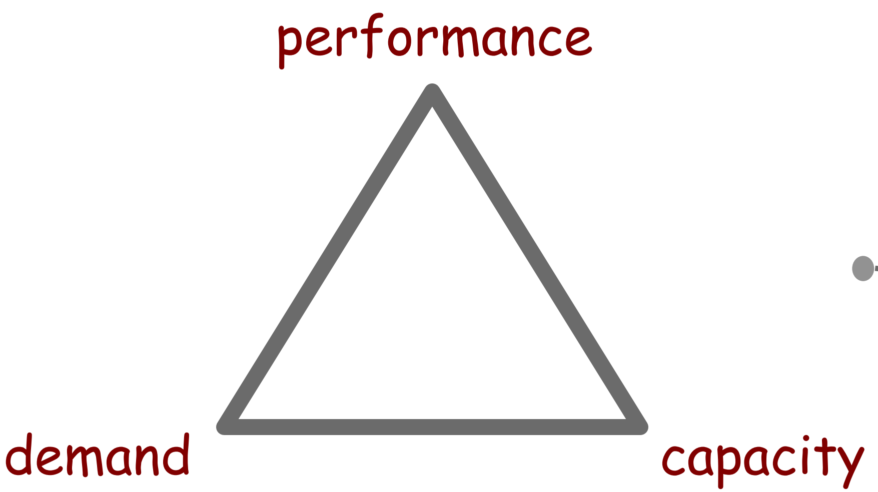
Trading off memory for bandwidth in a content-centric Internet

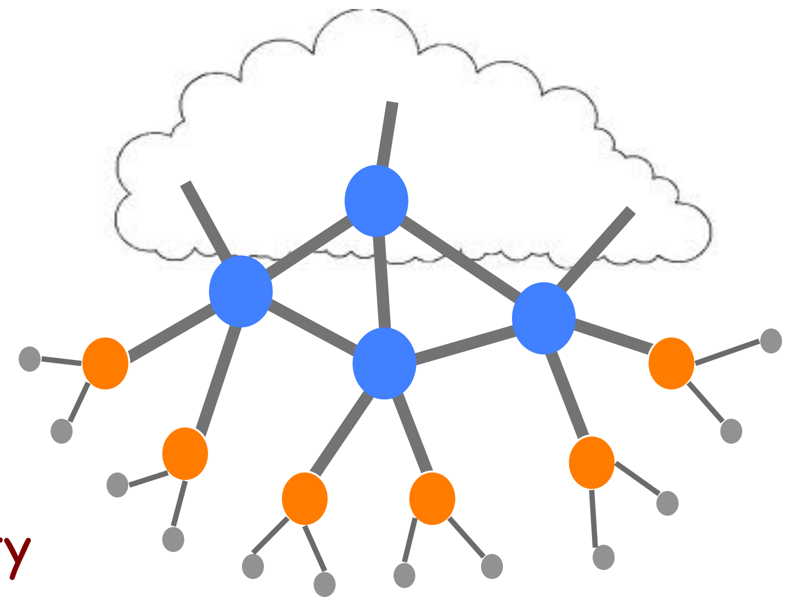
Jim Roberts (Telecom ParisTech)

Talk at LIG, 7 Feb 2019

Engineering the Internet

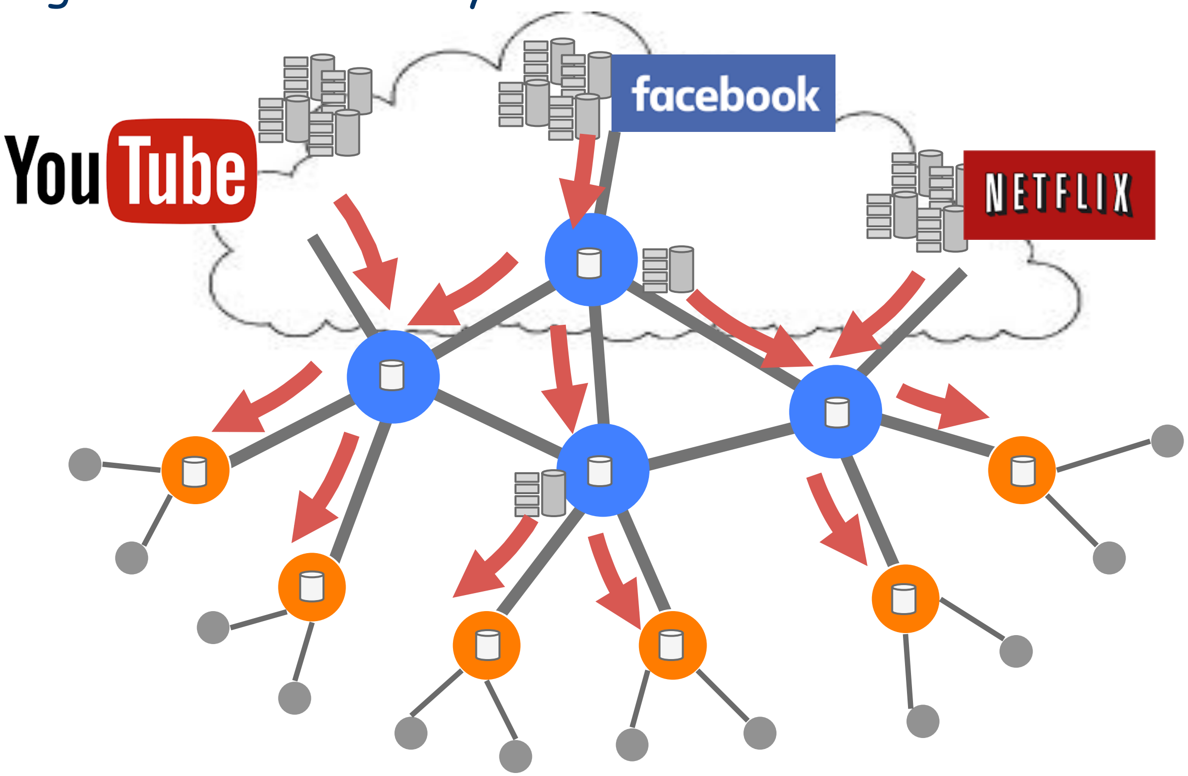
- understanding the relation between demand, capacity and performance
- to design a cost efficient network that satisfies quality of service requirements





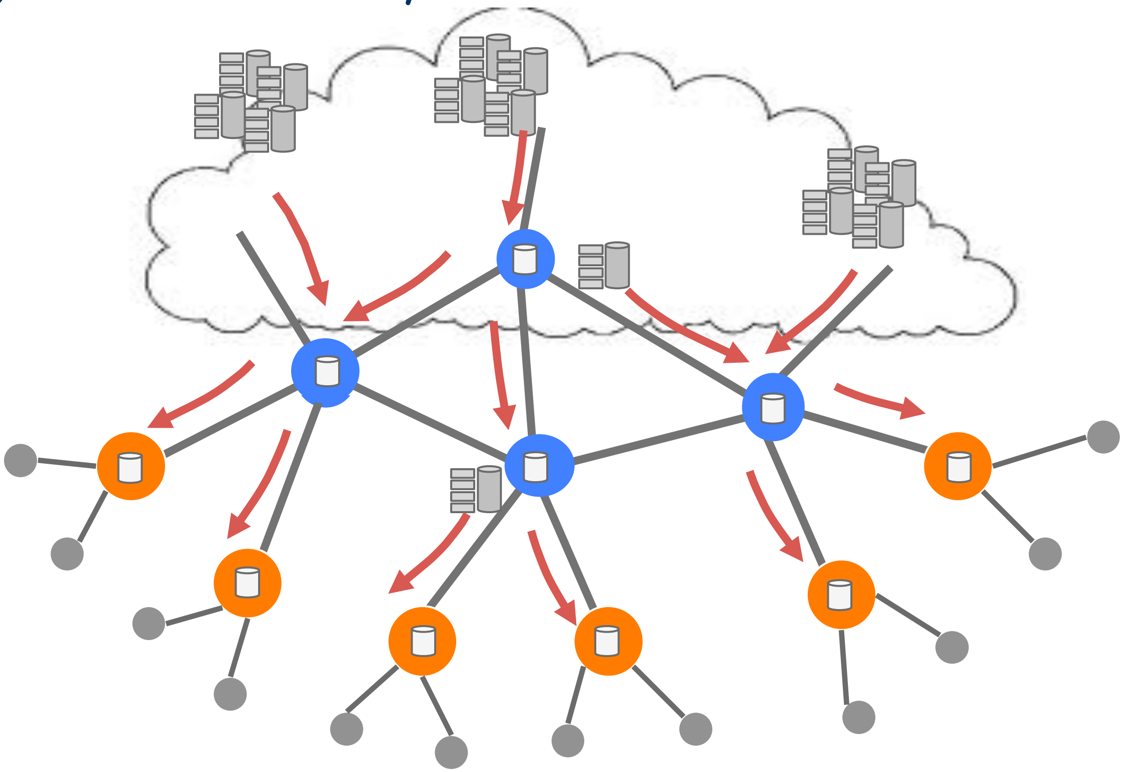
From connecting endpoints to content delivery

- 96% of traffic is content
  - web, file sharing, social networks, video streaming,...
- demand depends on content placement
  - caching realizes a memory for bandwidth trade-off



