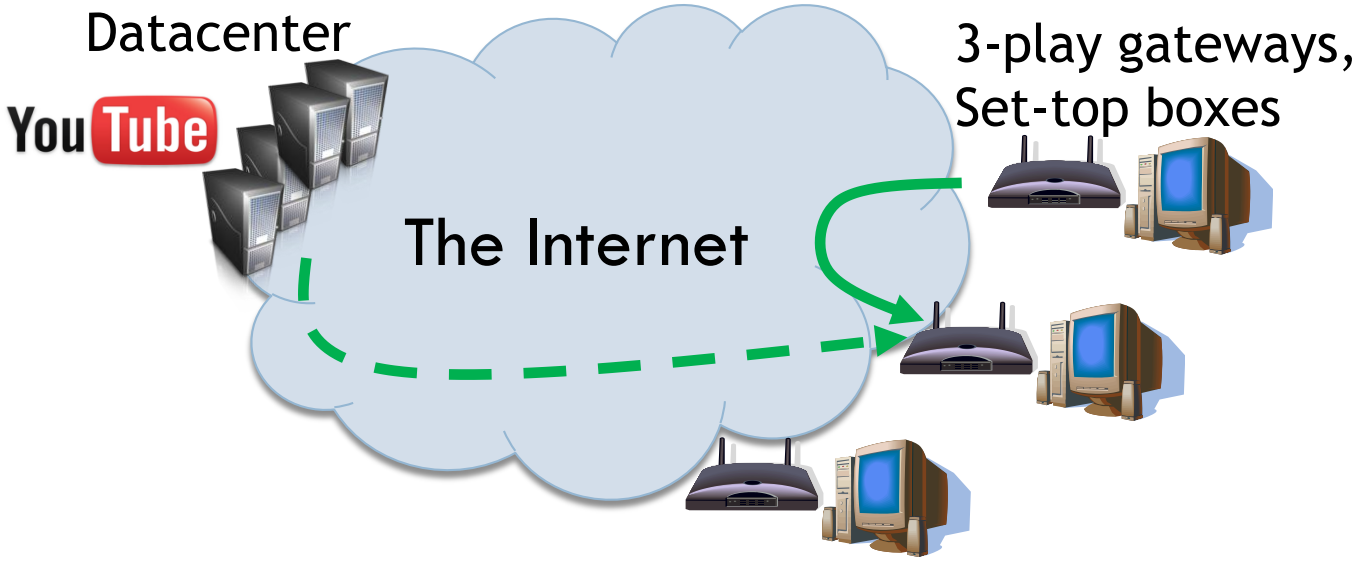
LOAD BALANCING OF HETEROGENEOUS LOADS

Mathieu Leconte (Inria-Technicolor), Marc Lelarge (Inria), Laurent Massoulié (MSR-Inria Joint Centre)

Current architecture: Internet content delivered from "cloud"

