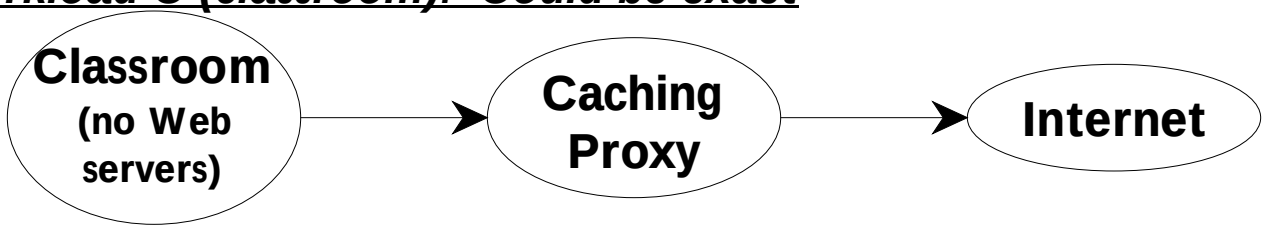
| *Work-load* | *Days* | *Accesses* | *Size (Gb)* | *Most* | |
|---|---|---|---|---|---|
| | | | | *%Refs* | *%Bytes* |
| **U** | **185** | **188,674** | **2.26** | **graphics** | **graphics** |
| **C** | **95** | **13,127** | **0.15** | **text** | **graphics** |
| **G** | **78** | **45,400** | **0.56** | **graphics** | **graphics** |
| **BR** | **37** | **227,210** | **9.38** | **graphics** | **audio** |
| **BL** | **37** | **91,188** | **0.64** | **graphics** | **graphics** |

# Workload Representativeness

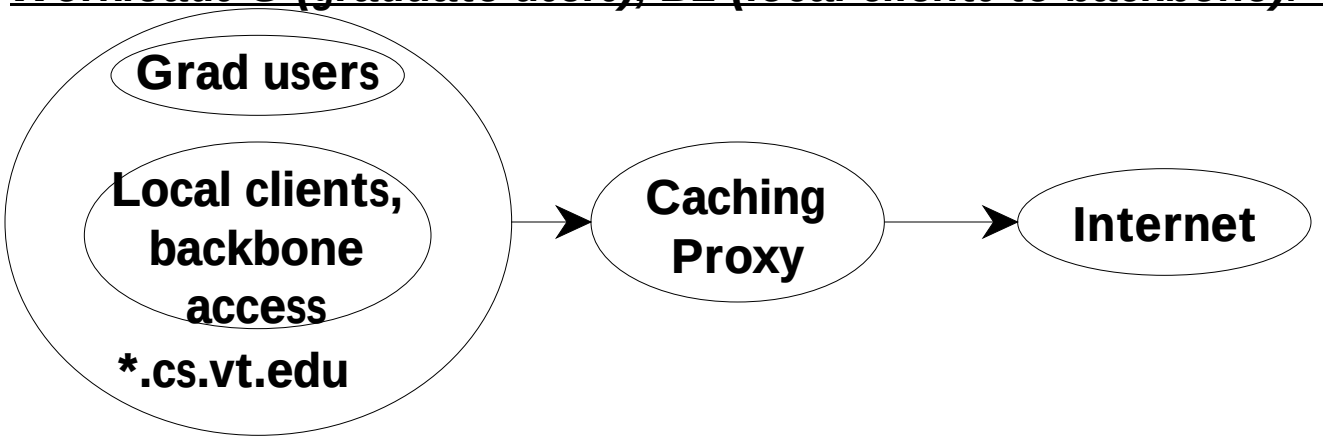
***Workload U (undergrad): Exact***



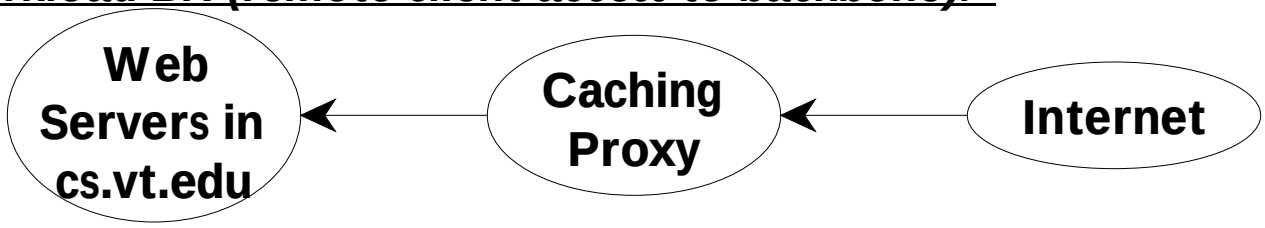

***Workload C (classroom): Could be exact***




***Workloads G (graduate users), BL (local clients to backbone):****




***Workload BR (remote client access to backbone):****




***Measures reported are upper bounds on performance**

# Sorting Key Example

| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| URL | *A* | *B* | *C* | *B* | *B* | *A* | *D* | *E* | *C* | *D* | *F* | *G* | *A* | *D* | *H* |
| Size(kB) | 1.9 | 1.2 | 9 | 1.2 | 1.2 | 1.9 | 15 | 8 | 9 | 15 | 0.3 | 1.9 | 1.9 | 15 | 5.2 |