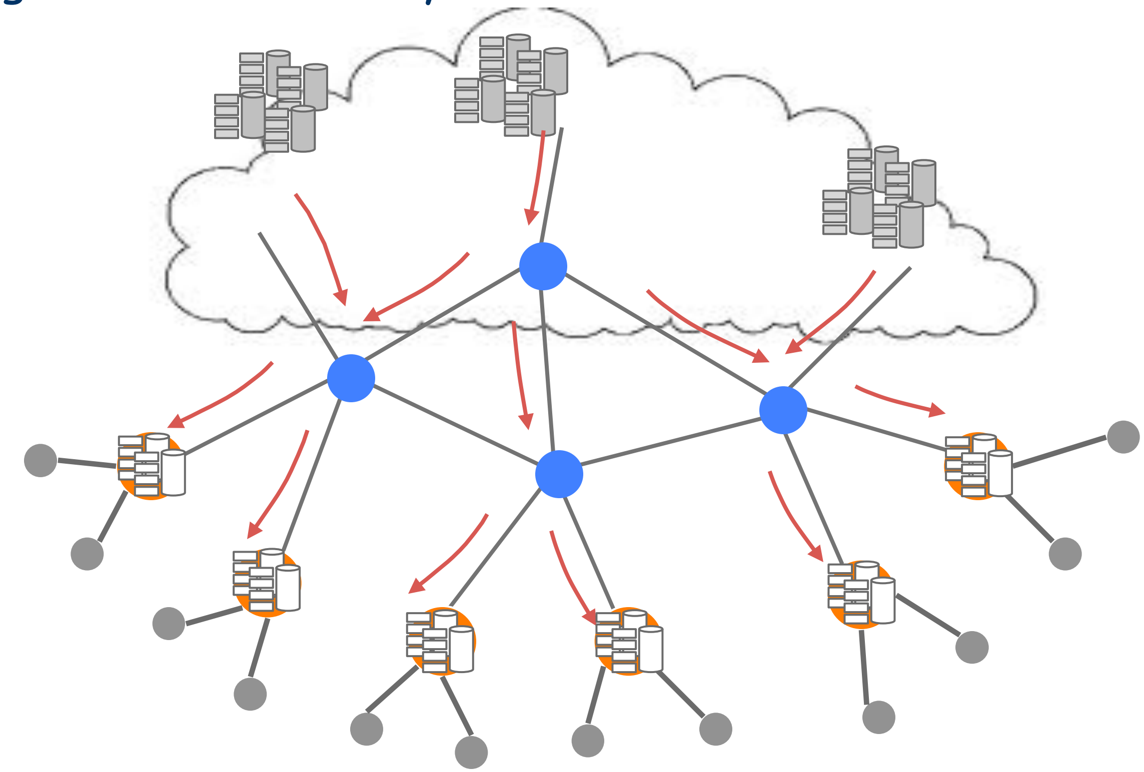
From connecting endpoints to content delivery

- 96% of traffic is content
  - web, file sharing, social networks, video streaming,...
- demand depends on content placement
  - caching realizes a memory for bandwidth trade-off



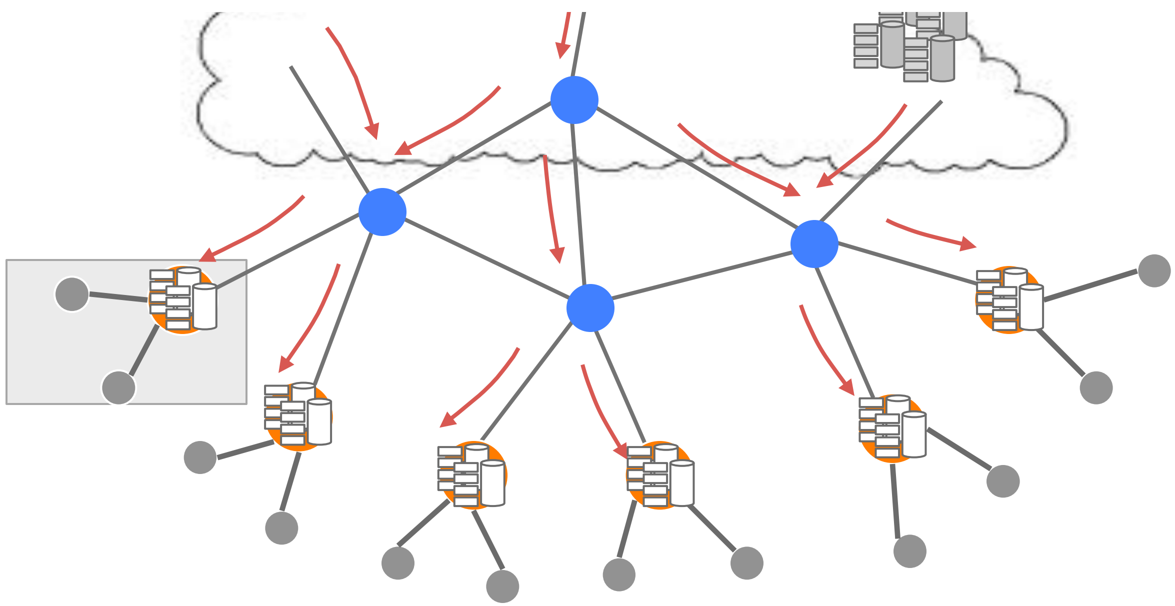
From connecting endpoints to content delivery

- 96% of traffic is content
  - web, file sharing, social networks, video streaming,...
- demand depends on content placement
  - caching realizes a memory for bandwidth trade-off



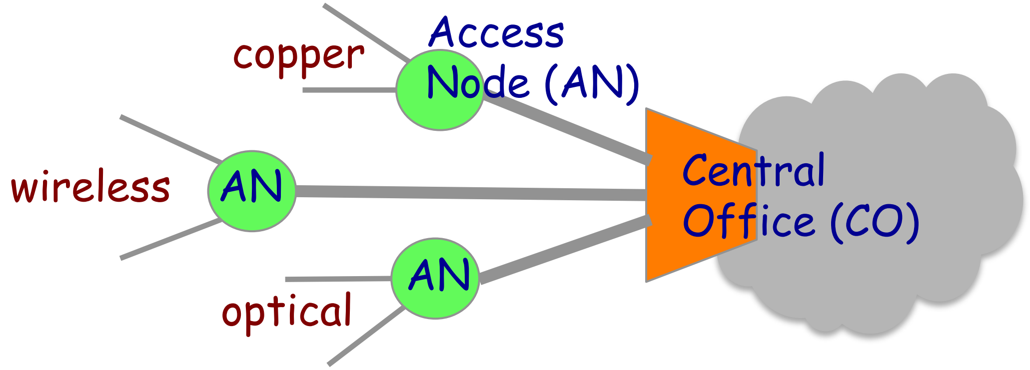
From connecting endpoints to content delivery

- 96% of traffic is content
  - web, file sharing, social networks, video streaming,...
- demand depends on content placement
  - caching realizes a memory for bandwidth trade-off
- caching "at the edge" brings the optimal trade-off
  - but where is the edge?



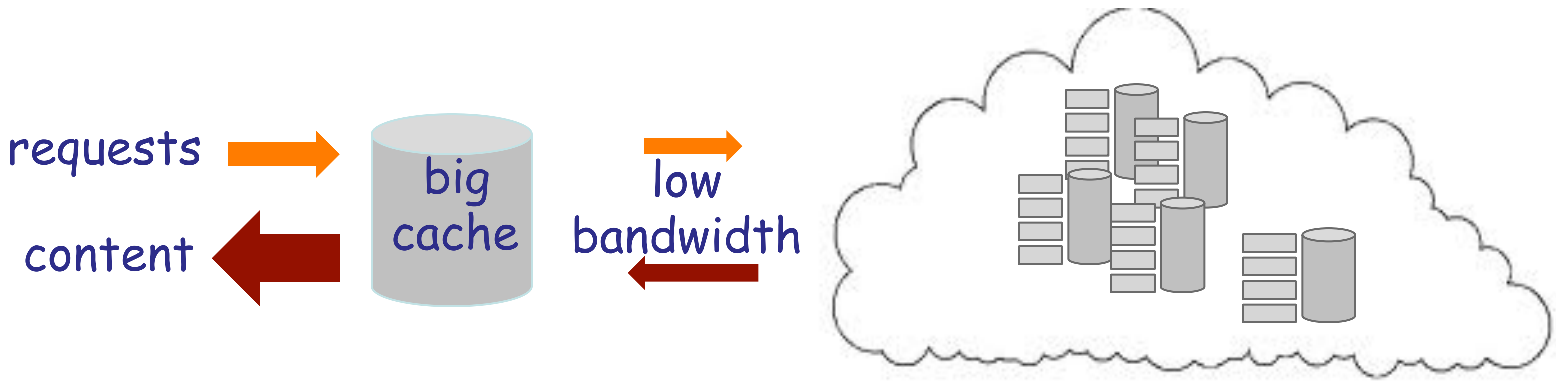
From connecting endpoints to content delivery

- 96% of traffic is content
  - web, file sharing, social networks, video streaming,...
- demand depends on content placement
  - caching realizes a memory for bandwidth trade-off
- caching "at the edge" brings the optimal trade-off
  - but where is the edge?
- QoS (latency, throughput) is not an issue
  - made equally good by adequate sizing



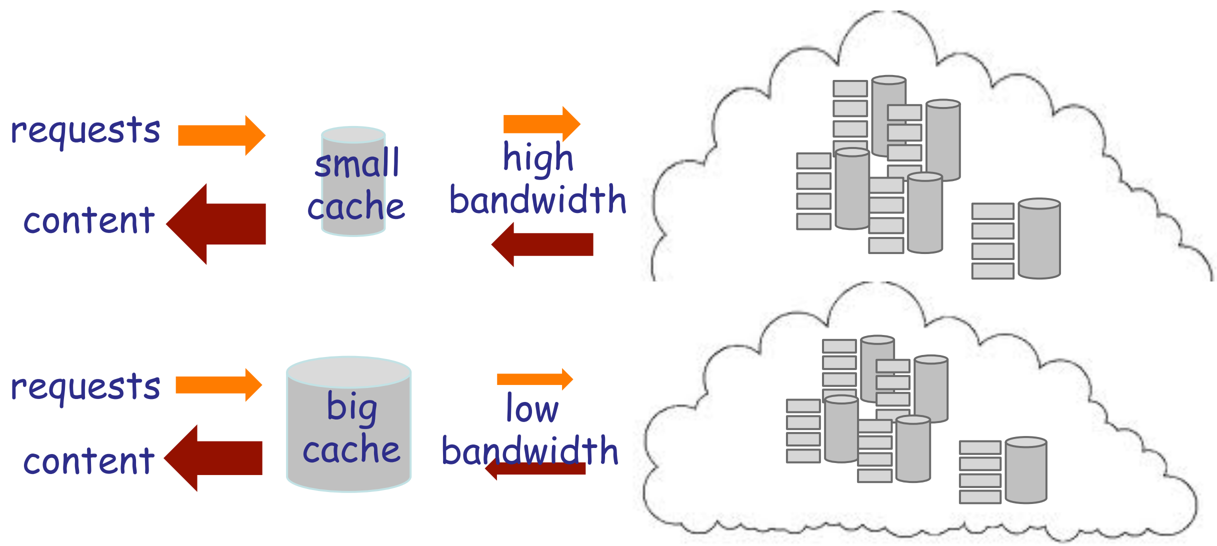
An optimal memory-bandwidth trade-off

- preferred cache size depends on overall cost of memory (cache capacity) and bandwidth (including routers)
  - more memory means less traffic and therefore less bandwidth



