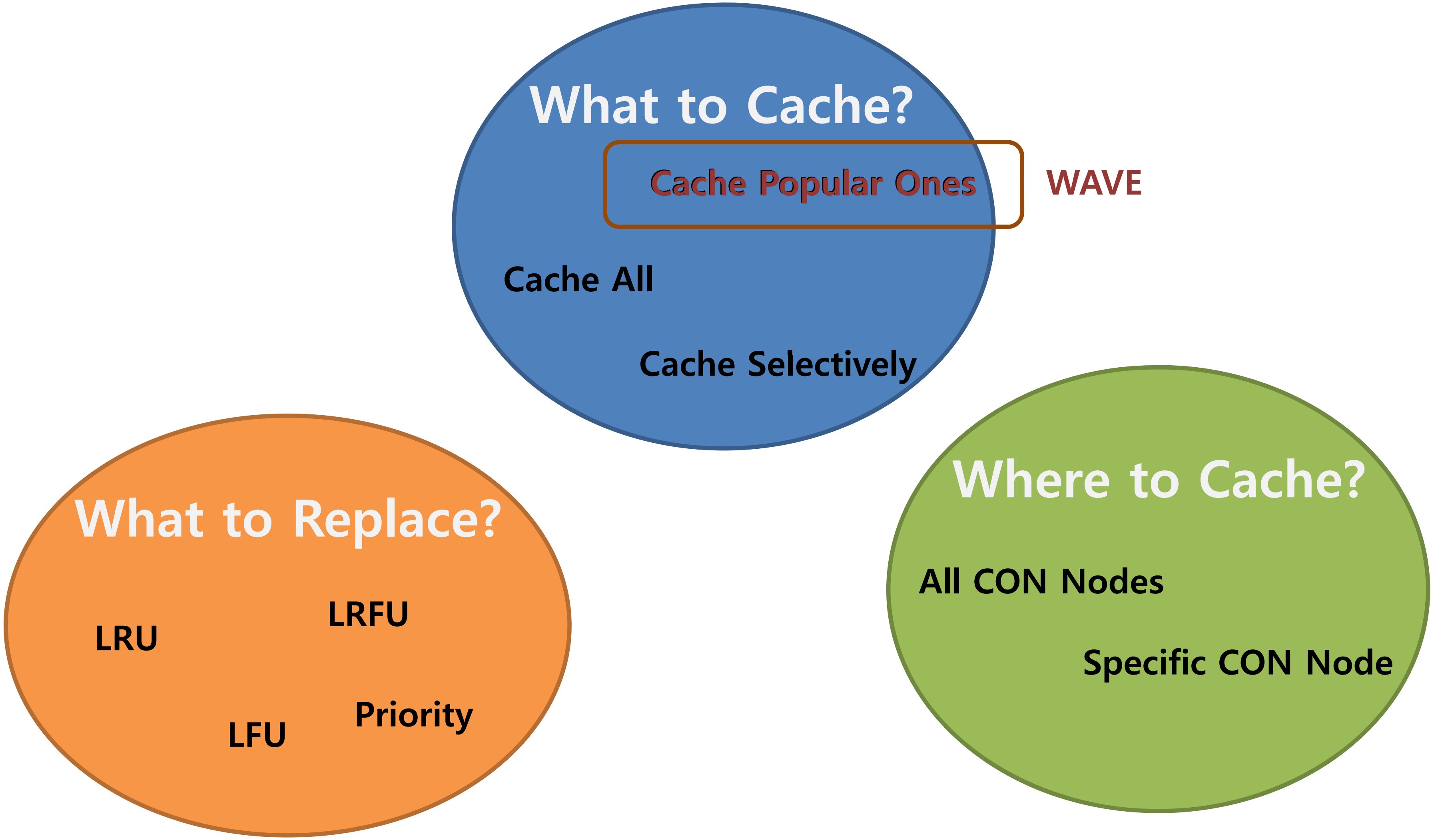
WAVE: Popularity-based and Collaborative In-network Caching for Content-Oriented Networks

Kideok Cho, **Munyoung Lee**, Kunwoo Park,
Ted "Taekyoung" Kwon, Yanghee Choi, and Sangheon Pack*

Seoul National University, Korea University*
2012. 3. 30.

