Optimal Cache Allocation for Content-Centric Networking

Yonggong Wang, Zhenyu Li, Gaogang Xie
Chinese Academy of Sciences

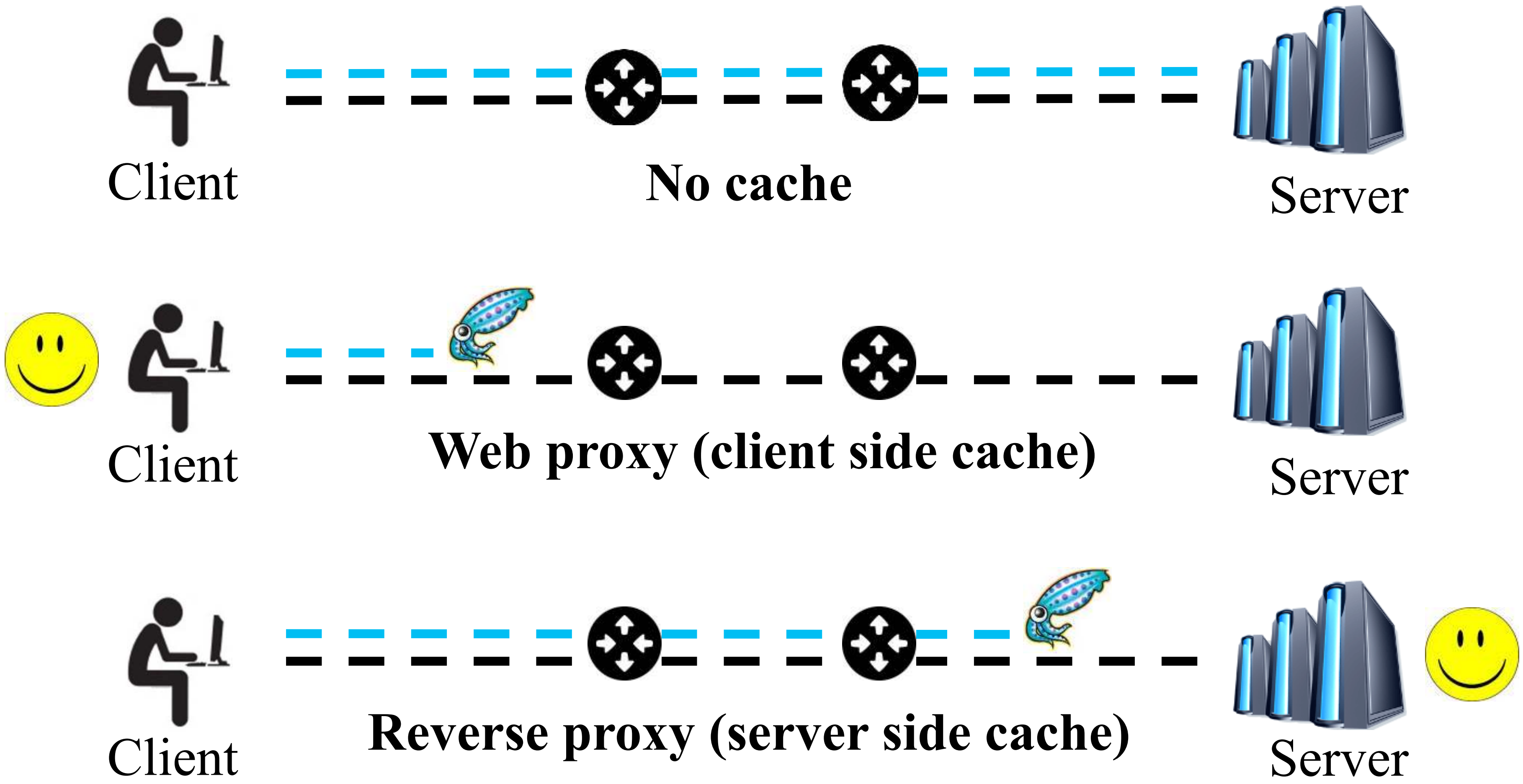
Gareth Tyson, Steve Uhlig
QMUL

Yonggong Wang, Zhenyu Li, Gareth Tyson, Steve Uhlig, Gaogang Xie. ***Optimal cache allocation for Content-Centric Networking.*** Proc. of IEEE ICNP, 2013.

Outline

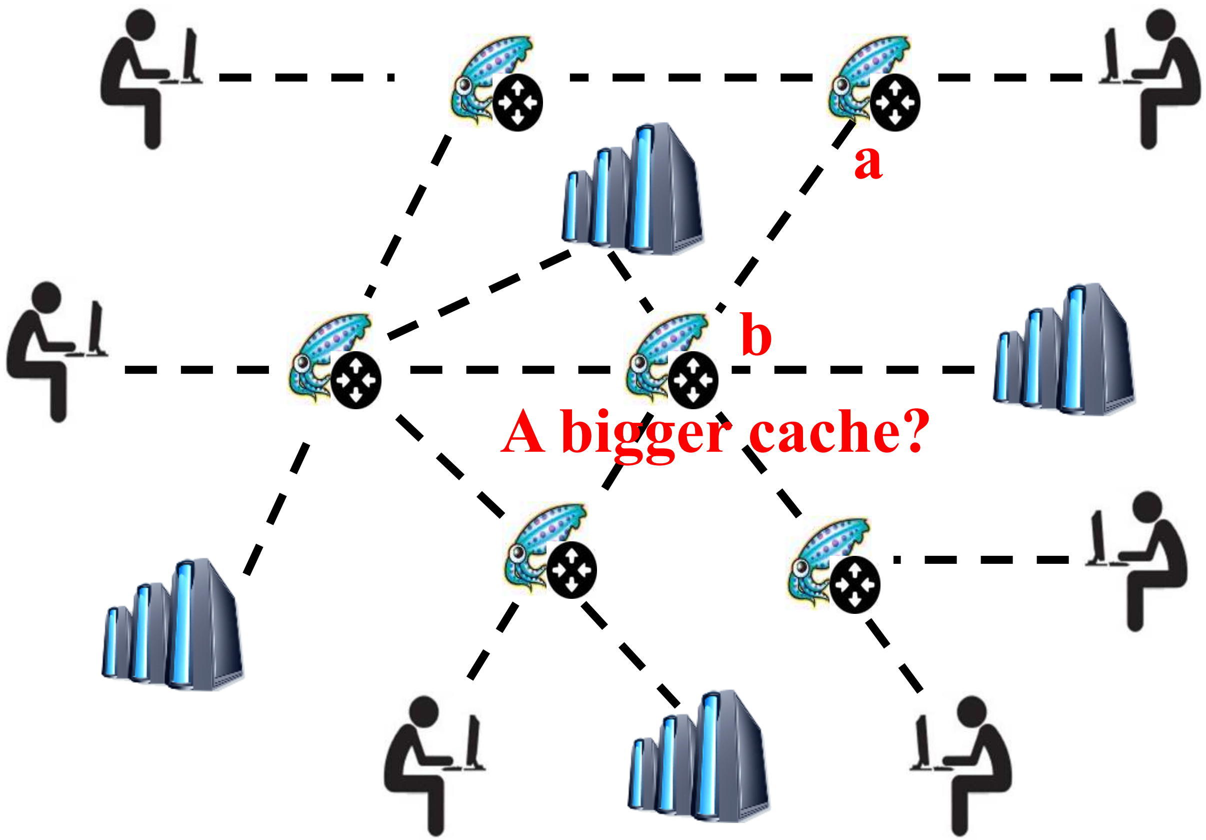
- **Background**
- Optimal cache allocation
- Evaluation
- Conclusion

Caching in the Internet



 is a widely used open-source caching proxy software

Caching in CCN



Cache everything everywhere

Motivation/related work

Q: Where should we allocate the cache space? Core, edge, or both?