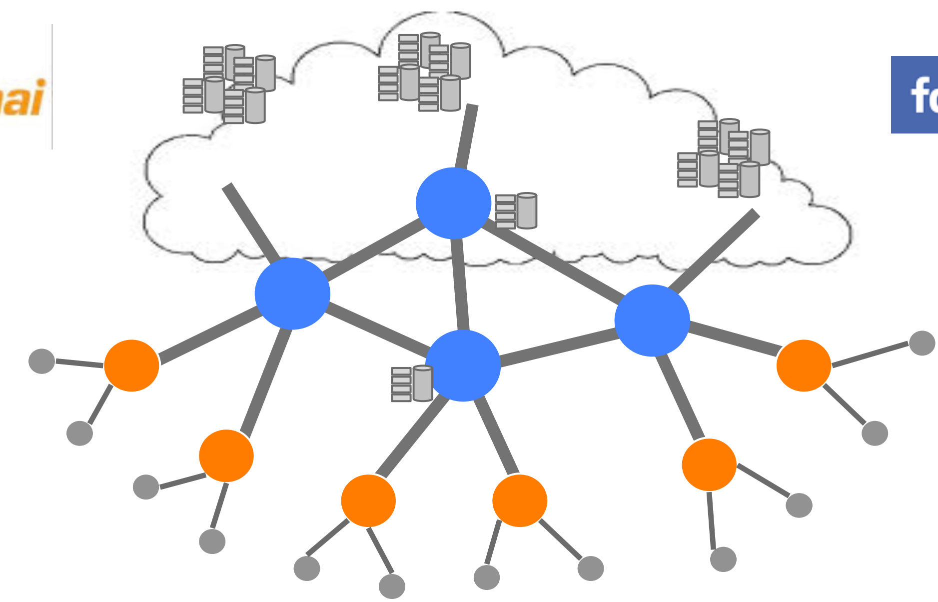
An optimal memory-bandwidth trade-off

- preferred cache size depends on overall cost of memory (cache capacity) and bandwidth (including routers)
  - more memory means less traffic and therefore less bandwidth
- an infrastructure provider (bandwidth and storage) would seek to optimize the trade-off
  - but must do this in a complex business environment



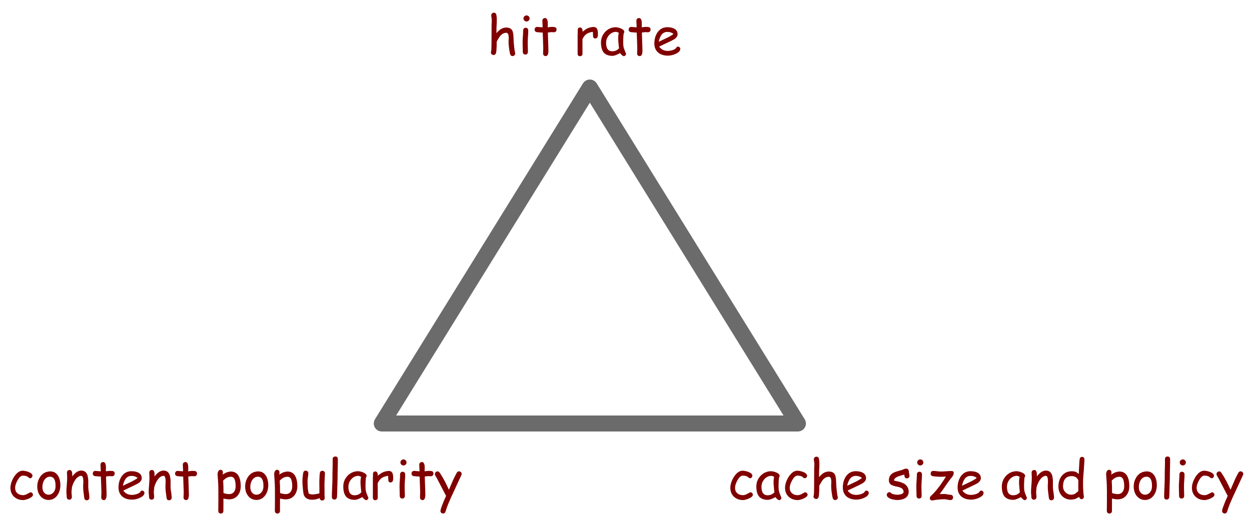
The content delivery business

- since the birth of the web, ISPs have unsuccessfully sought to realize a favourable memory for bandwidth trade-off
- instead, most content is delivered using overlay content delivery networks (eg, Akamai, but also Google, Facebook, Netflix,...)
- who optimize their own costs and performance while preserving their profitable business models



Outline

1. cache hit rate performance
2. optimizing the memory bandwidth trade-off



Internet content mix

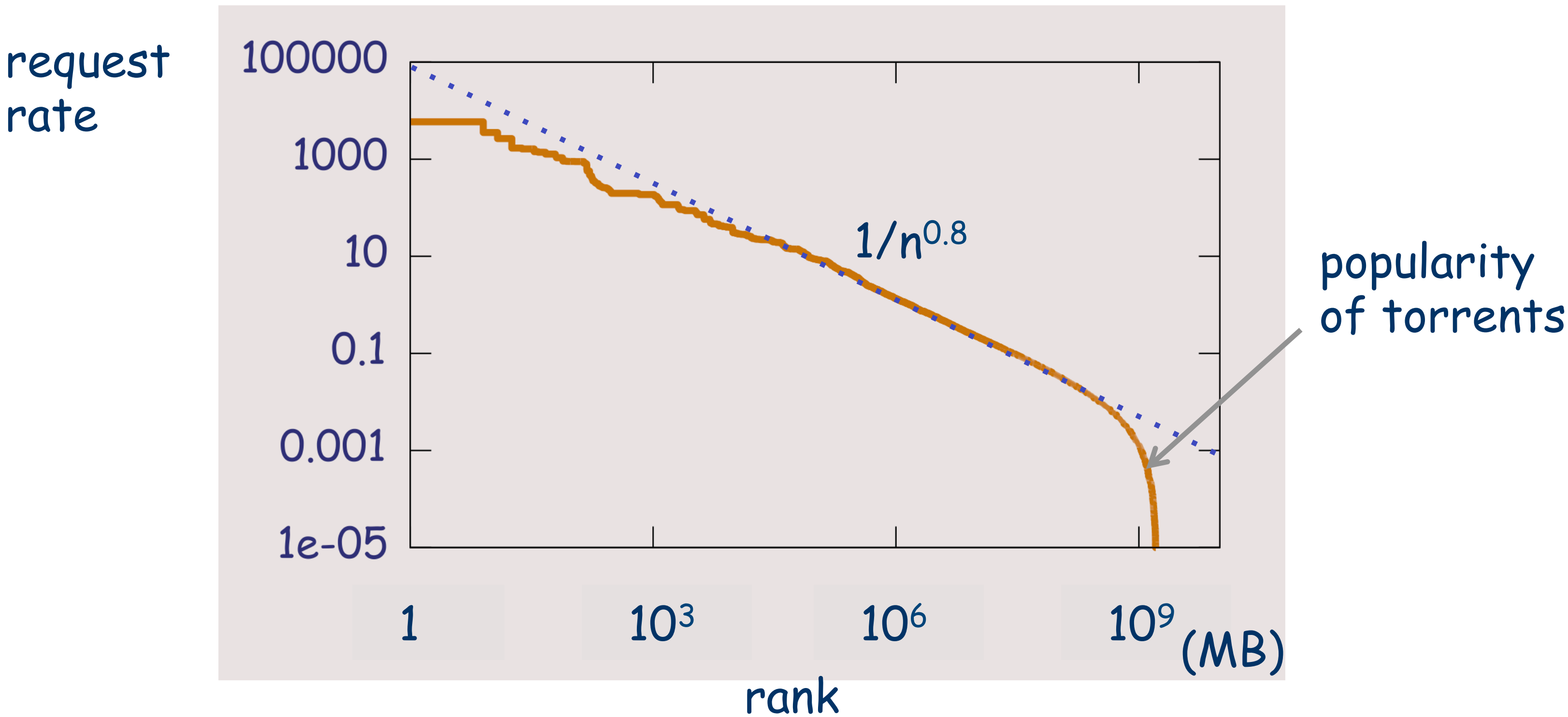
- Cisco VNI: "96% of traffic is content transfer"
- web, file sharing, user generated content, video on demand, social networks
- billions of objects, petabytes of content!

	objects	size	volume	share
web	10^{11}	10 KB	1 PB	17%
file sharing	10^{5}	10 GB	1 PB	3%
UGC	10^{8}	10 MB	1 PB	11%
VoD	10^{4}	100 MB	1 TB	47%
...				