Multimedia and Mobile communications Laboratory

Three Main Issues in CON Caching



Design Principles of WAVE

Popularity-based

- More chunks to be cached for more popular contents
- WAVE **exponentially increases** the number of chunks to be cached as the access count increases

Simple

- No prior knowledge of content access patterns
- WAVE uses only two counters per content file to decide caching

Decentralized

- No central server for caching decision
- In WAVE, content caching is decided by each CON router independently with its local information

WAVE Overview

- Distribute/diffuse content chunks to the network entities (such as routers)
  - Diffuse chunks as the content request changes
  - To make "the cache hits of contents" happen earlier (closer to end users)

- As a consequence,
  - Network utilization will be improved: the number of duplicate content delivery (thus total traffic volume) can be reduced
  - End users will experience reduced latency for content download
  - The overhead of cache management will be reduced

WAVE Algorithm: What to cache

- As the access count of a content file increases, WAVE exponentially increases the number of chunks of the content file to be cached

Algorithm 1 Chunk Diffusion Algorithm

x: caching base (e.g., 2,3,...)
n: chunk window state (initial value: 0)
t: total number of cached content chunks
$cached$: cached chunk id at the downstream router
i: id of requested chunk

if $cached < i \leq \sum_{j=0}^{n} x^j$ **then**
  mark chunk i to be cached
  $cached \leftarrow i$
else if $i \leq cached$ **then**
  mark chunk i to be cached
  $cached \leftarrow i$
  $n \leftarrow \lfloor \log_x i \rfloor$
end if