- A1: More cache in the "Core"
  Cache space should be proportional to the centrality metric, e.g., the degree of node. (INFOCOM NOMEN 2012)

- A2: More cache at the "Edge"
  Keeping more cache at the "edge" is more efficient than at the "core". (SIGCOMM ICN 2012)

- A3: Cache at the "Edge" is good enough
  The benefit of caching at "Both" is very limited: < 10%. (SIGCOMM 2013)

Aims of this work

1. Find the optimal cache allocation in a given topology assuming a given content popularity distribution and pre-fetching
2. Explore the factors that impact the optimal cache allocation and the corresponding caching performance

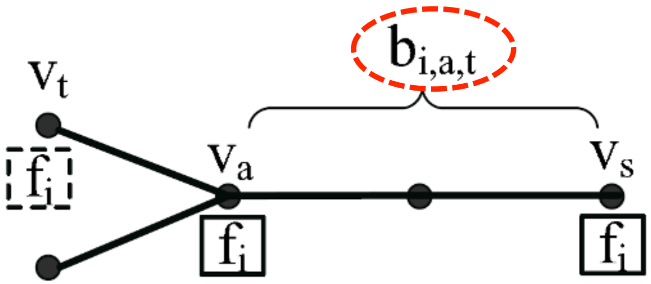
Approach: Black-box ~ use optimization to guess which strategy fits which situation

Outline

- Background
- **Optimal cache allocation**
- Evaluation
- Conclusion

Benefit of caching

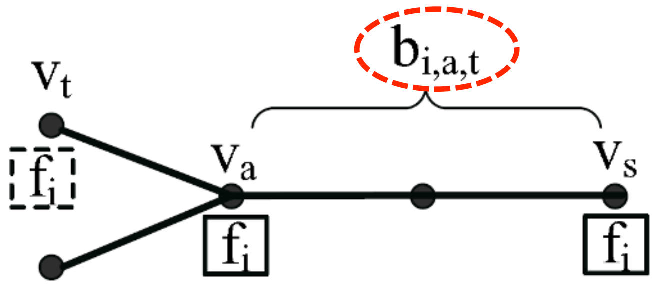
- CCN network: G=(V, E)
- Every content f_i is originated at a single server node
- Benefit of caching:



- v_t and v_s denote the client and server of content f_i
- Without caching, traffic flow for v_t to get $f_i = 3$ (hops)
- If f_i cached at v_a, traffic flow = 1.
- Benefit of caching f_i at v_a for $v_t = 2$, i.e., $b_{i,a,t} = 2$

Optimal content placement

- Probability p_i of the content f_i to be requested
- Bounded total cache space c_{total}



Optimal content placement

Maximize:

$$\sum_{f_i \in F} \sum_{v_t, v_a \in V} p_i \cdot x_{i,a} \cdot b_{i,a,t} \tag{1}$$

Subject to:

$$\sum_{f_i \in F} p_i = 1 \tag{2}$$

$$x_{i,a} = \{0, 1\}, \forall f_i \in F, v_a \in V \tag{3}$$

$$\sum_{f_i \in F} \sum_{v_a \in V} x_{i,a} \leq c_{total} \tag{4}$$

Knapsack formulation

If p_i is known, the objective function can be rewritten into the following Knapsack problem:

$$max(\sum_{f_i \in F} \sum_{v_t, v_a \in V} p_i \cdot x_{i,a} \cdot b_{i,a,t}) \quad (5)$$

$$= max(\sum_{f_i \in F} p_i \sum_{v_t, v_a \in V} x_{i,a} \cdot b_{i,a,t}) \quad (6)$$

$$= max(\sum_{f_i \in F} p_i \cdot b_i^{c_i}) \quad (7)$$

Knapsack problem!

where b_i^{ci} is the benefit of allocating c_i cache entries for content f_i across the whole network.

- Assuming unique origin for f_i, solve the cache location problem in the SPT rooted at the origin server of f_i

Methodology

Input: topology, content popularity, content server

Steps:

1. Compute the benefit of cache placement on the SPT rooted at each server
2. Resolve the final objective function as a knapsack problem

Output: a N x N binary matrix, X, describing the optimal content placement

N: #content chunks; n: #CCN routers

- Solution: Optimal cache allocation can be obtained by summing the columns of X.

Outline

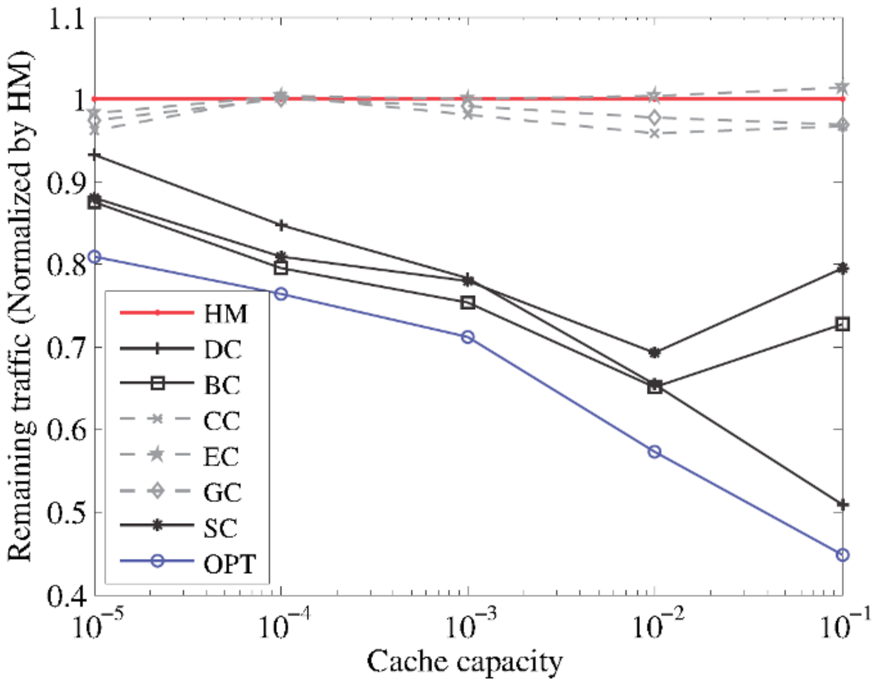
- Background
- Optimal cache allocation
- **Evaluation**
- Conclusion