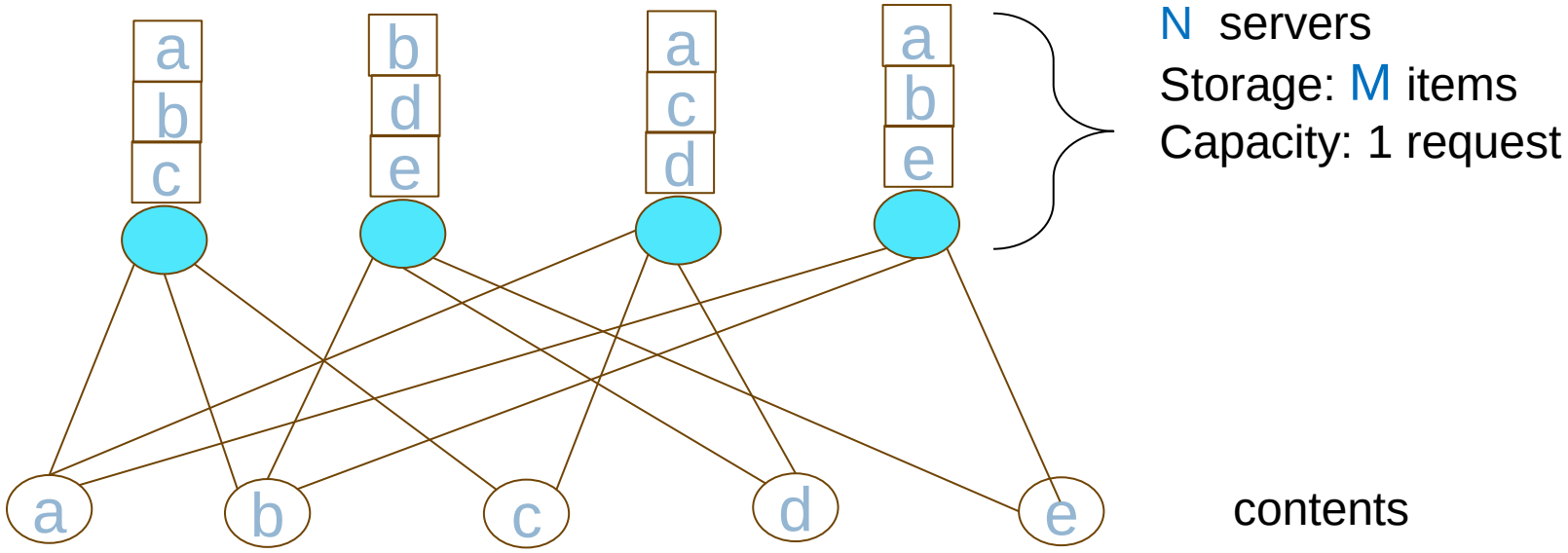
→Why not leverage bandwidth & memory resources at the network's edge?



Questions

- What to store where, given heterogeneous content popularity
- how to match requests to servers
- resulting load reduction on datacenter

Content placement: bipartite graph representation



Bipartite graph: each content **c** linked to servers storing **c**

→Aim: find graph structures maximizing system's capacity, i.e. Such that content replication bottleneck disappears for smallest M

Outline

- Proportional placement at subcritical load
- Greedy matching – phase transition at critical load
- Optimal matching – characterization via “cavity method”

Statistical Assumptions

Fixed number K of item classes

α_k N items of class k,

each with demand rate (popularity) ν_k

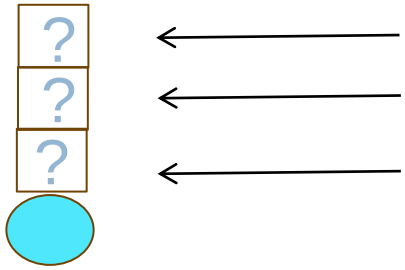
i.e. at any time, Poisson nb of requests per item.

→Replication can rely on expected numbers of requests, not on actual numbers

Request sizes i.i.d. with mean 1

→System load $\rho = \sum_{k=1}^{K} \alpha_k \nu_k$

A family of randomized placement schemes



I.I.D. selection of contents in each memory slot according to probability distribution $\{p_c\}$