if $i == t$ **then**
  $n \leftarrow n + 1$
end if

Transfer the requested chunk i

Mark content chunk to be cached

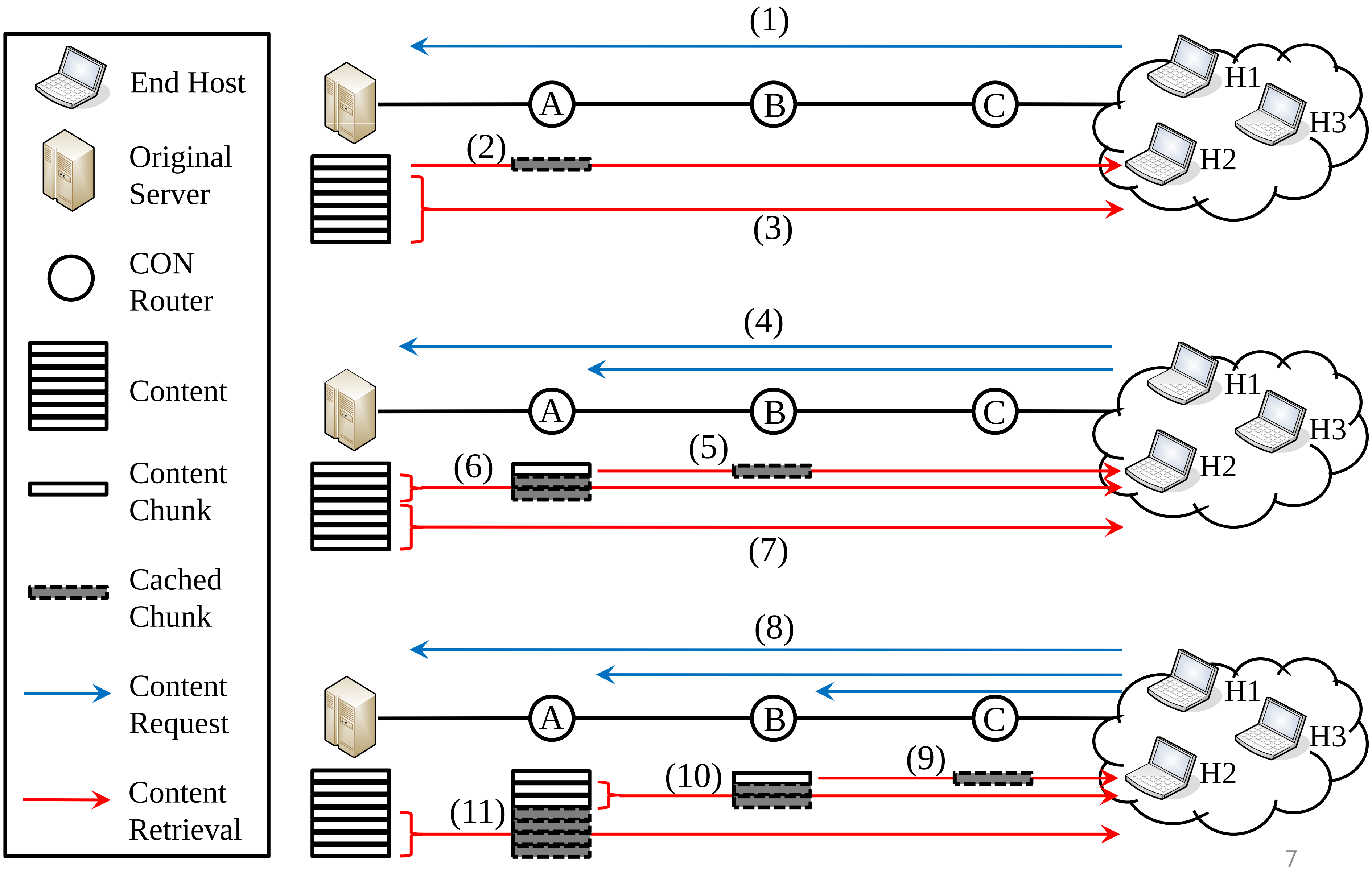
Increase window size

What to replace, Where to cache

- What to replace
  - WAVE uses least recently used (LRU) and maintains access history in the unit of a content to find a victim to be replaced
  - WAVE replaces the last chunk for the incoming chunk

- Where to cache
  - The content chunks are cached towards the direction where the content request comes considering the spatial locality
  - WAVE caches content chunks in a hop-by-hop manner to fully utilize the in-network storages

WAVE Operation Illustration



Simulation Environments

Simulation Environments	
Simulator	Discrete event-driven simulator
Topology	1 transit domain and 5 stub domains generated using GT-ITM
Number of routers & end hosts	55 routers & 1,000 end hosts
Content distribution	Randomly distributed 100,000 contents (1GBytes, 100 chunks)
Request distribution	Zipf distribution with parameter 1.0
Cache size	10GBytes
Content Routing	En-route Cache Model (shortest path to the original server)
Comparison	Client-Sever, CDN, AllCache, UniCache, ProbCache

