Cost-aware caching: optimizing cache provisioning and object placement in ICN

Andrea Araldo
M. Mangili
F. Martignon
D. Rossi

IEEE Globecom, 8-12 December 2014, Austin, Texas

Cost-aware caching: optimizing cache provisioning and object placement in $\boxed{\text{ICN}}$

Andrea Araldo
M. Mangili
F. Martignon
D. Rossi

IEEE Globecom, 8-12 December 2014, Austin, Texas

Cost-aware caching: optimizing cache provisioning and object placement in ICN

Andrea Araldo
M. Mangili
F. Martignon
D. Rossi

IEEE Globecom, 8-12 December 2014, Austin, Texas

Outline

- Context
- Scenario
- Our contribution
  - Design of a cache system with the aim to reduce inter-domain traffic cost of ISPs
  - Optimization models: CLASSIC vs. COST-AWARE
    - Forwarding, object placement, cache sizing
  - Greedy algorithms
- Results & findings
  - Classic caching is cost-ineffective
  - Cost-aware caching can bring sizable cost savings
- Conclusion

Context

- Work on **cache design** focuses on classic metrics:
  - hit ratio, hop distance, traffic load

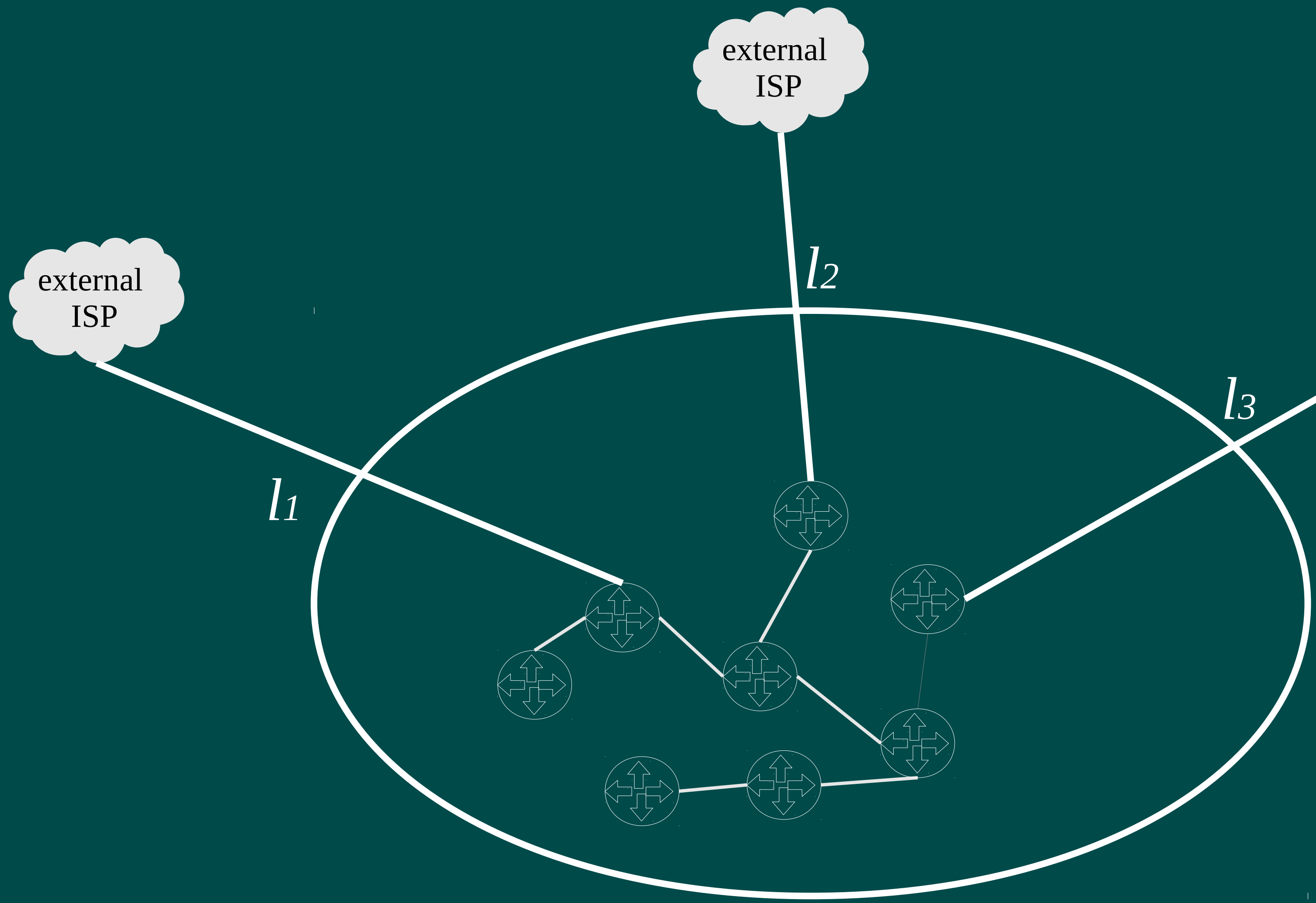
- Some works on **economic implications** of caching
  - Only interaction between ISPs is considered
  - No consideration on cache design inside ISPs network