Special case: the "proportional" placement strategy

$$p_c \propto \nu_{k(c)}$$

Proportional placement

Each server independently stores M-set m with probability

$$\pi(m) \propto \prod_{c \in m} \nu_{k(c)}$$

Motivation

steady-state distribution of cache-management strategy where

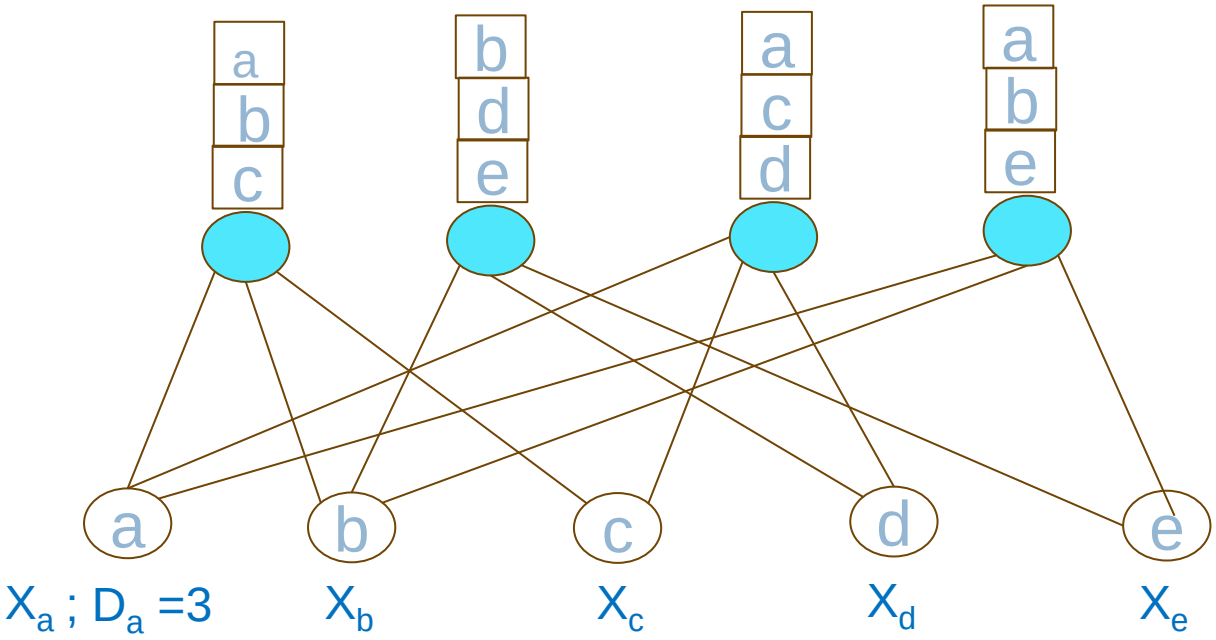
- object c brought into cache at rate $\nu_{k(c)}$
- uniform selection of object to be evicted

→Adapts implicitly to content popularities – no need for inference

More general placement schemes

Each content c: nb of requests X_c & replicas D_c drawn from joint distribution p(x,d)

→ Uniform random bipartite graph under such degree assumptions



→Previous model: with prob. α_k / α content of type k , conditionally on which

X_c : Poisson (ν_k), D_c : Poisson ($M \nu_k / \rho$)

Sub-critical systems: logarithmic memory is enough

Assume system load $\rho<1$

For $M> h(\rho)\log(N)$, with probability proportional placement suffices to absorb all load