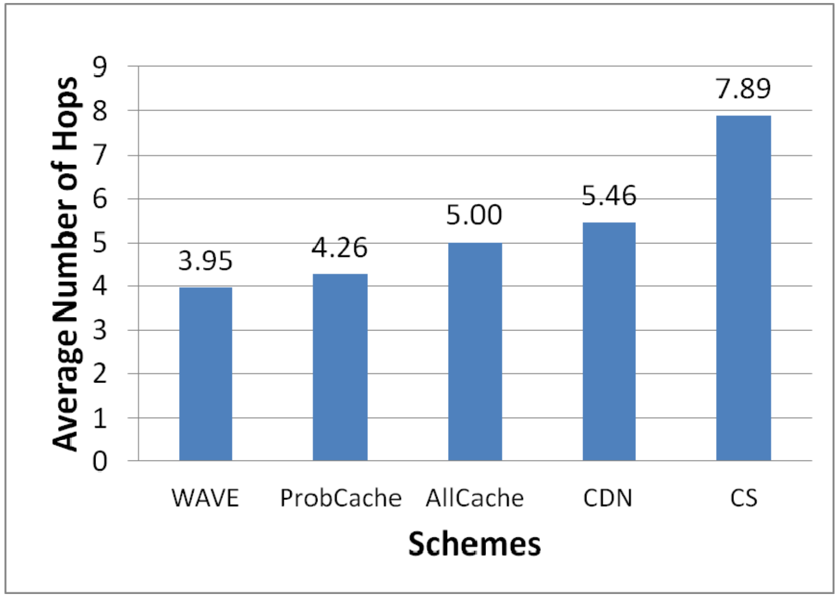
Comparison

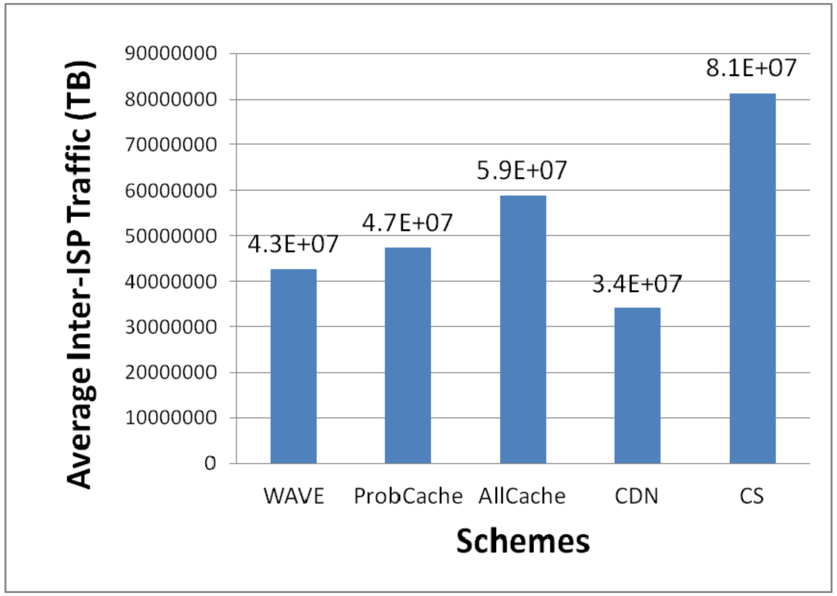
	What to cache	Where to cache	What to replace
WAVE	Exponentially Increasing number of chunks	Next downstream CON router	LRU*
AllCache (Cache all)	All incoming chunks	All CON routers	LRU
ProbCache (Probabilistic caching)	Incoming chunks with a certain probability	All CON routers	LRU
UniCache (Uniform caching)	Incoming chunks	One CON router along the returning path	LRU
CDN	Popular contents** (Top x%)	One CDN server per AS (at the best position)	N/A
Client-Server	N/A	N/A	N/A

*: Results with LFU show similar performances

**: The total # of files stored in CDN servers is the same as those of other schemes

Simulation Results (1/2)