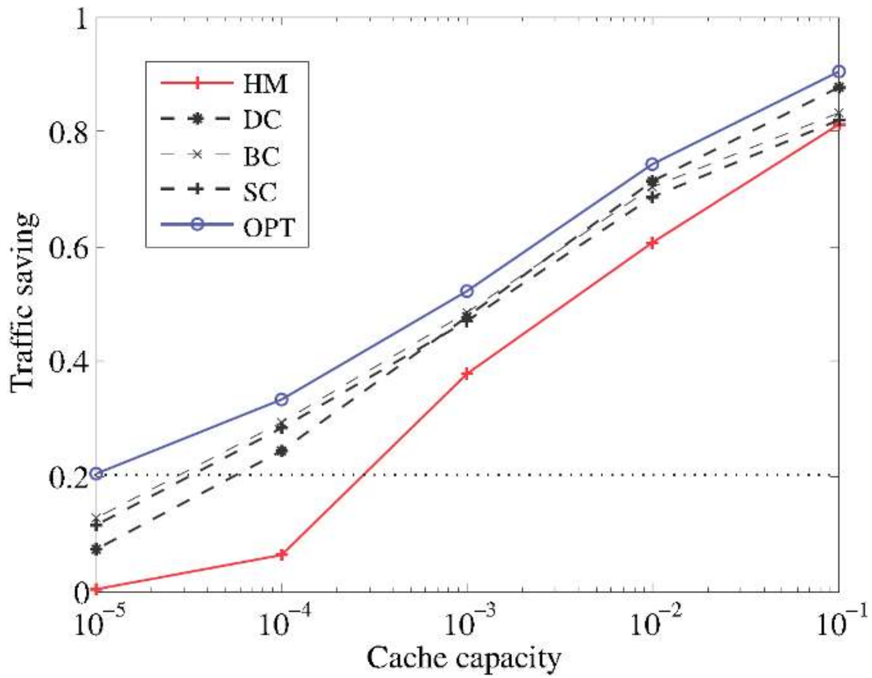
Evaluation setup

- **Simulation tool**: custom-made lightweight discrete event based simulator designed to scale to 1000s nodes
- **Topology**: Barabási-Albert (BA) & Watts-Strogatz (WS)
- **Content popularity**: Zipf
- **Cache placements:** Pre-fetching (OPT) vs. LFU
- **Cache capacity:** c_{total} expressed as fraction of nN
- **Default parameters**:
  - #Routers (N): 1000
  - #Servers: 100 (randomly chosen across network)
  - #Content (n): 10k equal-sized objects (randomly distributed across servers)
  - C_{total}: 1%

Traffic savings

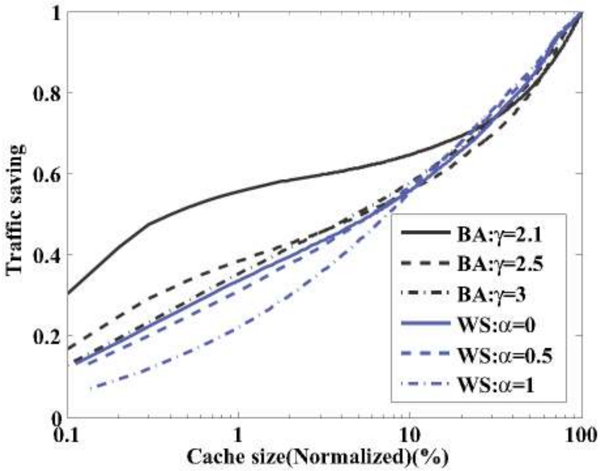
HM: homogenous allocation; OPT: optimal cache allocation; DC: Degree Centrality; BC: Betweenness Centrality; CC: Closeness Centrality; EC: Eccentricity Centrality; GC: Graph Centrality; SC: Stress Centrality.

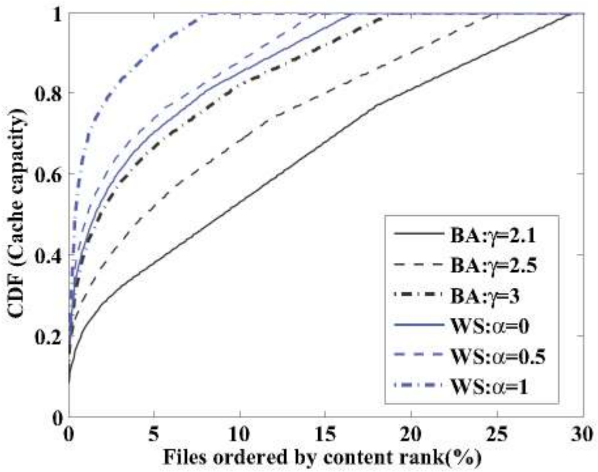


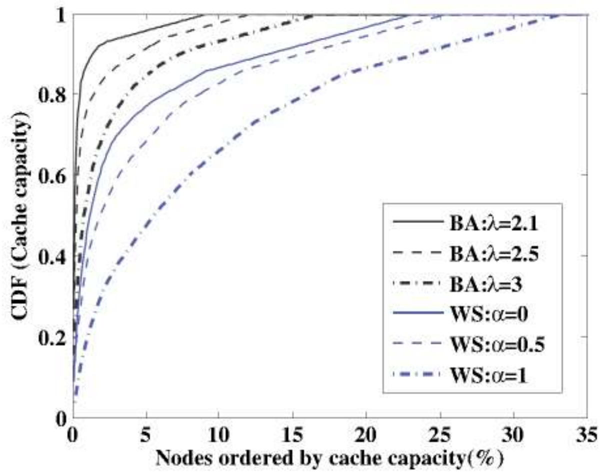


Cache allocation strategy matters, especially when the total cache budget is small.

Topological properties







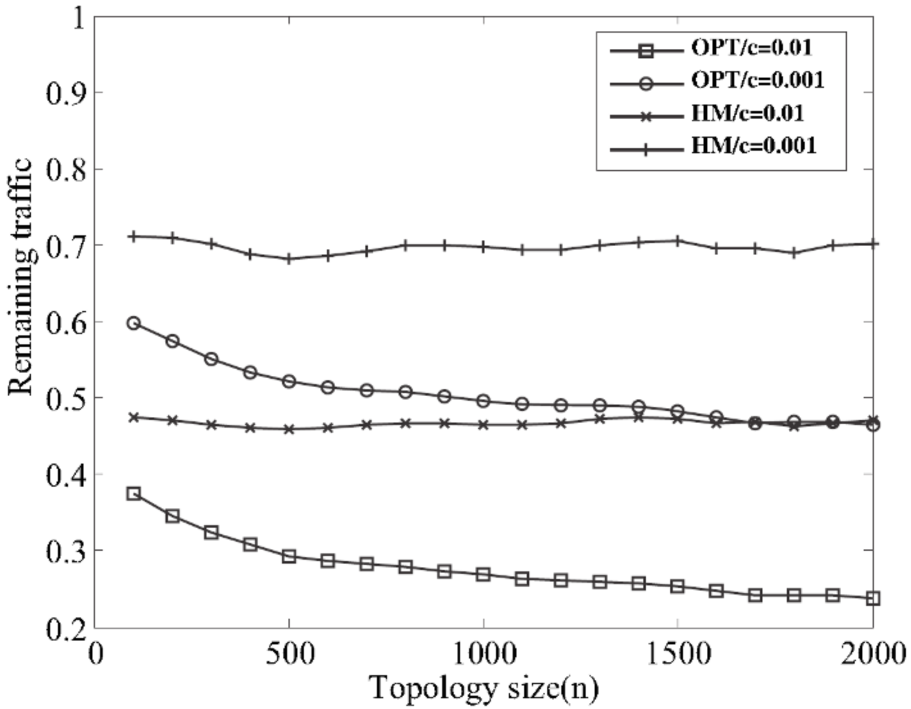
Hierarchical topology better than meshed. Clustering helps though.

Highly popular content has to be highly replicated on non-hierarchical topologies.

Capacity budget spending: Non-hierarchical topologies require spreading of the cache budget across more nodes (at the edge).

Topology structure fundamentally affects the appropriate caching strategy.