(NB. very rough, order of magnitude estimates)

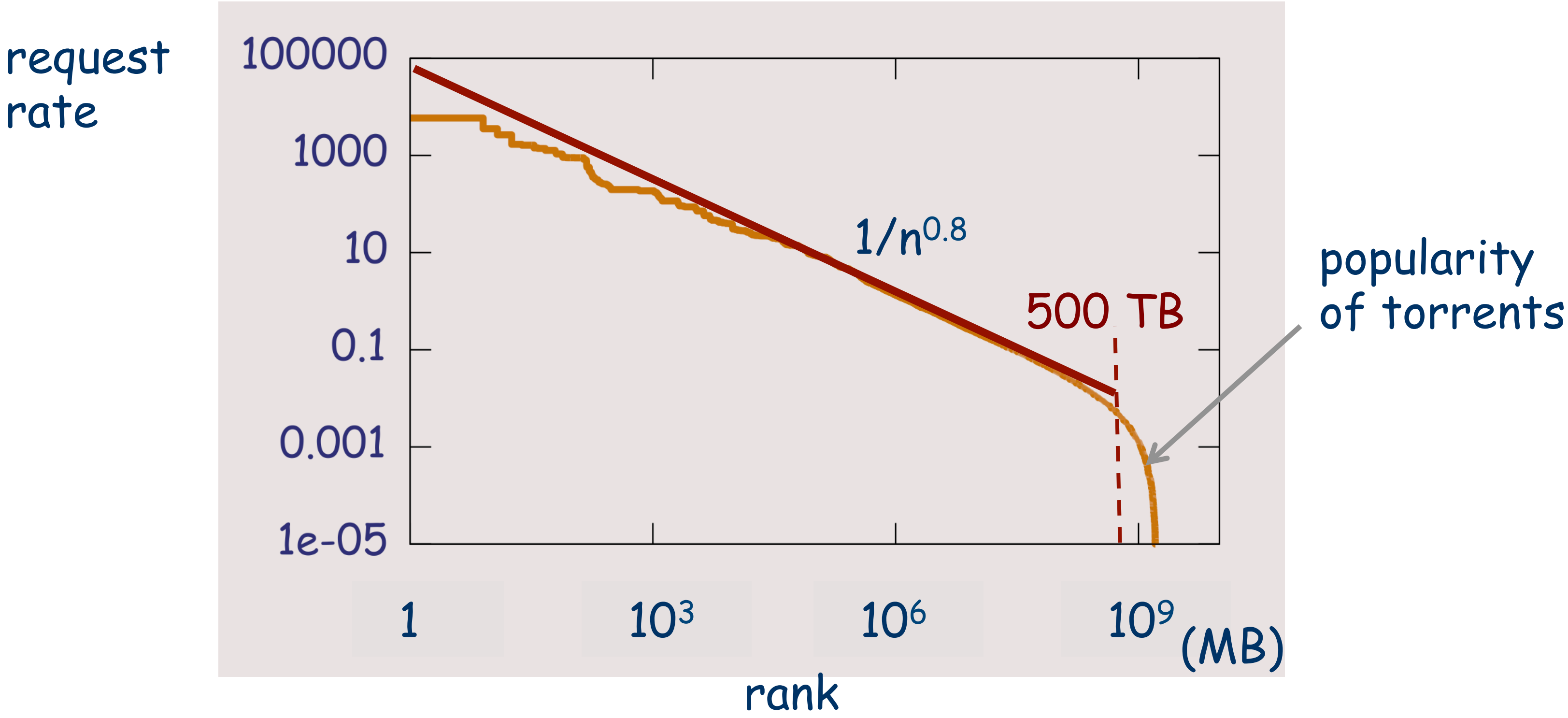
Content popularity

- popularity is measured by request arrival rate per byte
  - eg, chunk downloads by BitTorrent peers
- measurements reveal popularity decreases as a power law:
  - request rate of n^{th} most popular chunk $\propto 1/n^{\alpha}$
  - a generalized Zipf law; typically, $\alpha \approx 0.8$



Content popularity

- cache performance depends significantly on catalogue size
- our guesstimates
  - 1 PB for all content (YouTube, web, social networks, P2P, ...)
  - 1 TB for a VoD catalogue

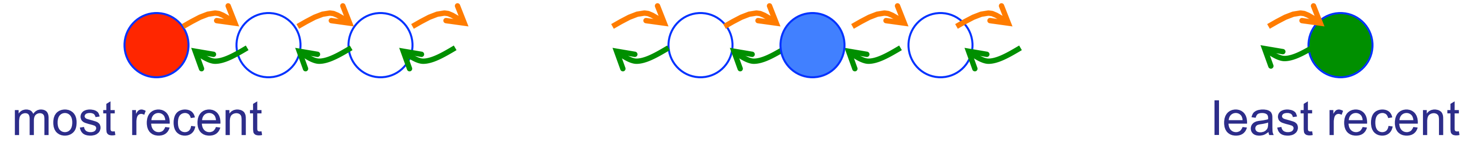


Content popularity

- cache performance depends significantly on catalogue size
- our guesstimates
  - 1 PB for all content (YouTube, web, social networks, P2P, ...)
  - 1 TB for a VoD catalogue
- for illustration, assume Zipf(.8) popularity
  - $q_i \propto 1 / i^{.8}$ and $\sum_{1 \le i \le N} q_i = 1$,
  - N and chunk size set so catalogue size is 1 TB or 1 PB
  - (for large systems, performance depends on catalogue size in bytes and not on chunk or object size)
- the independent reference model (IRM)
  - request is for i with probability q_i independently of all past requests
  - as if requests occur as stationary Poisson streams of rate q_i

Hit rate and cache policy – stationary demand

- "ideal" cache
  - cache holds most popular items
  - hit rate, $h(C,N) = \sum_{i \le C} q_i$
    $\approx (C/N)^{(1-\alpha)} = h(C/N)$
- least recently used (LRU)



Hit rate and cache policy – stationary demand

- "ideal" cache
  - cache holds most popular items
  - hit rate, $h(C,N) = \sum_{i \le C} q_i$

    $\approx (C/N)^{(1-\alpha)} = h(C/N)$
- least recently used (LRU)
  - "characteristic time" approx.

    $h_i = 1 - \exp(-q_i t_c)$ where t_c

    satisfies $C = \sum h_i$ and

    $h = \sum_{i \le N} q_i h_i$

Characteristic time approximation

~~(Che, Tung and Wang, 2002)~~

The "Fagin approximation", 1977 *

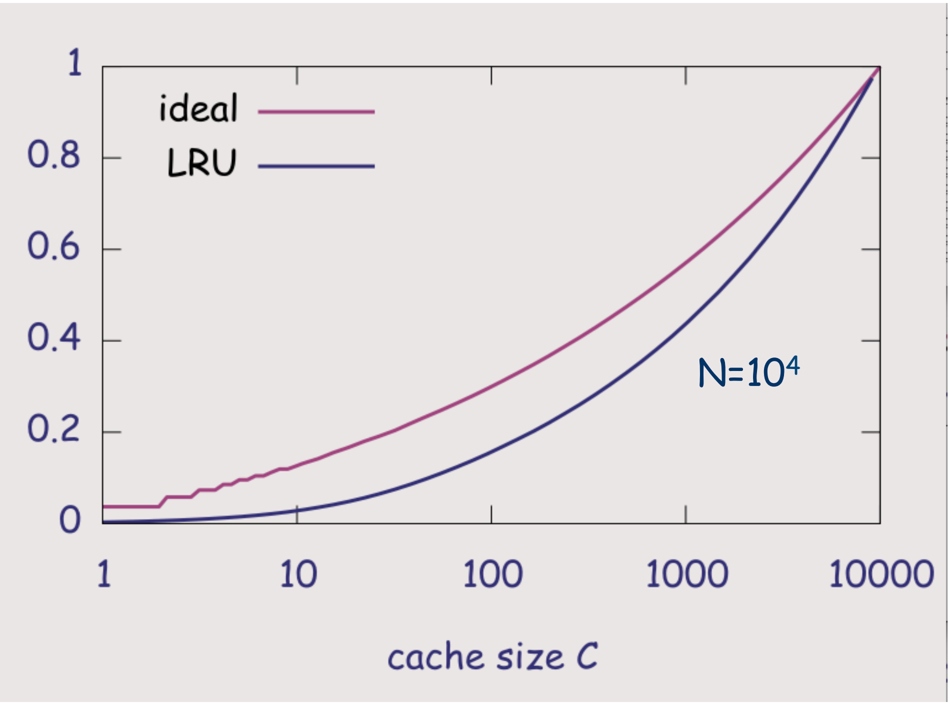
- "characteristic time" T_C is time for C different objects to be requested

- assume random variable T_C is approximately deterministic, $T_C \sim t_C$
- then, hit rate for object n is $h_i = 1 - \exp(-q_i t_C)$
- now, $C = \sum_i 1\{\text{object } i \text{ is in cache}\}$
- taking expectations, $C = \sum_i h_i = \sum_i (1 - \exp(-q_i t_C))$
- solving numerically for t_C yields h_i
- approximation justified in (Fricker et al, 2012)

* R. Fagin. 1977. Asymptotic Miss Ratios over Independent References. J. Comput. System Sci. 14, 2 (1977), 222–250.
(thanks to Christian Berthet)