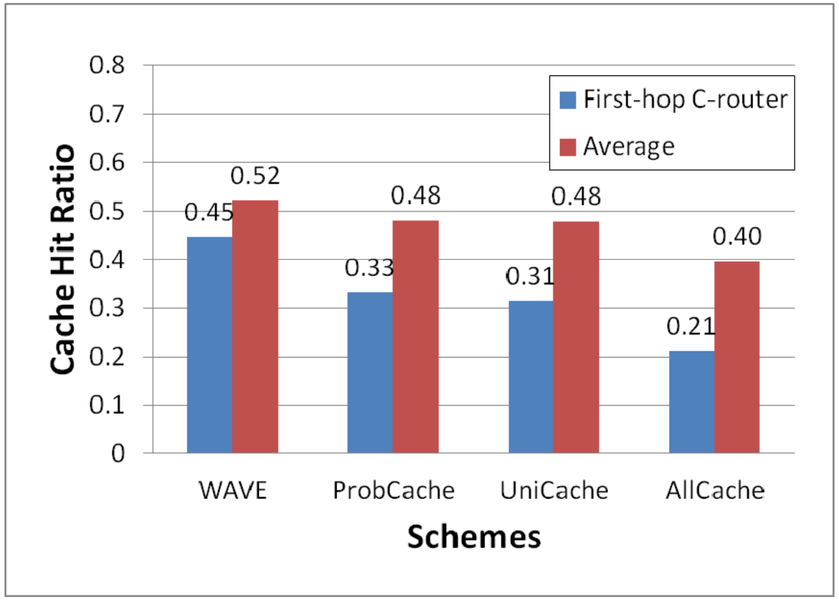


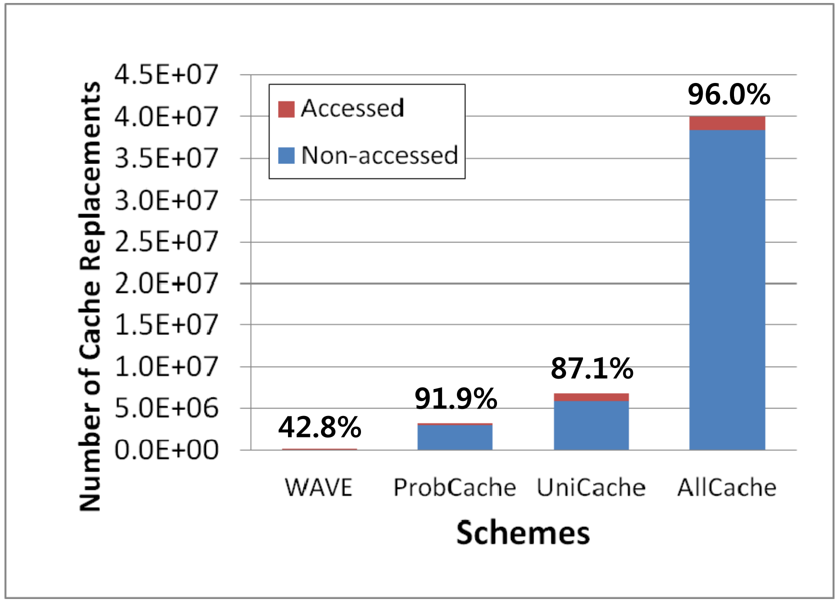


- WAVE achieves the smallest average hop count between a host and a content-holding place
  - Resulting in faster content retrieval
- By exploiting in-network storages, WAVE can cache the contents at CON routers which are closer to the end hosts than the CDN servers

- WAVE achieves the smallest inter-ISP traffic volume (except for CDN)
- Since CDN stores most popular contents in advance, it can achieve smaller inter-ISP traffic volume than WAVE
  - WAVE downloads content from outside the ISP at least once

Simulation Results (2/2)





- WAVE achieves the highest cache hit ratio than the other schemes (both 1st hop router and on average)
  - By caching the popular chunks more (exponentially increasing caching)
- AllCache shows the lowest cache hit ratio due to its popularity-blind and aggressive caching

- Less than half (42.8%) of chunks are replaced without being accessed in WAVE due to its popularity-based chunk caching algorithm
  - Resulting in efficient cache management
- In the other schemes, more than 87% of content chunks are not accessed before being replaced

Conclusion

- WAVE is a simple and decentralized caching algorithm in content-oriented networks
- WAVE exponentially increases the number of cached chunks of a content as its access count increases
  - WAVE achieves higher cache hit ratio and lower number of cache replacements
- We will implement WAVE using CCNx and conduct large-scale experiments over PlanetLab