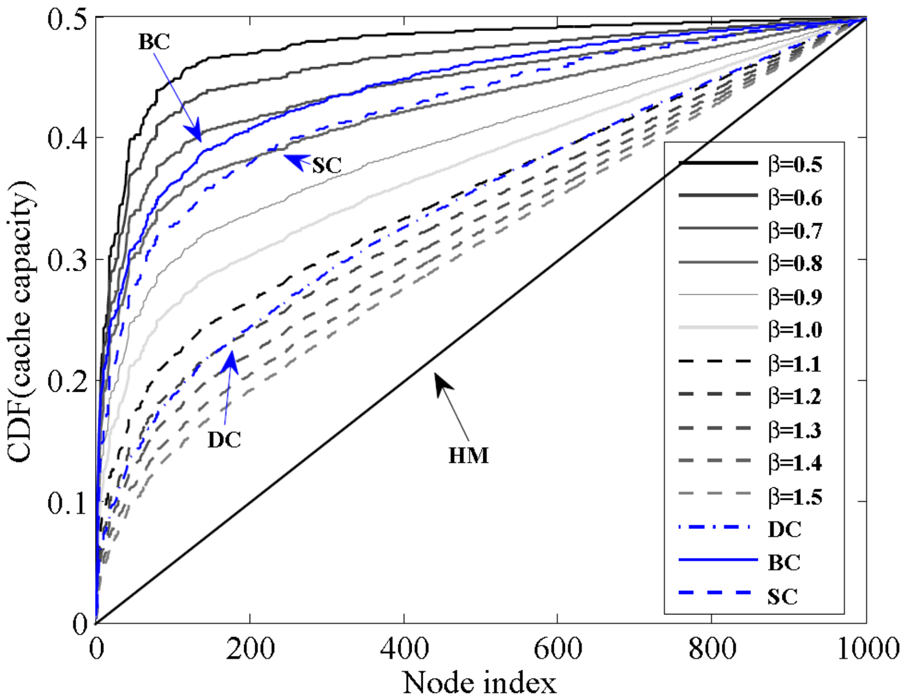
Scaling network size



Traffic saving of homogeneous allocation does not depend on network size.

Optimal allocation benefits from "economies of scale", by exploiting the topology structure.

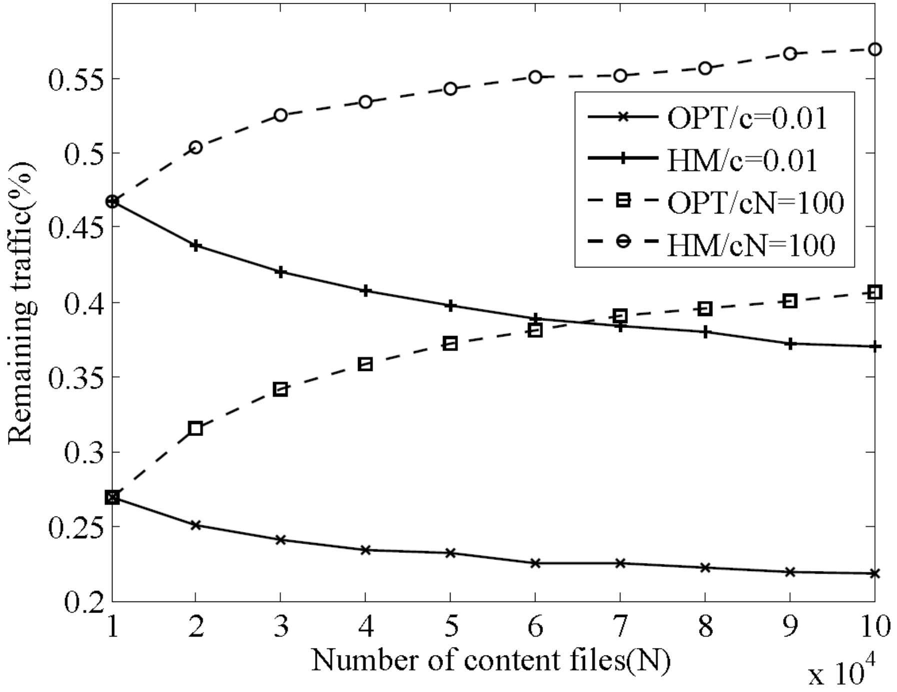
Content popularity distribution



Uniform content popularity distribution leads to cache capacity allocated to a few central nodes.

Depending on the skew in the content popularity, centrality-heuristics may be appropriate in allocating the cache capacity.

Number of content objects

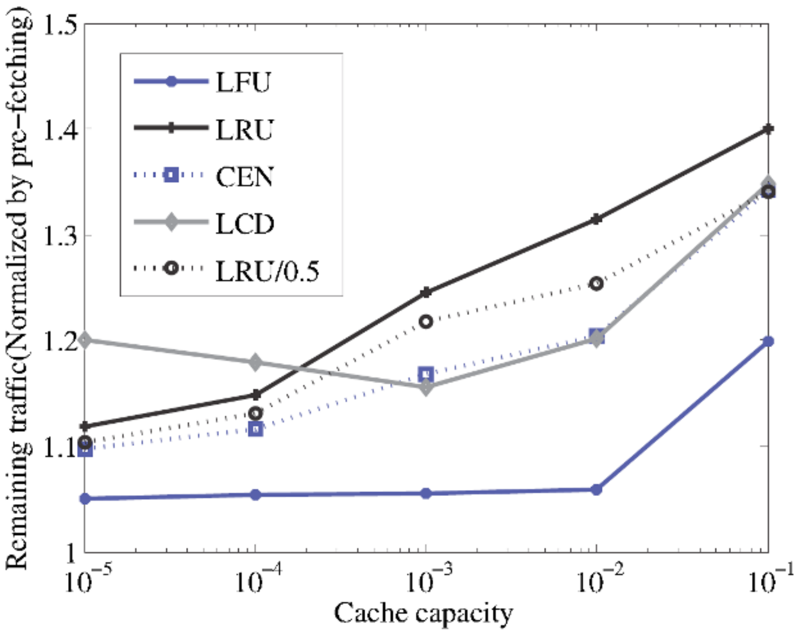


If cache budget increases proportionately with content objects (c = fixed percentage), traffic savings improve, irrespective of the content placement strategy.

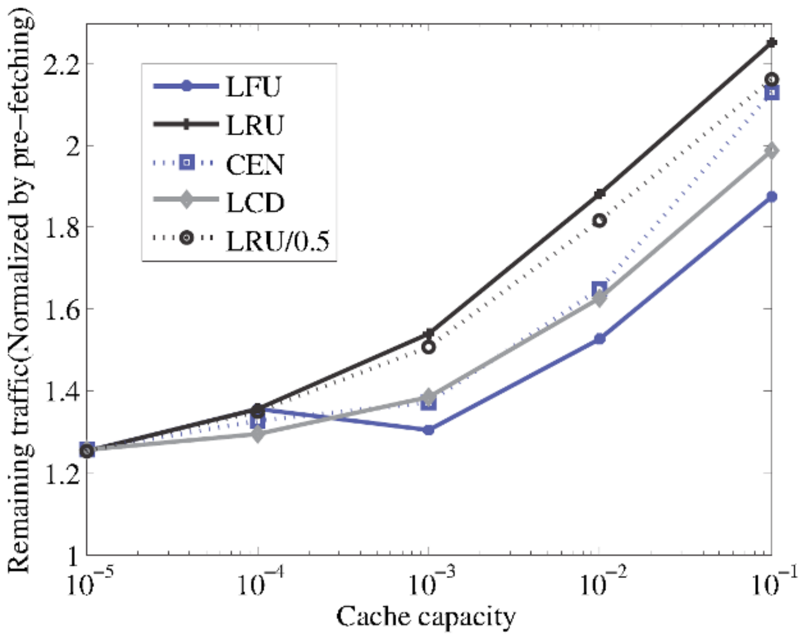
If total cache budget is constant while number of objects increase, caching degrades.