Orthogonal aspects

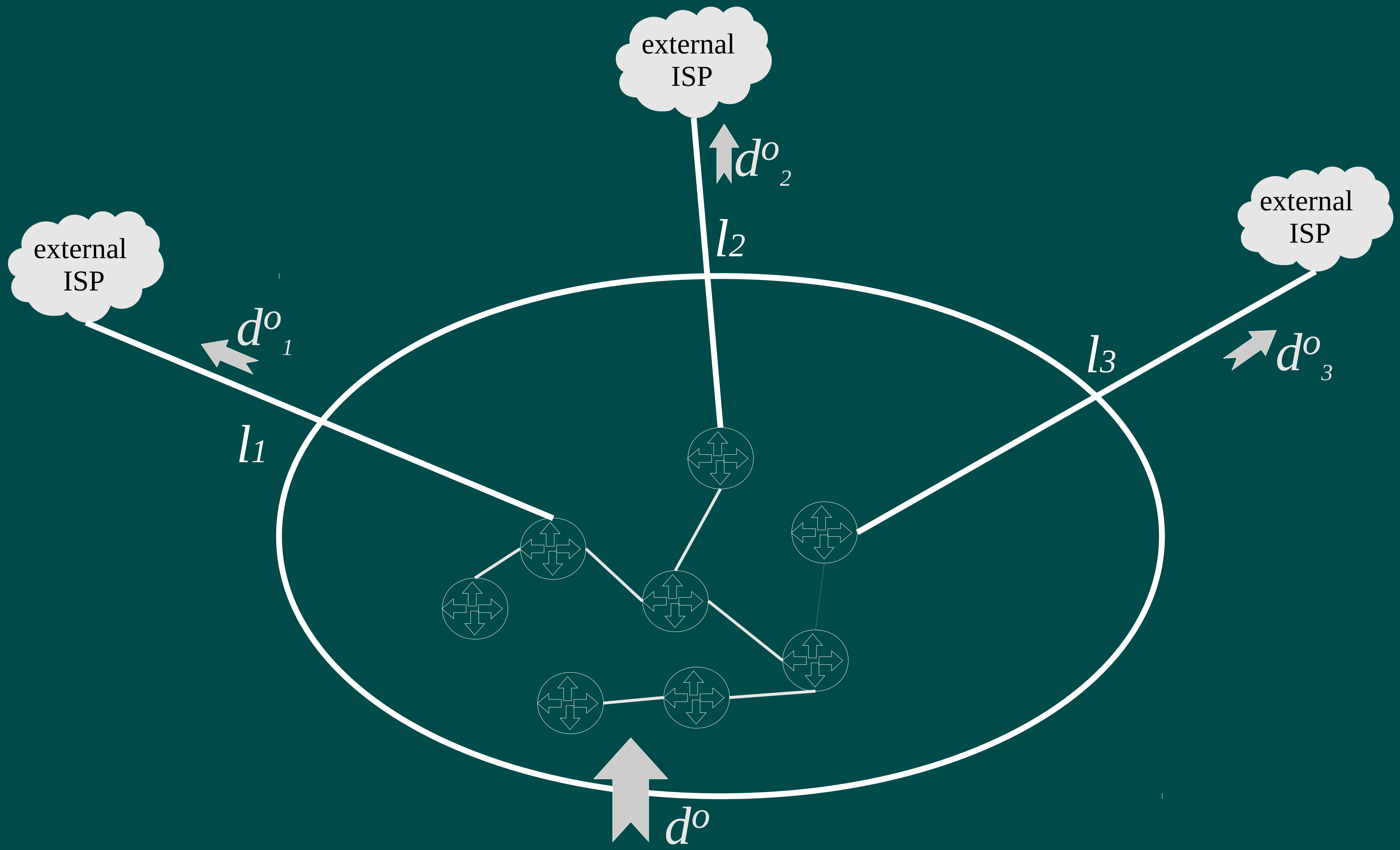
We study how an ISP can reduce costs by properly designing its cache system