| *URL* | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| SIZE | 1.9 | 1.2 | 9.0 | 15.0 | 8.0 | 0.3 | 1.9 | 5.2 |
| $\lfloor \log_2(\text{SIZE}) \rfloor$ | 10 | 10 | 13 | 13 | 13 | 8 | 10 | 12 |
| ETIME | 1 | 2 | 3 | 7 | 8 | 11 | 12 | 15 |
| ATIME | 13 | 5 | 9 | 14 | 8 | 11 | 12 | 15 |
| NREF | 3 | 3 | 2 | 3 | 1 | 1 | 1 | 1 |

| *Primary Key* | *Secondary Key* | *Sorted list of URLs at time* 15+ | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| SIZE | ATIME | *D** | *C* | *E* | *H* | *G* | *A* | *B* | *F* |
| $\lfloor \log_2(\text{SIZE}) \rfloor$ | ATIME | *E** | *C* | *D* | *H* | *B* | *G* | *A* | *F* |
| ETIME | **n/a** | *A** | *B* | *C* | *D* | *E* | *F* | *G* | *H* |
| ATIME | **n/a** | *B** | *E** | *C* | *F* | *G* | *A* | *D* | *H* |
| NREF | ETIME | *E** | *F* | *G* | *H* | *C* | *A* | *B* | *D* |

# Removal Algorithms

## (What to Remove When Cache is Full)

**In use or in literature:**

- **LRU**
- **Hyper-G: LFU, then LRU, then SIZE**
- **Pitkow/Recker: server algorithm that replaces files entering on oldest days first, then LRU in current day**
- **LRU-MIN: In round *i*, use LRU on documents larger than $2^{-i}$ of incoming doc size**

**Our study -- sorting keys:**

| | |
|---|---|
| **SIZE** | **document size (bytes)** |
| **ETIME** | **time doc entered cache** |
| **NREF** | **number references to doc** |
| **ATIME** | **time doc last accessed** |

**Two more motivated by literature, which exercise secondary key to greater degree:**

- **DAY(ATIME)** **day doc last accessed**
- **$\lfloor \text{Log}_2 \text{SIZE} \rfloor$**

# Choice of Performance Measures

***Three objectives of proxy caching:***

1. **Reduce number of requests reaching server**
2. **Reduce volume of network traffic**
3. **Reduce end user's latency**

***Corresponding measures:***

1. **Hit Rate (HR):**
   **% requests satisfied by cache**
   **(shows fraction of requests not sent to server)**

2. **Volume measures:**

   - **Weighted hit rate (WHR):**
     **% client-requested bytes returned by proxy**
     **(shows fraction of bytes not sent by server)**