Finding the sweet spot: if number of objects increase, cache budget has to increase, but less than linearly to keep traffic saving constant.

Cache replacement policy



(a) Optimal allocation.



(b) Homogeneous allocation.

Compared to pre-fetching, cache replacement policies perform worse as cache capacity increases.

As expected, LFU performs better than other cache replacement policies.

Outline

- Background
- Optimal cache allocation
- Evaluation
- **Conclusion**

Conclusion

Q: What is the right cache allocation strategy for my network?

A: It depends, but has to be smart enough depending on your specific context.

- BA-like topologies (i.e., interdomain): cache in the core.
- WS-like topologies (i.e., ISP): cache at the edge.
- Larger network requires smart caching strategy.
- More content to be cached => cache placement strategy matters more.
- Uniform popularity => caching in the core
- Heterogeneous popularity => spread caches across network
- LFU fine for small cache budget
- Large cache => smarter cache strategy (e.g., OPT)

ICN: pain or gain?
a data-driven perspective

Yi Sun, Wei Wang, Yang Guo, Bo Deng (CAS), Steve Uhlig (QMUL), Mohamed-Ali Kaafar (INRIA), Alexander Afanasyev (UCLA), Yun Jin (PPLive), Haiyong Xie (USTC)

Outline

- Dataset & methodology
- NDN background
- Evaluation
- Conclusion

Dataset (1)

Topology:

- Traceroute-enabled PPTV clients
- Collected traceroutes over 2 month (10/11 2012) performed from clients (full-mesh)
- 26GB of data, from 1.68 million users
- Sampling: 80k routers, 82 ISPs and 559 cities in China
- Inferred "link latencies" from traceroutes
- Alias resolution for router-level topology

Dataset (2)

Demand:

- 2-week long logs from PPTV servers
- 4.5M users
- 270K content objects
- 26M viewing records

Methodology

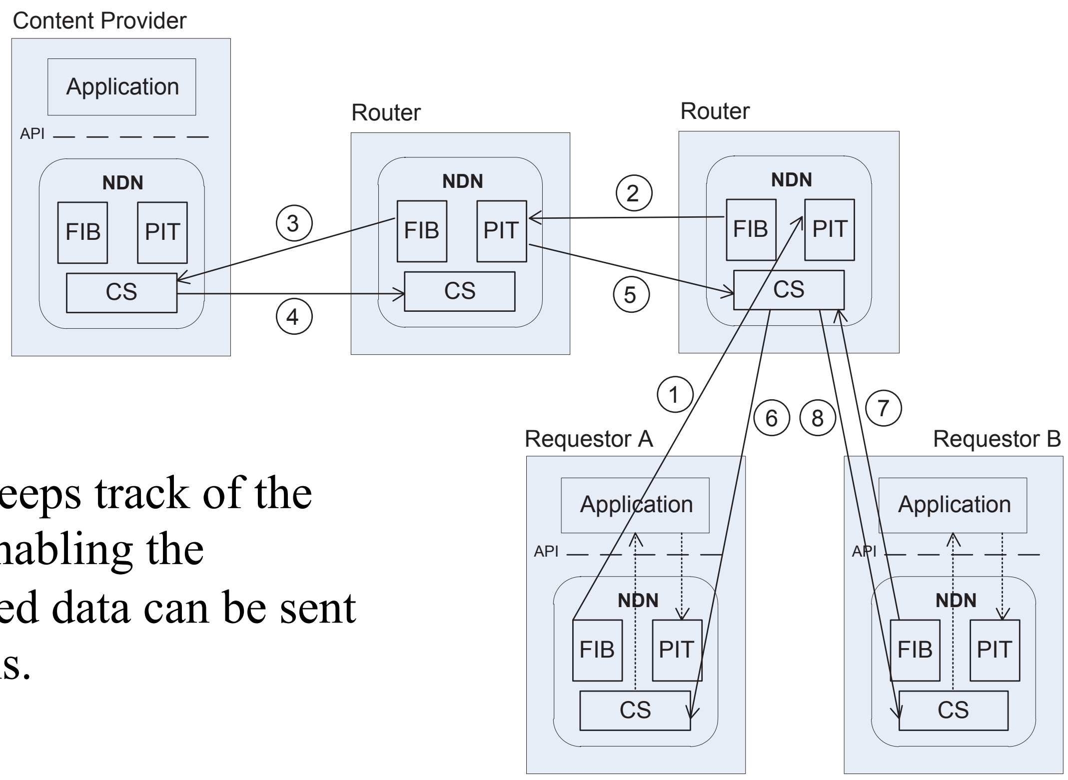
- Simulator: ns-3 ndnSIM
- Build router-level topology between clients based on the traceroutes:
  - Use link delays
  - Link bandwidth set to 622Mbps
- Content originators: 224 PPTV CDN servers that can serve any content
- Compare to pure CDN-based content delivery
- Cache sizes: 1GB, 10GB, 100GB and 1TB
- Cache replacement policies: LRU, LFU and FIFO

Outline

- Dataset & methodology
- **NDN background**
- Evaluation
- Conclusion

NDN background

Content Store (CS): cache the named data packets according to a specific policy (e.g., LRU, LFU, FIFO)



Pending Information Table (PIT): keeps track of the pending forwarded Interest packets, enabling the aggregation of requests, so that returned data can be sent downstream to multiple request origins.

Forwarding Information Base (FIB): used to forward Interest packets towards potential providers of the content.

Assumptions

NDN-related:

- Every router has a CS
- Object is broken down into packets, each cached and transmitted separately
- Object is cached along the path between any CS storing it and origin CDN server

Non-NDN:

- All caches have the same size
- Every CDN server stores ALL content
- Closest CDN server is the origin of a given request