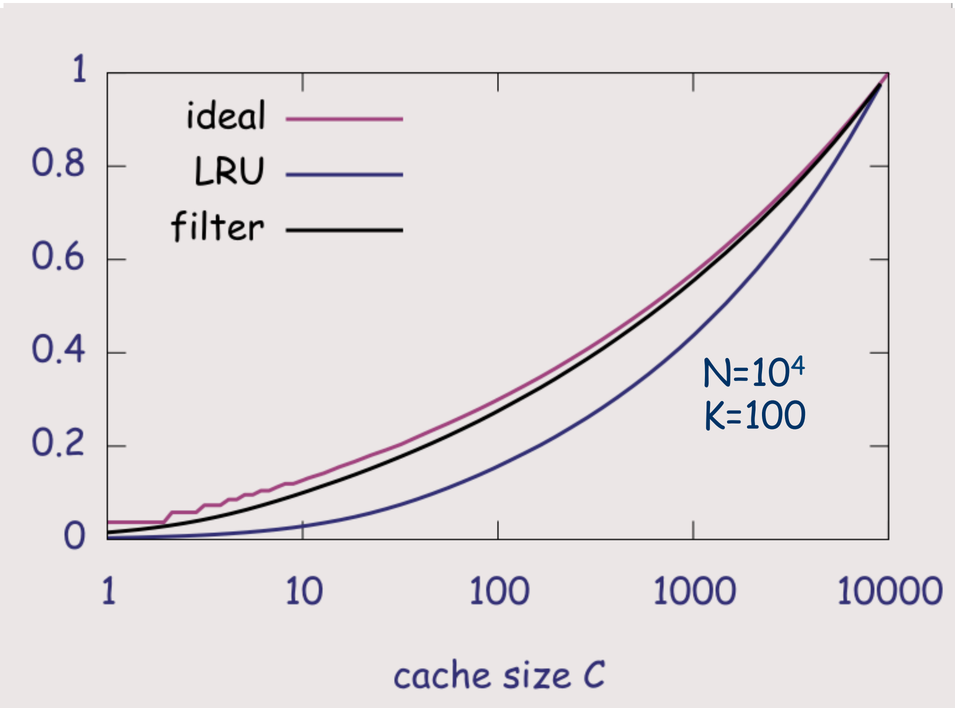
Hit rate and cache policy – stationary demand

- "ideal" cache
  - cache holds most popular items
  - hit rate, $h(C,N) = \sum_{i \le C} q_i$
    $\approx (C/N)^{(1-\alpha)} = h(C/N)$
- least recently used (LRU)
  - "characteristic time" approx.
    $h_i = 1 - \exp(-q_i t_c)$ where t_c
    satisfies $C = \sum h_i$
  - a significant performance penalty for small caches



Hit rate and cache policy – stationary demand

- cache with "pre-filter"
  - on cache miss, only add new item if included in previous K requests
  - $h_i^{(n+1)} = (1 - \exp(-q_i t_c)) \times$
    $(h_i^{(n)} + (1-h_i^{(n)})(1 - (1-q_i)^K))$
  - where $h_i^{(n)}$ is hit rate of n^{th} request for item i
  - for stationary demand $h_i^{(n+1)} = h_i^{(n)} = h_i$, $C = \sum h_i$ yields t_c
- but pre-filters slow reactivity to popularity changes ...



Time varying popularity

- many items are short-lived, cf. [Traverso 2013]
  - we assume the most popular have shortest lifetimes
- IRM assumption is not appropriate when demand is low
  - eg, the first request for a new item is necessarily a miss

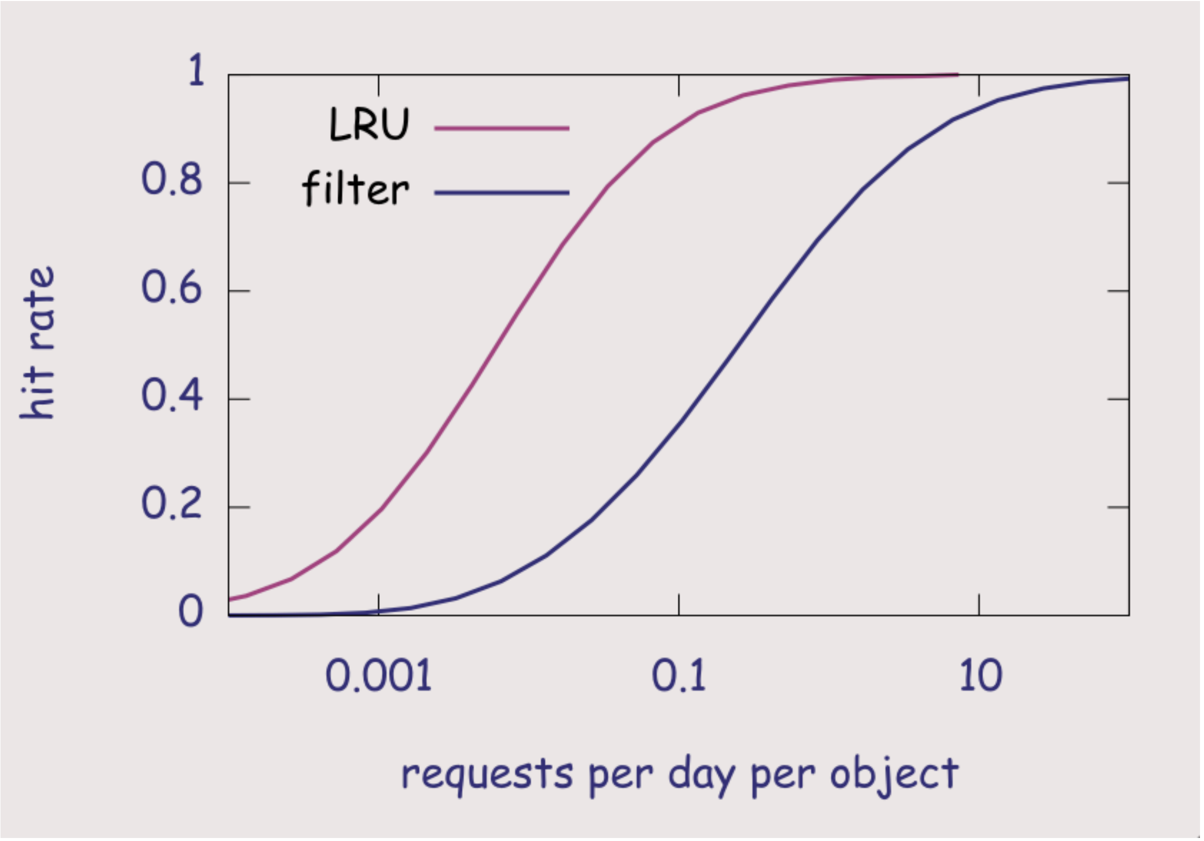
lifetime interval	proportion of items	mean lifetime
0-2 days	.5 %	1.1 days
2-5 days	.8 %	3.3 days
5-8 days	.5 %	6.4 days
8-13 days	.8 %	10.6 days
> 13 days (or < 10 reqs)	97.4 %	1 year

Hit rates with finite lifetimes

- model after [Wolman 1999]: item i always has popularity q_i but changes after each lifetime
- LRU hit rate with mean item lifetime τ_i
  - first request after change must miss
  - $h_i = (1 - \exp(-q_i t_c)) \times (q_i \tau_i / (1 + q_i \tau_i))$
- LRU hit rate with pre-filter
  - recall: $h_i^{(n+1)} = (1 - \exp(-q_i t_c)) \times (h_i^{(n)} + (1-h_i^{(n)})(1 - (1-q_i)^K))$ $(*)$
  - assume item i changes after n^{th} request with probability $1 - \eta_i$ where $\eta_i = q_i \tau_i / (1 + q_i \tau_i)$
  - then, $h_i = h_i^{(1)} (1 - \eta_i) + h_i^{(2)} \eta_i (1 - \eta_i) + h_i^{(3)} \eta_i^2 (1 - \eta_i) + \cdots$
  - multiply $(*)$ by η_i^n and add eventually yields h_i

Impact of time-varying popularity

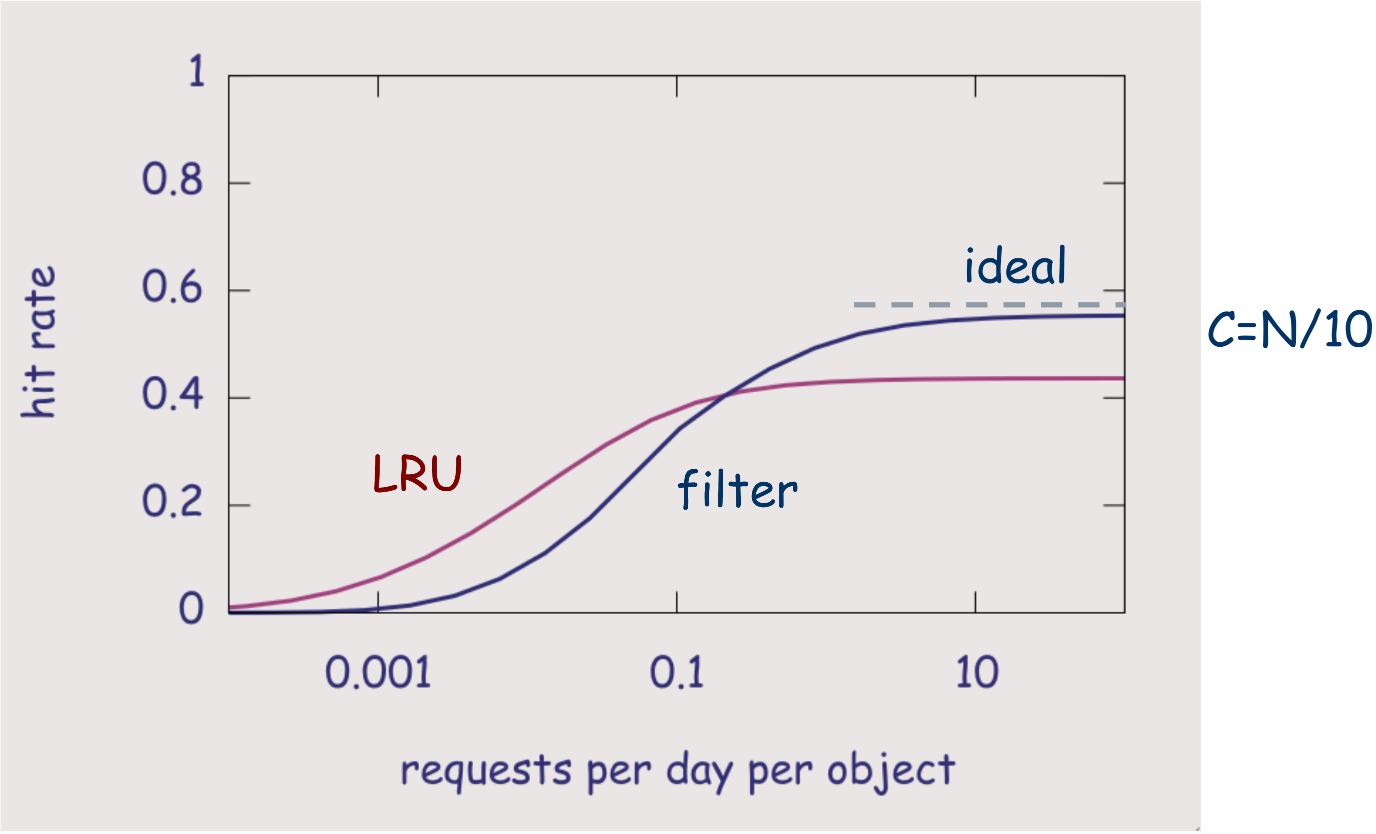
- hit rate depends on demand since first requests in lifetime always miss (first for LRU, first 2 for LRU with pre-filter)