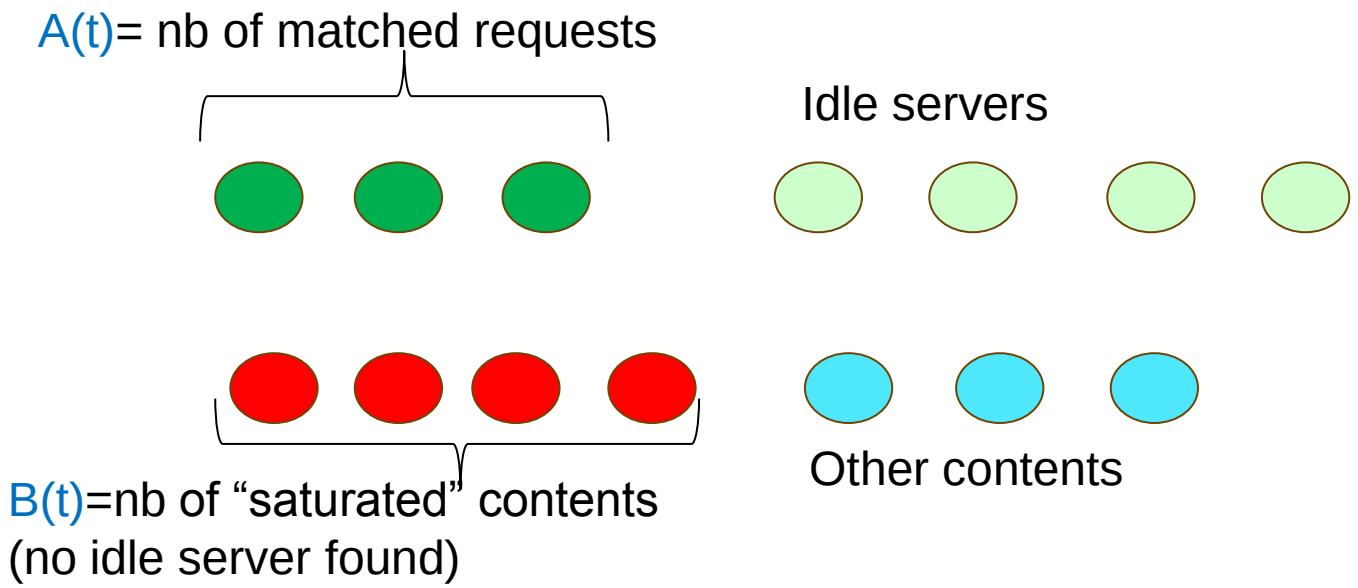
Remarks

- Logarithmic storage also necessary to absorb all requests without prior knowledge of their exact numbers
- Constant $h(\rho)$ scales like $(1-\rho)^{-2}$ as $\rho \rightarrow 1$

→ We don't know what happens near and above criticality

Greedy Matching (homogeneous case $\nu_k \equiv \nu$)

- Consider requests in arbitrary order. Associate current request with random idle server holding requested content if any



- Markov process (A(t),B(t)) approximated by ODE for large N

→ Allows to characterize ***inefficiency***: min(1, ρ) minus fraction of matched servers)

Greedy Matching (homogeneous case $\nu_k \equiv \nu$)

Resulting inefficiency:

$$\rho \neq 1 \Rightarrow \varphi = e^{-M|\rho-1|(+o(1))\alpha}$$

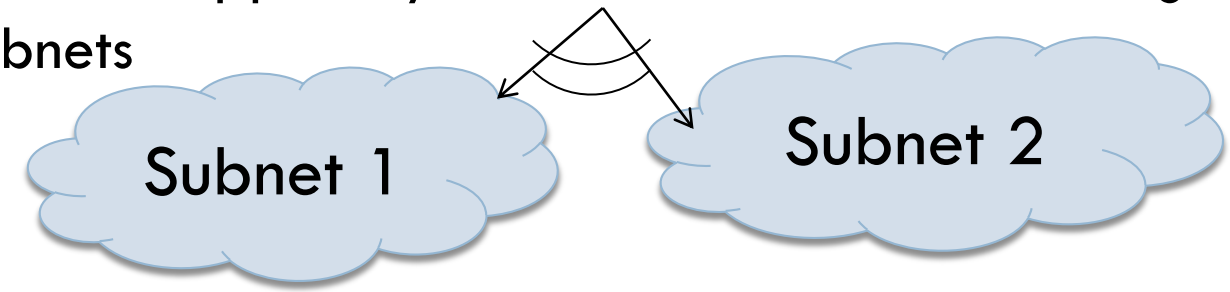
$$\rho = 1 \Rightarrow \varphi = \frac{\alpha \log(\quad)}{\mathrm{M}}(+o(\quad)$$

→Exponentially small away from critical load,

→Comparably large at criticality

Critical load: may be the typical case:

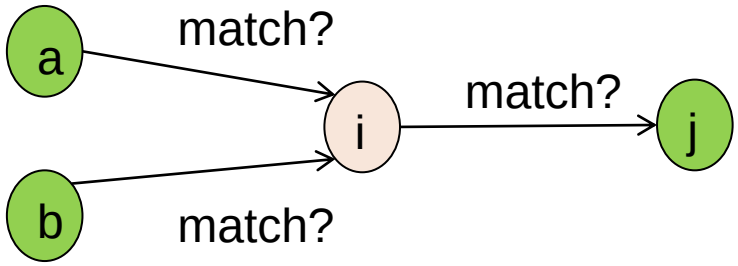
load distribution at upper layers can induce critical loading of particular subnets



Question: is optimal matching qualitatively better?