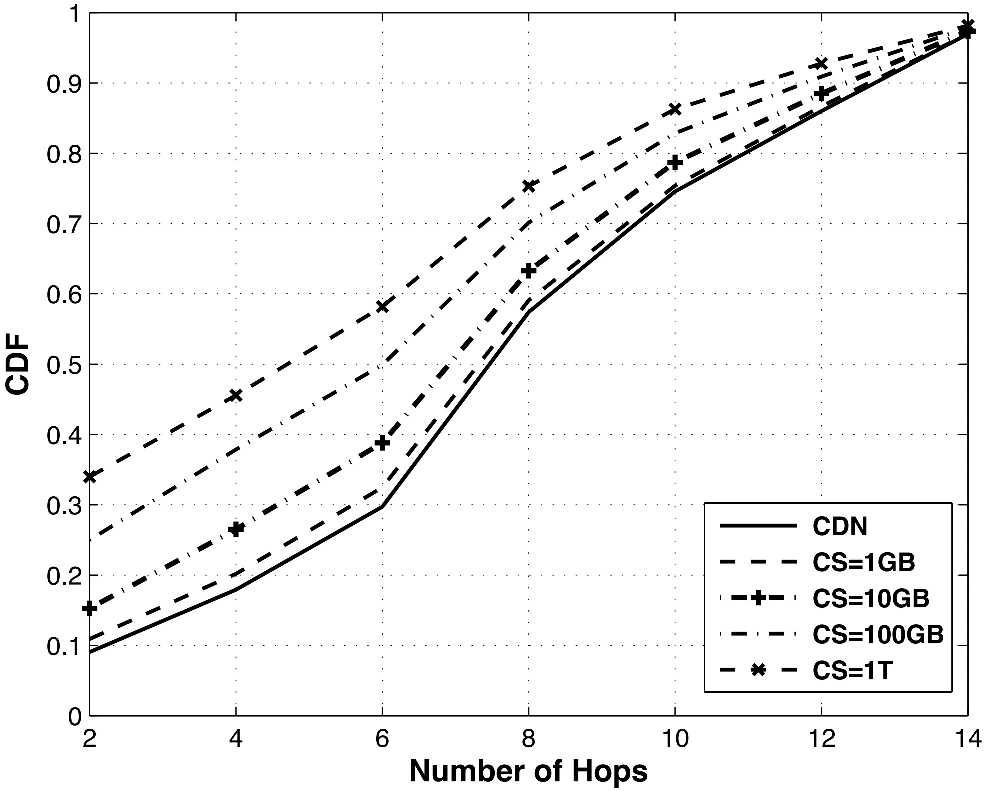
Outline

- Dataset & methodology
- NDN background
- **Evaluation**
- Conclusion

Evaluation metrics

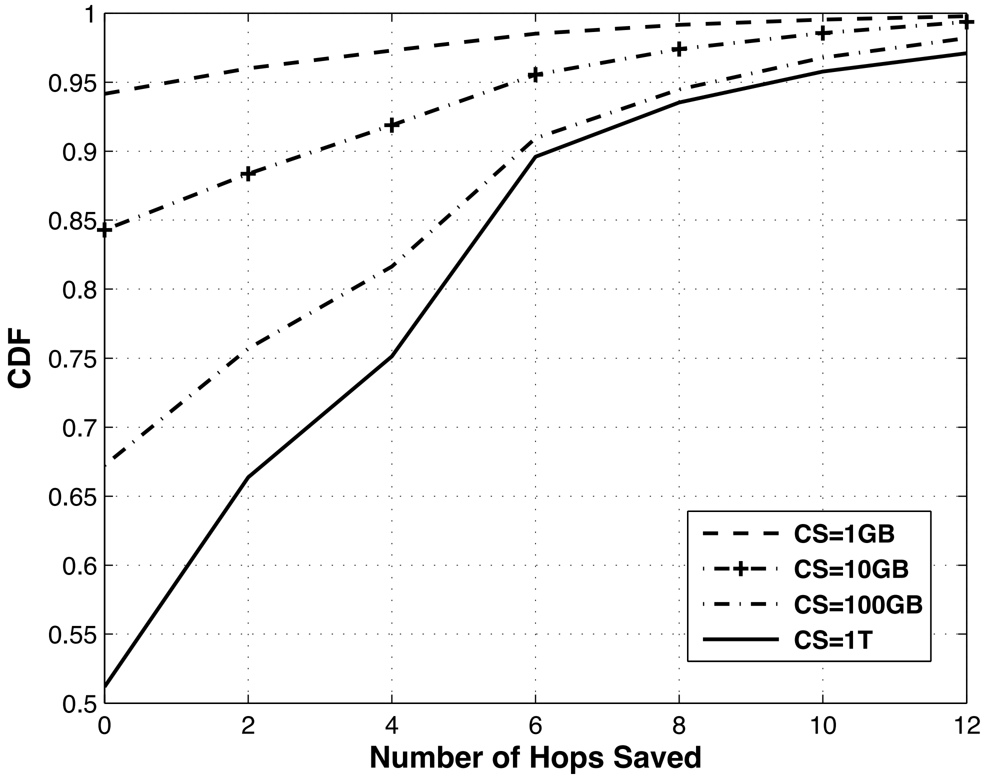
- **Hops of transmission path**: distance between clients and CS hit
- **Hops saved compared to CDN**: difference in hop count between cache and origin CDN server
- **Traffic reduction**: fraction of traffic saved from hop reduction of transmission path
- **Hit rate**: location where hits take place in distance from the CDN server

Cache size: transmission path



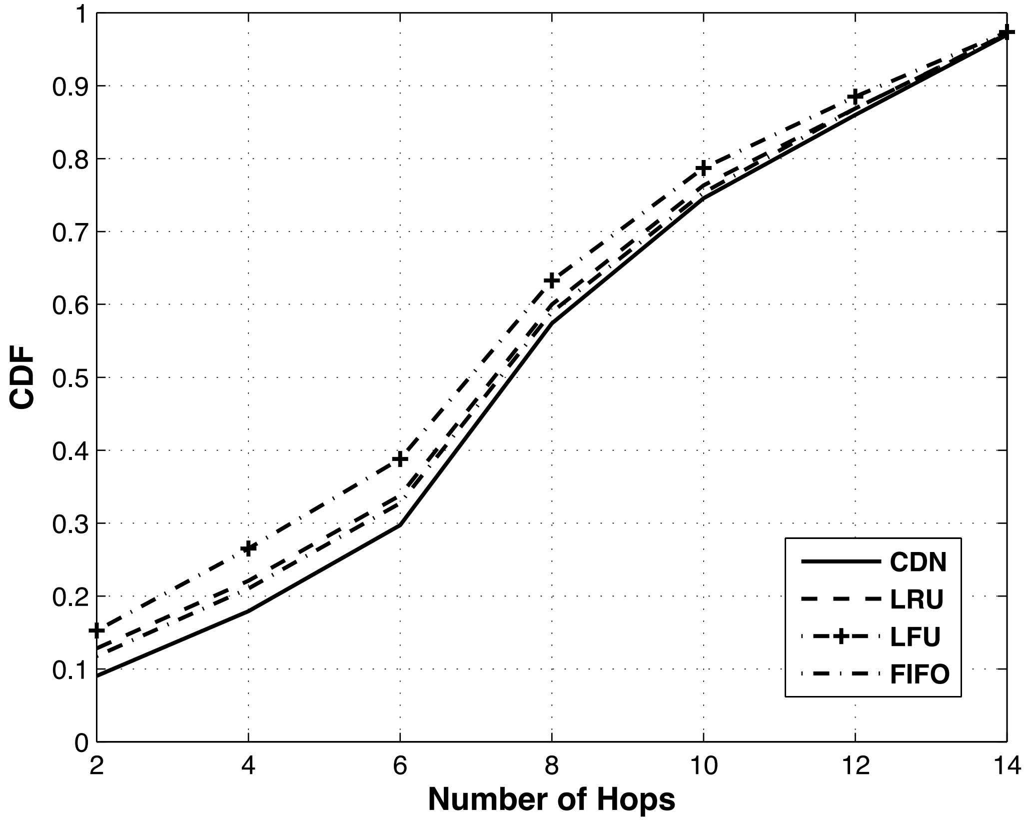
Larger cache size provides diminishing returns in transmission path length.