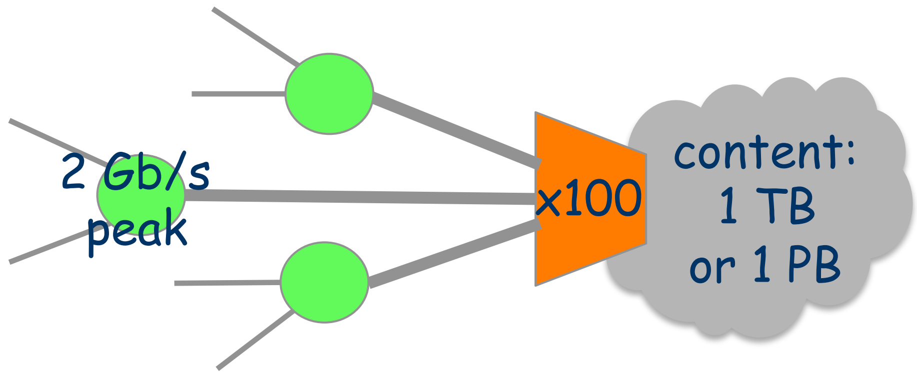
full capacity (C=N)

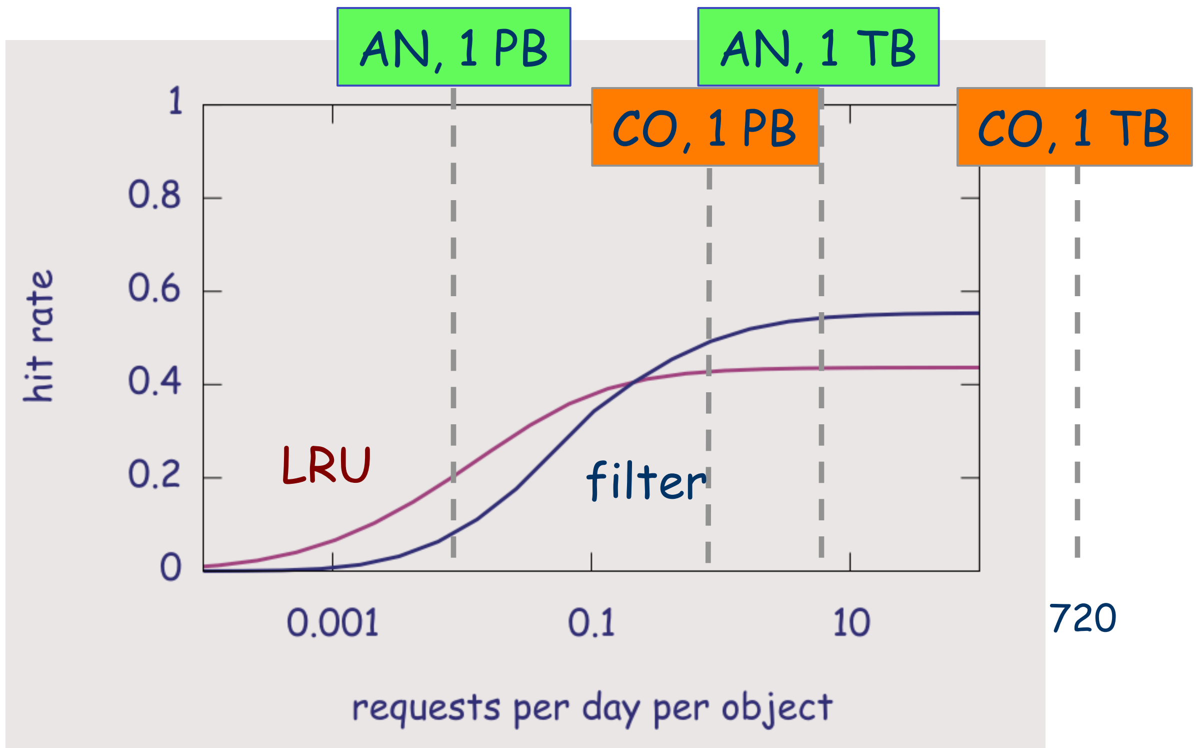
Impact of time-varying popularity

- hit rate depends on demand since first requests in lifetime always miss (first for LRU, first 2 for LRU with pre-filter)

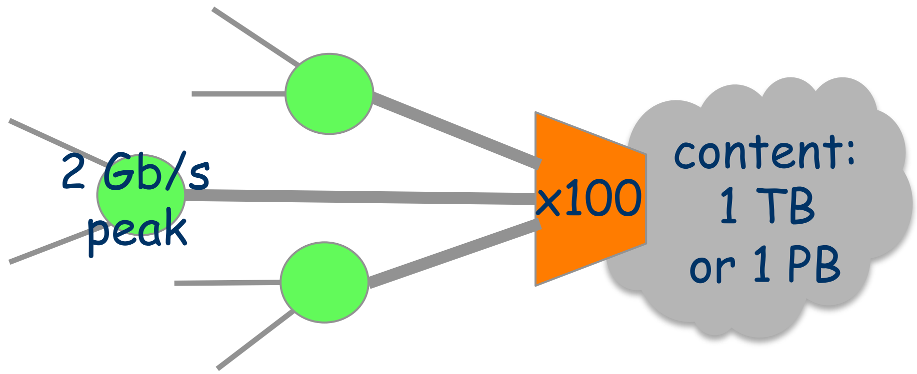


Application to access network





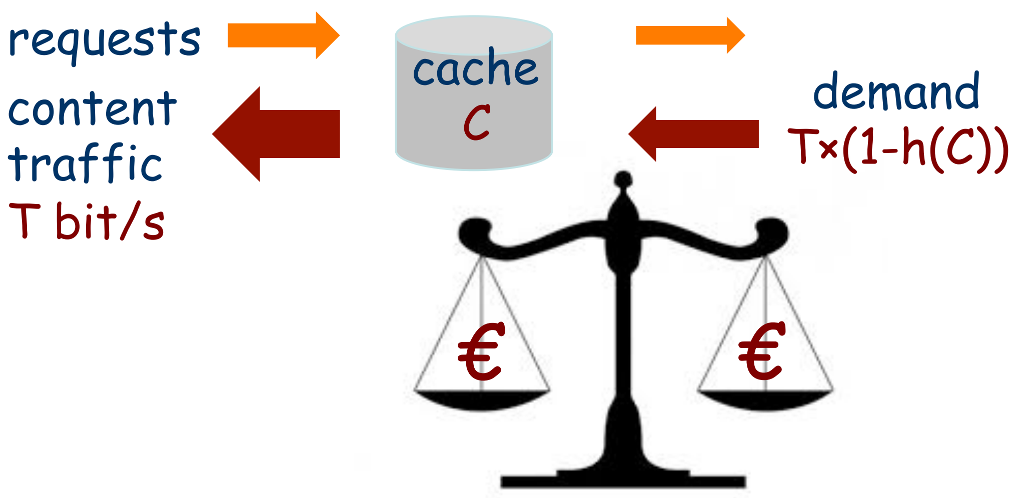
Implications



- we need proactive caching at AN and below (eg base stations)
  - ie, network must proactively upload the most popular items
- proactive caching needs some function to predict popularity
  - by being informed of requests from a large user population
  - and applying data analytics...
- content providers can measure popularity, ISPs typically can't
  - user preference data is highly sensitive and jealously guarded

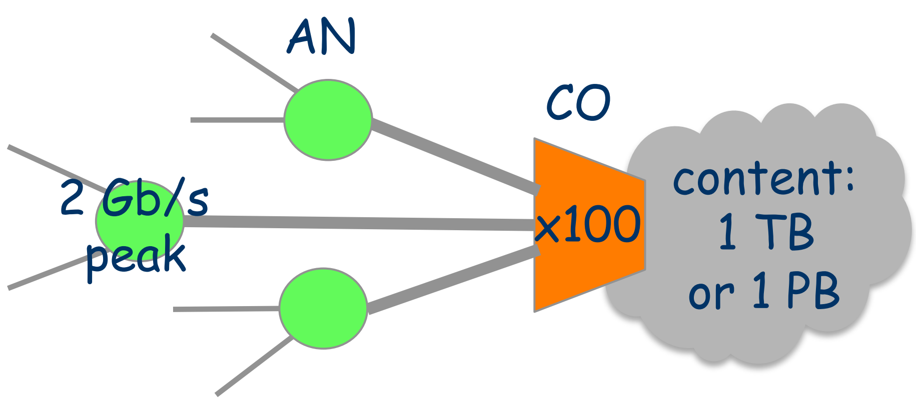
Outline

1. cache hit rate performance
2. optimizing the memory bandwidth trade-off



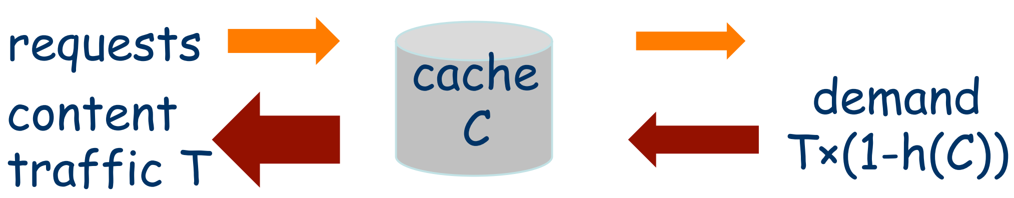
Evaluating the trade-off

- cache at Central Office (~200 Gb/s) or Access Node (~2 Gb/s)
- caches have ideal performance (eg, proactive or pre-filter)
- popularity is Zipf(.8) with a catalogue of 1 TB or 1 PB