Message passing and maximum matching size - finite graphs

Message passing dynamics for matching:



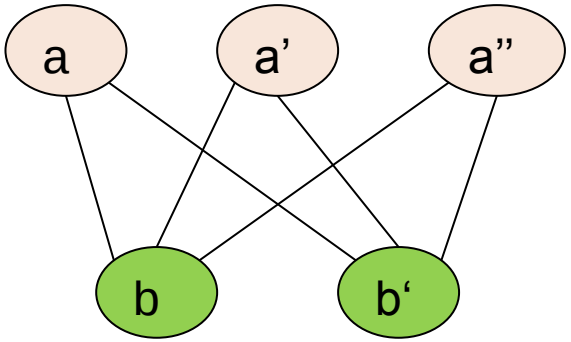
$m_{ij}=1$ if i proposes incorporation of edge (ij) in matching to j

X_i : matching capacity of node i →

$$m_{ij} = 1_{X_i > \sum_{k \neq j} m_{ki}}$$

Iterative updates of messages: m(t+1)=F(m(t))

Message passing and maximum matching size - finite graphs



Then for finite bipartite graph G(A+B,E), maximum matching size:

$$M(G) = \min_{m=F(m)} \left\{ \sum_{a \in A} X_a \wedge m_{\bullet a} + \sum_{b \in B} X_b 1_{m_{\bullet b} > X_b} \right\}$$

where

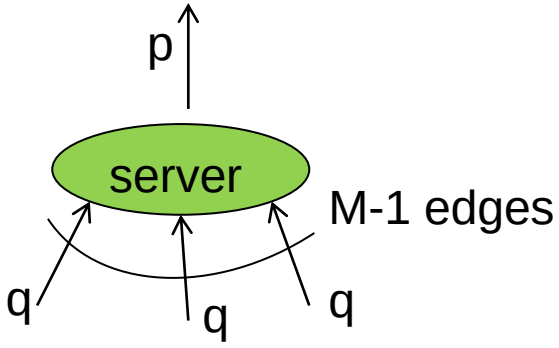
$$m_{\bullet a} = \sum_{b \sim a} m_{ba}$$

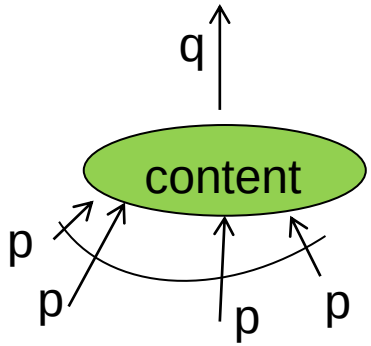
Large N-Asymptotics of matched requests under optimal matching

p (q): prob. msg from server (content) to content (server) equals 1

Condition on msg invariance yields "Recursive Distributional Equations" on p, q:

$$p = (1-q)^{M-1}, \quad q = \frac{1}{E(D)} E\left(D 1_{\mathrm{Bin}(D-1,p) \geq X}\right)$$





Size-biased degree distribution

$$\pi(d) = \frac{(d+1)P(D=d+1)}{E(D)}$$

and characterization of matching density:

$$\frac{M(G_N)}{N} \to \inf_p \left\{ 1 - (1-q)^M + \alpha \mathrm{E}\left(X 1_{\mathrm{Bin}(D,p) \geq X+1}\right) \right\}$$