Cache size: hops saved



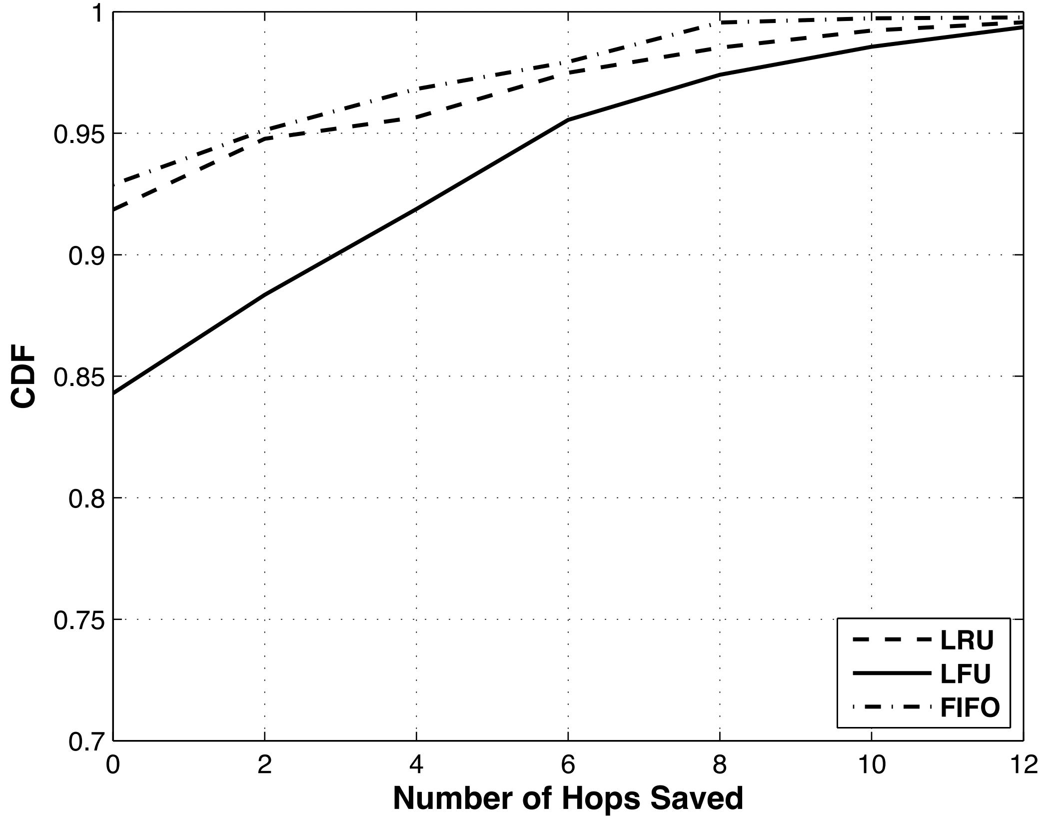
Only large caches provide significant hop savings compared to CDN.

Cache policy: transmission path



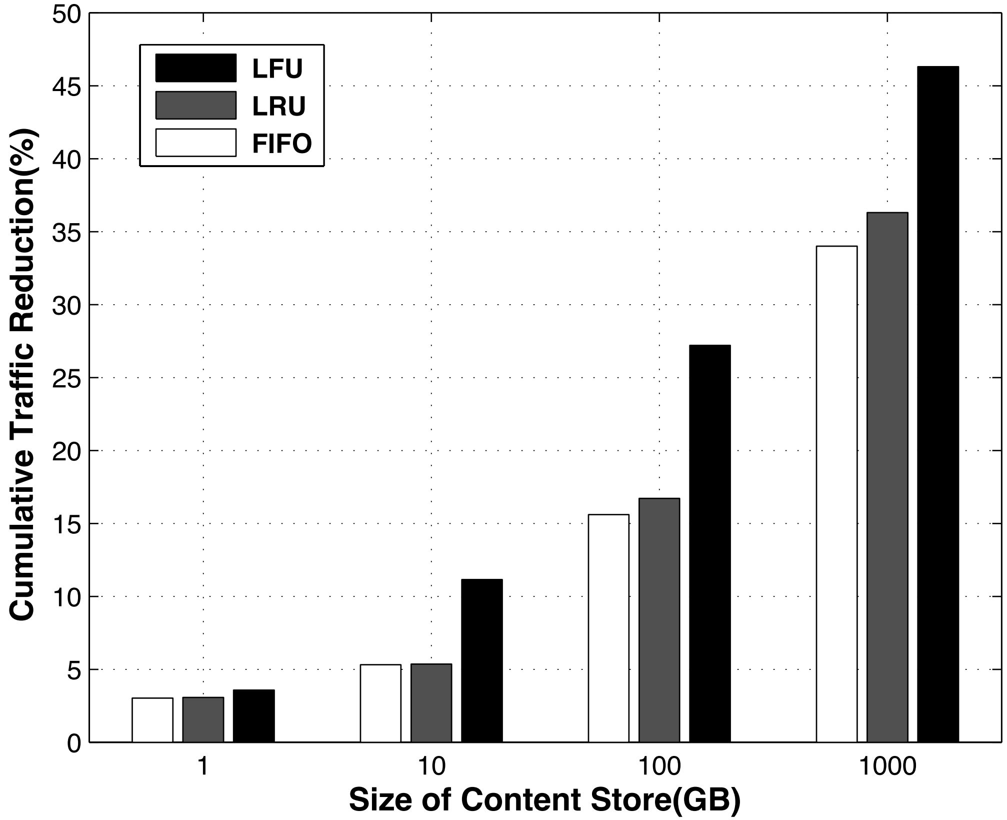
Limited impact of cache replacement policy on transmission path length, compared to CDN.

Cache policy: hops saved



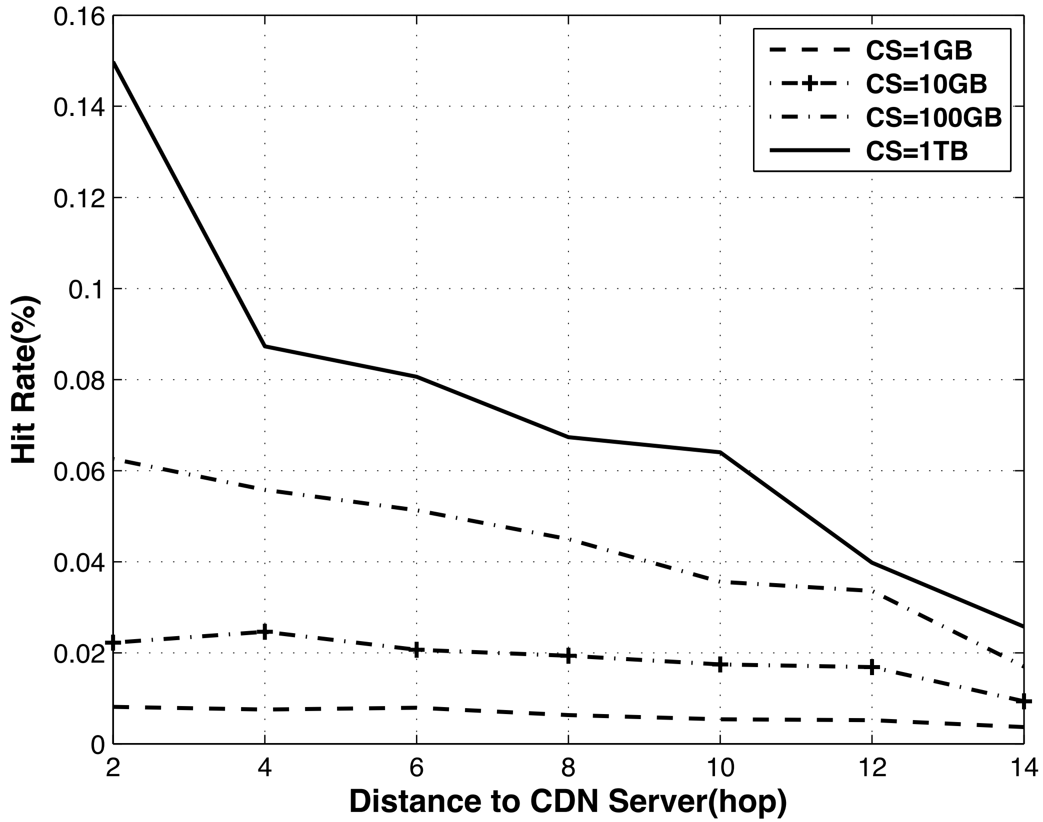
LFU provides best hop savings.