Evaluating the trade-off

- overall cost of cache and bandwidth is
  - $\Delta(C) = K_b(T \times (1-h(C))) + K_m(C)$
  - where T is download traffic, $h(C)$ is hit rate, $K_b(D)$ and $K_m(C)$ are cost functions for demand D and cache C
- to simplify, assume linear cost functions
  - $K_b(D) = k_b \times D$, $K_m(C) = k_m \times C$
  - where k_b and k_m are marginal costs of bandwidth and memory
- consider normalized cost $\delta(c)$ for relative cache size $c = C/N$
  - $\delta(c) = \Delta(C)/k_m N = \Gamma \times (1-h(c)) + c$ (ie, $\delta(1) = 1$ and $\delta(0) = \Gamma$)
  - where $\Gamma = k_b T/k_m N$ is ratio of max bandwidth cost to max cache cost



Normalized cost

- $\Delta(C)$ is combined cost of memory and bandwidth
- $\Delta(C) = K_b\big(T\times(1-h(C,N))\big) + K_m(C)$

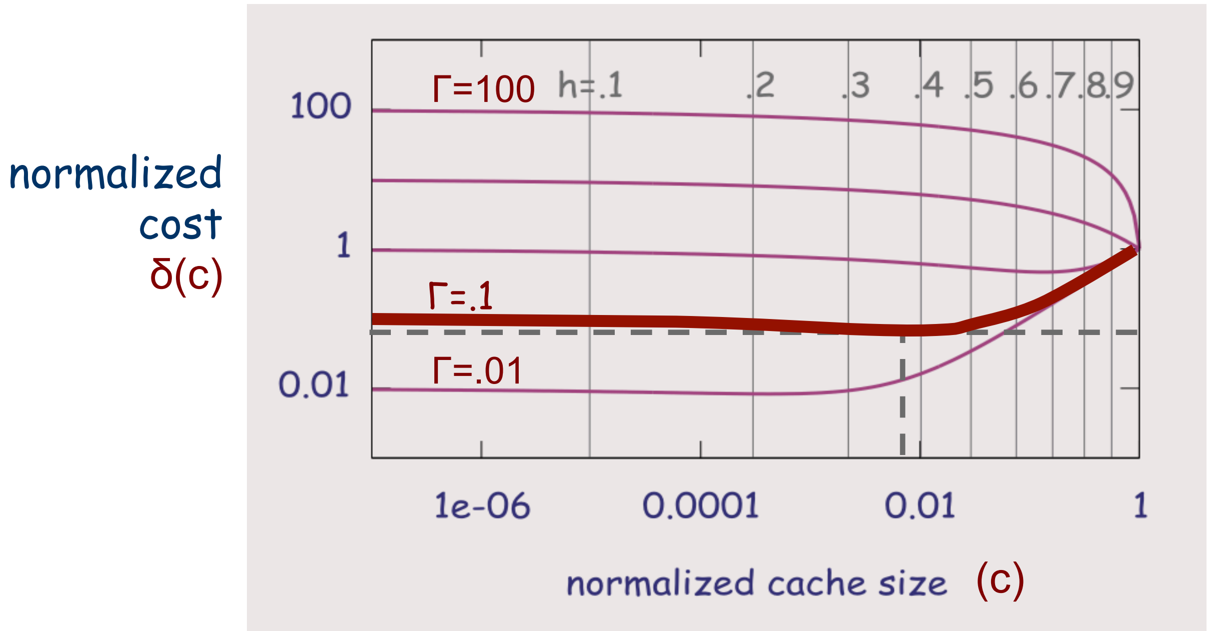
  $= k_b \times T\times(1-h(C,N)) + k_m\, C$
- let $\delta(c) = \Delta(C)/k_m N$ and write $h(C,N) = h(C/N) = h(c)$
- $\delta(c)$ is combined cost normalized by maximum storage cost
- $\delta(c) = k_b T \,/\, k_m N \times (1-h(c)) + c$

  $= \Gamma\,(1-h(c)) + c$ where
- $\Gamma = k_b T \,/\, k_m N$ = max bandwidth cost / max cache cost

- optimal trade-off maximizes $\Delta(C)$ and $\delta(c)$

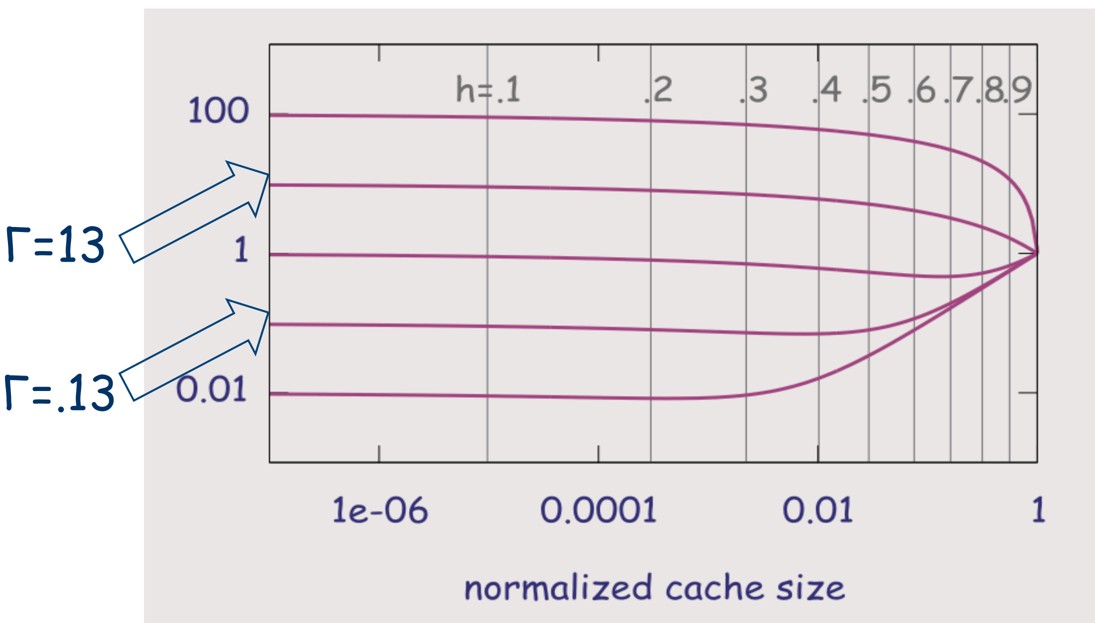
Normalized cost v normalized cache size

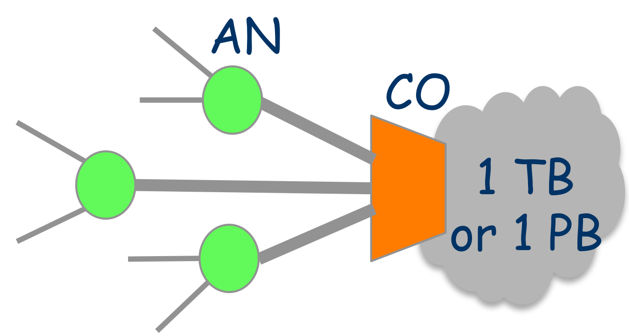
- normalized cost $\delta(c) = \Gamma\times(1-h(c)) + c = \Gamma\times(1-c^{0.2}) + c$
- where $\Gamma = k_b T / k_m N$ is max bandwidth cost / max cache cost
- if $\Gamma \geq 5$, max cache is optimal (c=1, ie, C=N)
- if $\Gamma < 5$, there is optimal cache size for $0<c<1$
  - eg, for Γ = .1, min cost for c=.008, h(c)=.37 for gain ≈ 30%



Cost and demand guesstimates

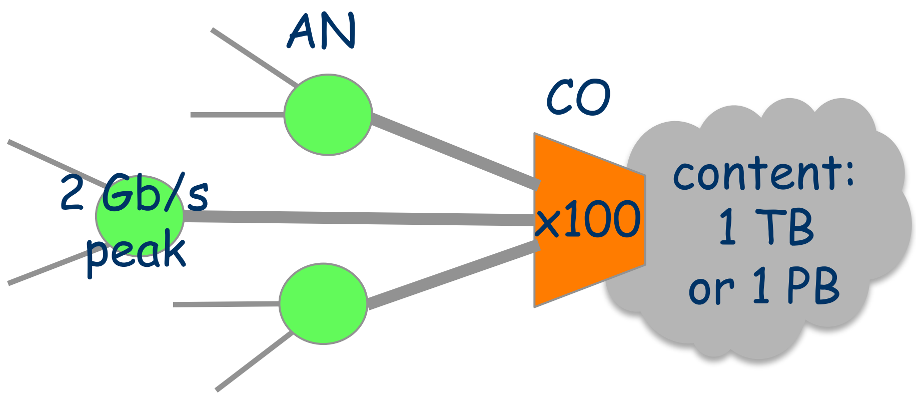
- cost of bandwidth: k_b = \$2 per Mb/s per month
- cost of memory: k_m = \$.03 per GB per month
- if N = 1 PB and T = 200 Gb/s, $\Gamma = k_b T / k_m N \approx 13$ (CO, large N)
- if N = 1 PB and T = 2 Gb/s, $\Gamma \approx .13$ (AN, large N)
- if N = 1 TB and T = 2 Gb/s, $\Gamma \approx 130$ (AN, small N)