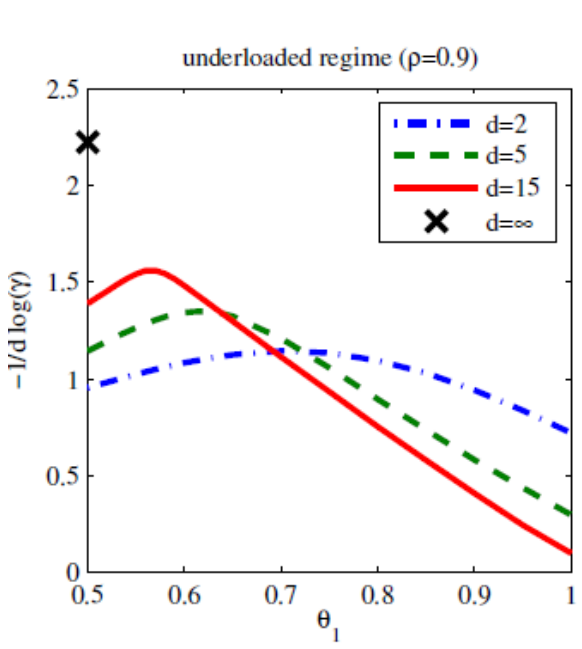
Corollaries

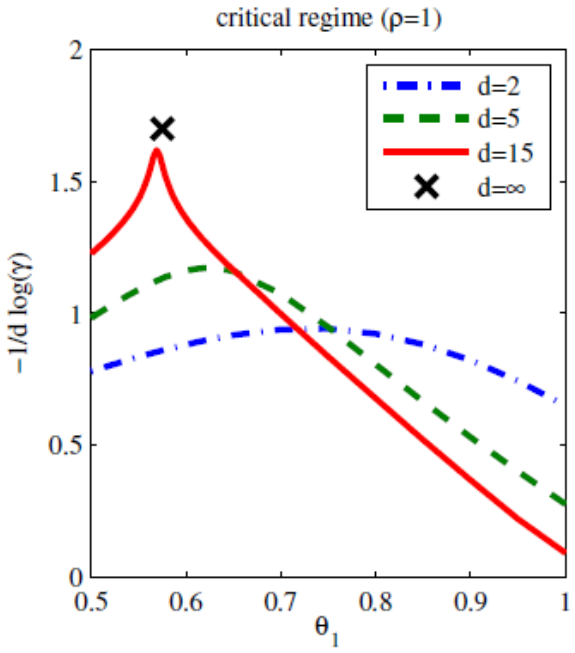
(i) For many placement strategies (including proportional), inefficiency exponentially small in M even at criticality $\rho=1$

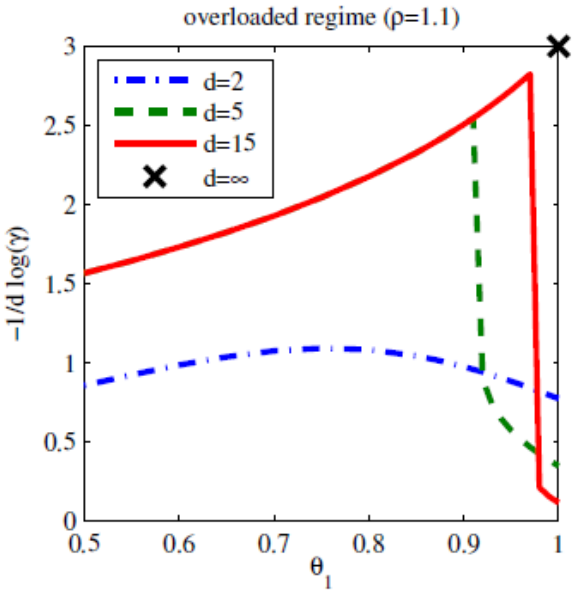
(ii) For underloaded system $\rho<1$, optimal allocation closer to uniform than to proportional

(iii) For overloaded system $\rho>1$, optimal allocation amplifies replication bias in favour of popular content more than proportional

Illustration on two-class scenario







Conclusions

Proportional placement achieves good performance for sub-critical loss network (logarithmic storage enough)

"Cavity method" used to show

-need for refined matching at criticality (greedy not good enough)

-potential to improve upon proportional placement

Open questions

- → Take into account user geographic distribution, network costs
- → Adaptive schemes inducing better than proportional content placement strategies
- → Consider more realistic, N-dependent popularity (Zipf distribution)

THANKS!

Backup: ODE for greedy matching analysis

$$a(t) = \frac{A(\alpha N t)}{\alpha N}, \quad b(t) = \frac{B(\alpha N t)}{\alpha N}$$

$$\frac{da}{dt} = (-b)\left[-e^{-M(\alpha^{-1}-a)/(-b)}\right]$$

$$\frac{db}{dt} = (-b)e^{-M(\alpha^{-1}-a)/(-b)}$$

Digression: a simple special case

- For M=1, matching becomes trivial
- Can handle Zipf popularity distribution

→ For critical loading:

- inefficiency:
  for a<1 — Omega(1)
  for a=1 — O(log(N)/log(log(N)))
  for 1<a<2 — O(N^{1/a-1})
  for a=2 — O(N^{-1/2}log(N))
  for a>2 — O(N^{-1/2} omega(N))
- →class-based scenario "worst-case", related to a<1 scenario