Traffic reduction



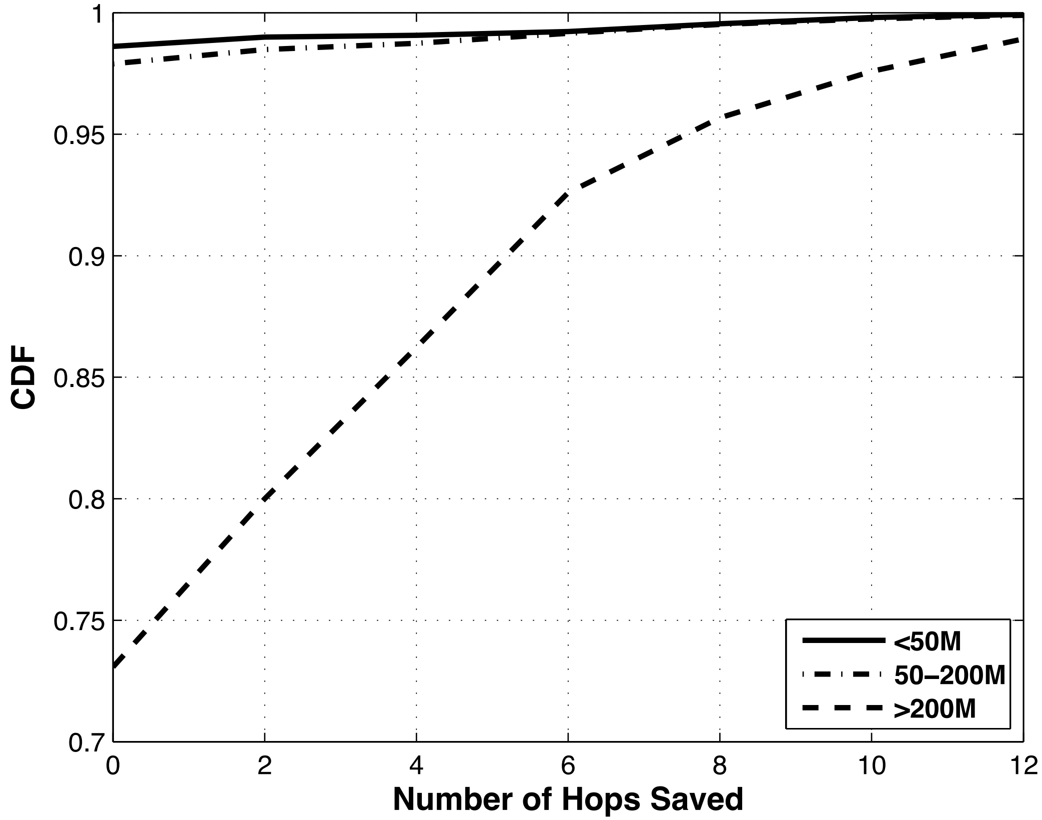
Significant traffic reduction requires large enough caches.

Hit rate



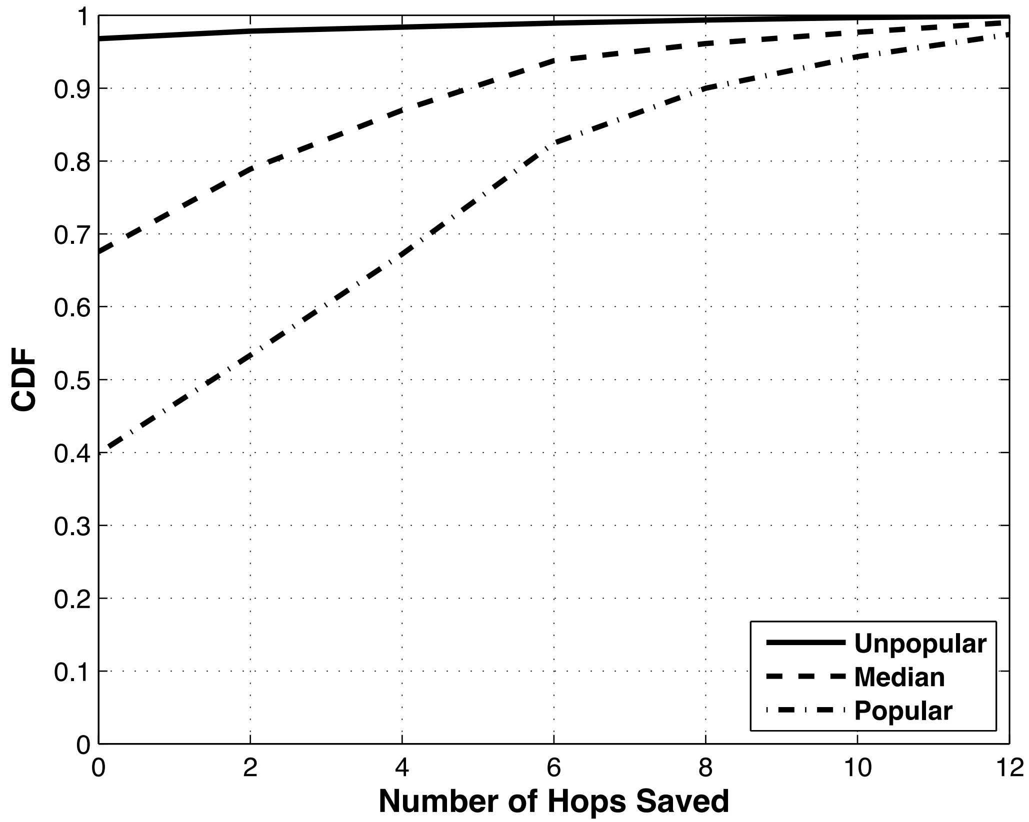
Hit rate of large caches take place close to content origin.

File size



Only large files bring significant transmission path savings.

Content popularity



More popular content see larger transmission path savings.

ICN and QoS

- Throughput: ICN 5 to 20% higher than CDN
- Avg transmission delay: ICN up to 25% lower than CDN
- Packet loss: ICN up to 30% lower than CDN
- Jitter: ICN up to twice larger than CDN

Outline

- Dataset & methodology
- NDN background
- Evaluation
- **Conclusion**

ICN: pain or gain?

- Strengths
  - Shorter transmission path (1.5 hop on avg)
  - Traffic saving (27% with 10GB cache and LFU)
  - Improved QoS
- Unclear
  - Recovery cost: 50 days for 1GB caches, 3.5 years for 100GB
- Weaknesses
  - Limited gain with small caches
  - Not worth caching small/unpopular/unskewed content
  - Jitter

=> Compared to a CDN, ICN may or may not look promising. However, very popular content is likely to benefit from it, as well as content that requires QoS.