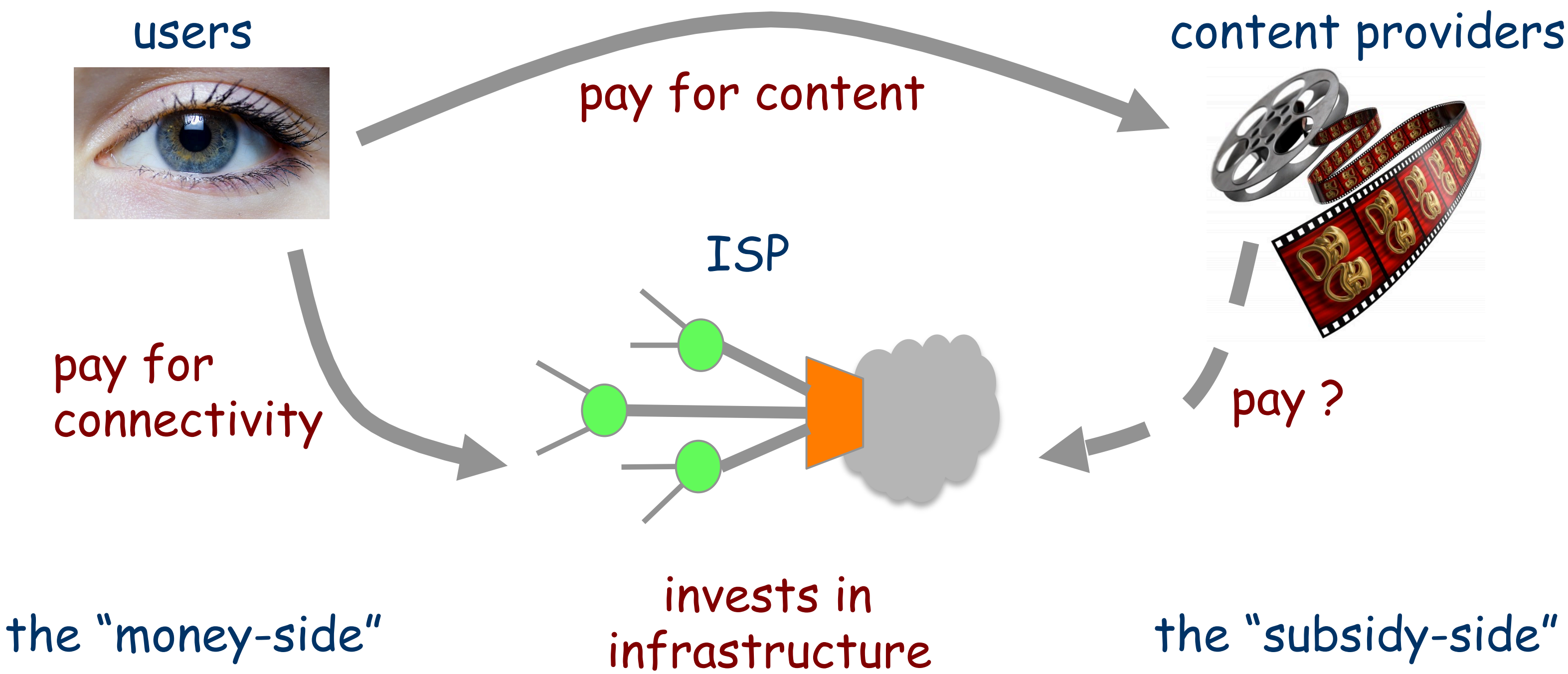
Remarks on trade-off

- key factor is $\Gamma = Tk_b / Nk_m$ where N is catalogue size
  - Γ = max bandwidth cost / max storage cost
- eg, trade-off is favourable at CO – ie, cache all
  - (except for lowest popularity items excluded in Zipf approx)
- eg, trade-off at AN is optimal if N = 1 PB at cache size ~30 TB
  - 40% hit rate, ~30% cost reduction over no cache
- realizing the optimal trade-off relies on CP cooperation
  - pushing the right amount of most popular contents to cache



Realizing the optimal trade-off

- in a 2-sided market, CPs have no cost incentive place content to optimize ISP infrastructure

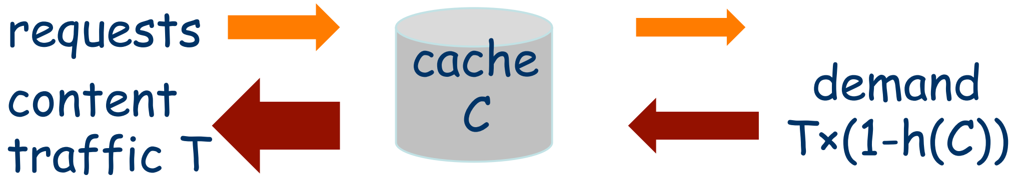


Optimal placement: CPs have the data but are hardly motivated in a 2-sided market

- CPs (eg, Akamai, Facebook, YouTube, Netflix) have highly profitable business models based on exclusive knowledge of customer usage
  - ad placement, recommendations, billing, marketing data, ...
- transparent caching by ISP is not an option
  - CPs need to track demand and control delivery
  - CPs know content popularity and don't want anyone else to know
- CPs can decide content placement but, as the subsidy side of a 2-sided market, have no incentive to optimize ISP investments
  - they currently do not pay ISPs for the cost of their traffic
  - they do install their own caches in the ISP (eg, Google Global Cache) but their economic motivation is different

Price subsidies for an optimal trade-off

- ISP advertises cost functions, $K_b(T)$ and $K_m(C)$
- charges CP $P_{cp}(T)$ for traffic T without cache ($C = 0$)
  - where $0 \le P_{cp}(T) \le K_b(T)$, depending on negotiation
- cost with cache C, $\Delta(C) = K_m(C) + K_b(T(1-h(C)))$ yielding gain $G_{cp}(C,T)$
  - $G_{cp}(C,T) = K_b(T) - K_m(C) - K_b(T(1-h(C)))$
- a subsidy $\alpha\, G_{cp}(C,T)$ for some α ($0 < \alpha < 1$) incites CP to optimize trade-off, yielding ISP gain $(1 - \alpha)\, G_{cp}$

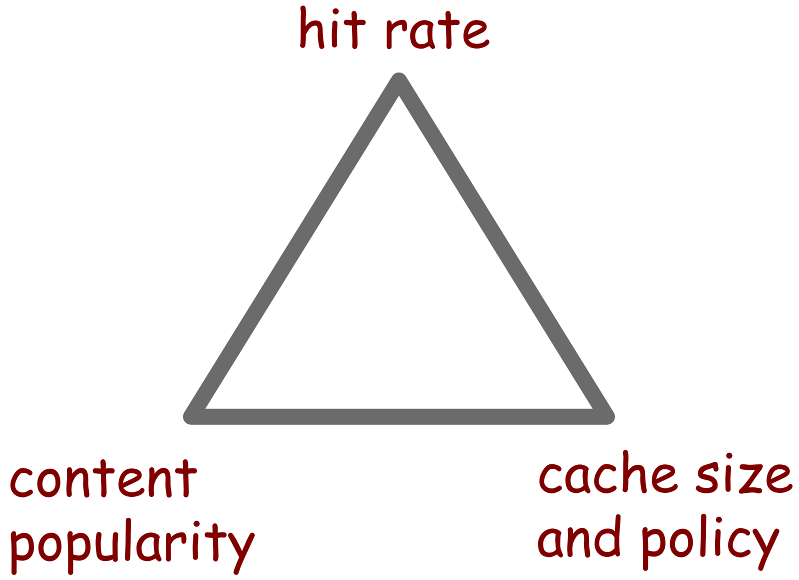


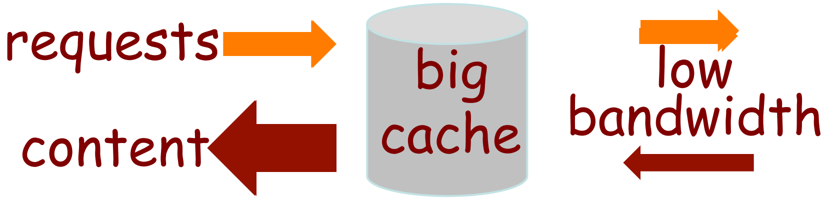
A workable solution?

- CPs currently pay varying amounts to ISPs, sometimes zero and never the full cost of their traffic
  - ISPs can play on performance to “extort” payment (cf. Comcast versus Netflix in 2014) but not to optimize content placement
- the memory for bandwidth subsidy proposal is mainly orthogonal to this 2-sided market negotiation
  - more favourable to high demand, small catalogue CPs (eg, Netflix)
  - but network neutral, transparent pricing
- ISP may not like paying CPs but subsidies are a win-win solution
  - both gain, it remains to decide the best sharing ratio ($\alpha : 1- \alpha$)
- more complex pricing is needed to optimize content placement downstream of the access node (eg, in 5G base stations)
  - work in progress ...

Summary

- understanding the relation between demand, capacity and performance, for a cost-effective infrastructure
- to evaluate the memory for bandwidth trade-off and optimize the cost of infrastructure





Summary

- a complex business environment
  - where content providers (Akamai, Google, Netflix,...) have acquired expertise and need to conserve their advantageous business models
  - as the subsidy side of a 2-sided market
- to realize the optimal trade-off, ISP must further subsidize CPs for their content placement decisions
  - pricing such that subsidy is maximal for the optimal trade-off