   - **Fraction of packets not sent**

   - **Reduction in distance traveled (e.g., hop count)**

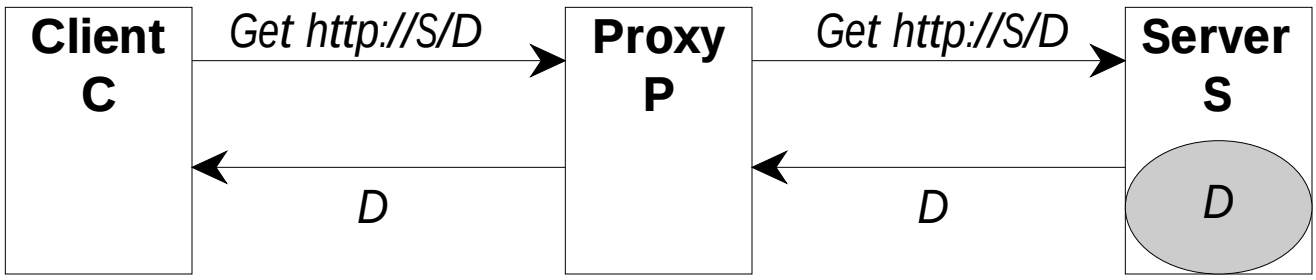
3. **Latency time**

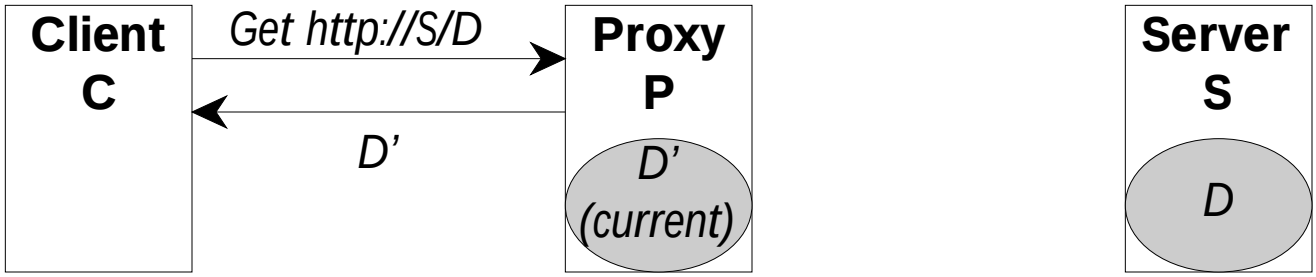
***We use HR and WHR.***

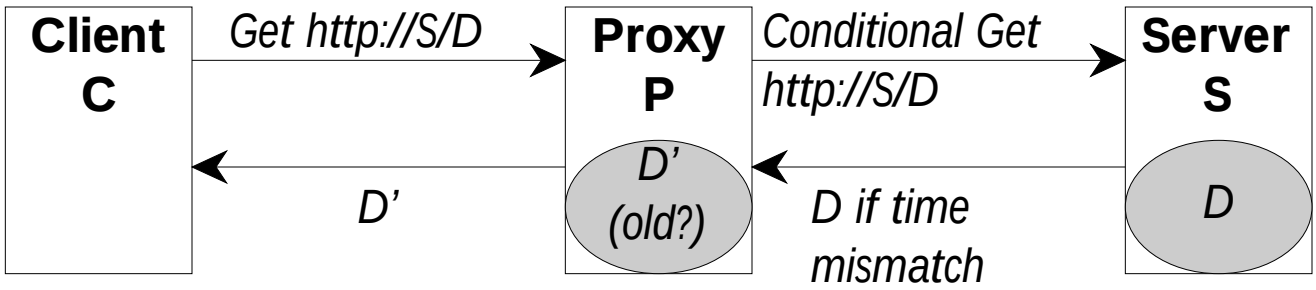
# Caching Proxies Limitations

- **Static and infrequently changing documents only**
- **Cache can't always recognize dynamic documents (e.g., script writers must use no-cache pragma in HTTP 1.1)**
- **Cache returns old document version if consistency estimation is wrong**
- **Copyright laws could legislate proxies out of existence**
- **Pay-per-view not accommodated today**

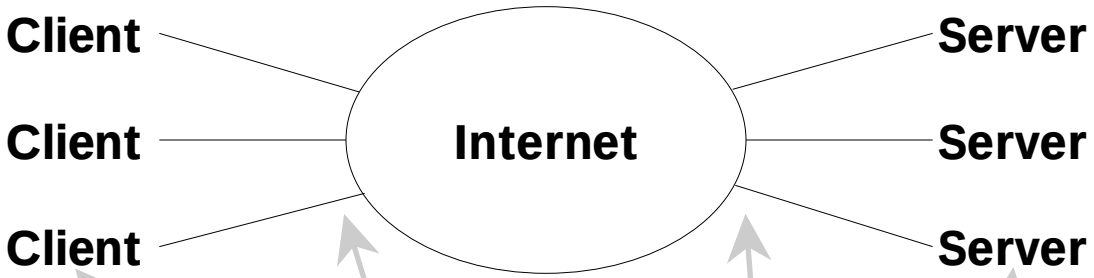
# WWW Caching Proxies



# Cache Locations




| | Client Cache | Proxy Cache *outbound* | Proxy Cache *inbound* | Server Cache |
|---|---|---|---|---|
| # clients | one | few | most | any |
| # servers | all | most | few | one |
| # docs | all | most | few | few |

***This talk***

# Why Caching Proxies?

***Problem:*** **Multiple, identical copies of WWW documents pass through same network links**

- **Wastes network resources**
- **Increases server workload**
- **Increases latency seen by user**

***Solution:*** **Migrate document copies from servers to users**

- **Distribution model**
- **Caching model**
  **[An old idea - FTP: Danzig, Hall, Schwartz SIGCOMM93]**
  - **At Web client**
  - **At Web server**
  - **In network through proxy servers**

# Outline

1. **WWW Caching Proxies**
2. **Choice of Performance Measures**
3. **Removal Algorithms**
4. **Workloads**
5. **Experiments**
6. **Conclusions**

# Removal Policies in Network Caches for WWW Documents

*Sigmetrics 96*

**Stephen Williams**
**Marc Abrams**
**Ed Fox**
**Ghaleb Abdulla**

30 August 1996
Department of Computer Science
Virginia Tech
Blacksburg, VA

**Charles R. Standridge**

Industrial Engineering
FAMU-FSU College of Engineering
Tallahassee, Fl

***Supported by***
***NSF SUCCEED consortium, NSF Educational Infrastructure grant, and NSF Research Infrastructure grant***