Andrea Araldo

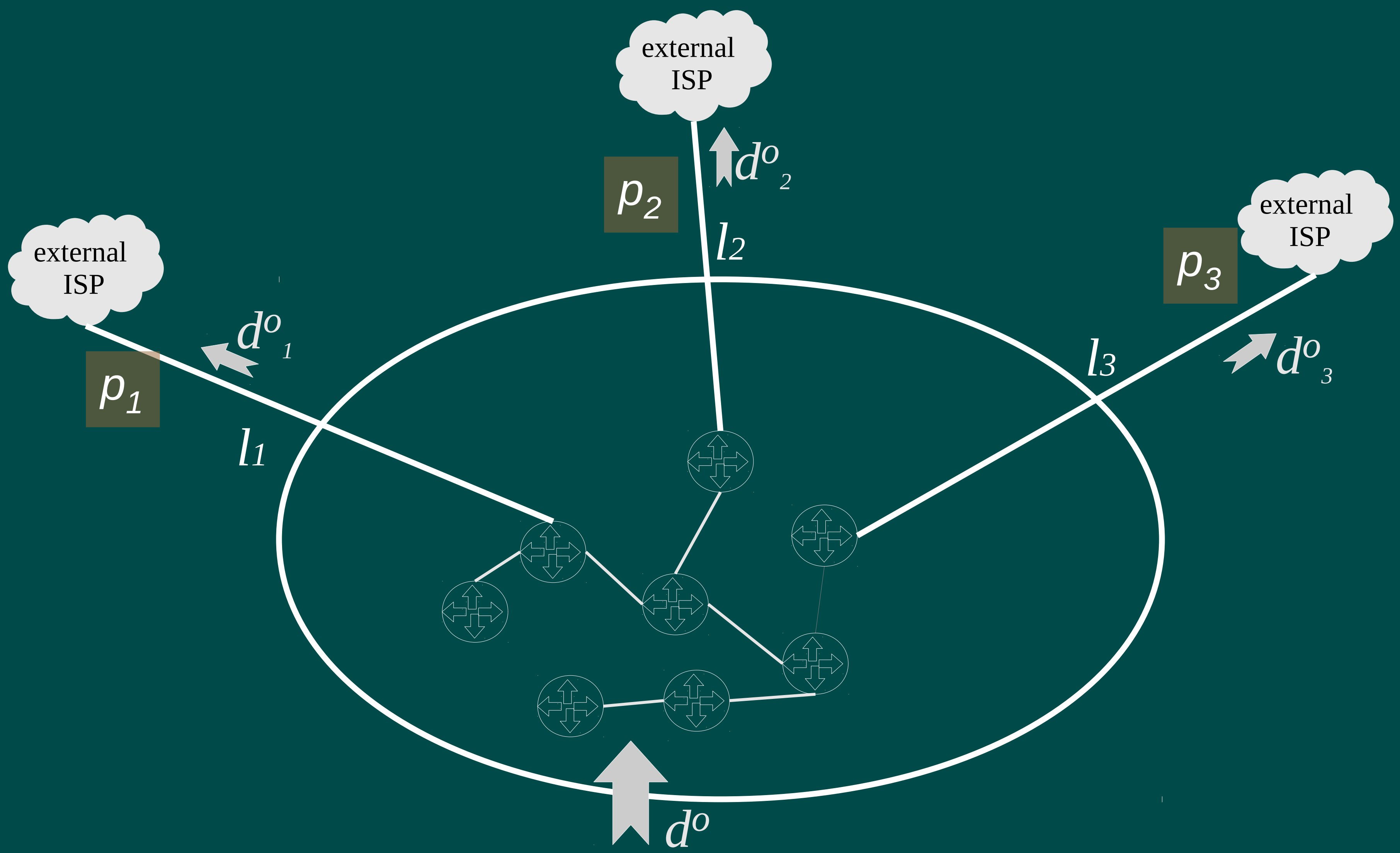
Scenario



Scenario

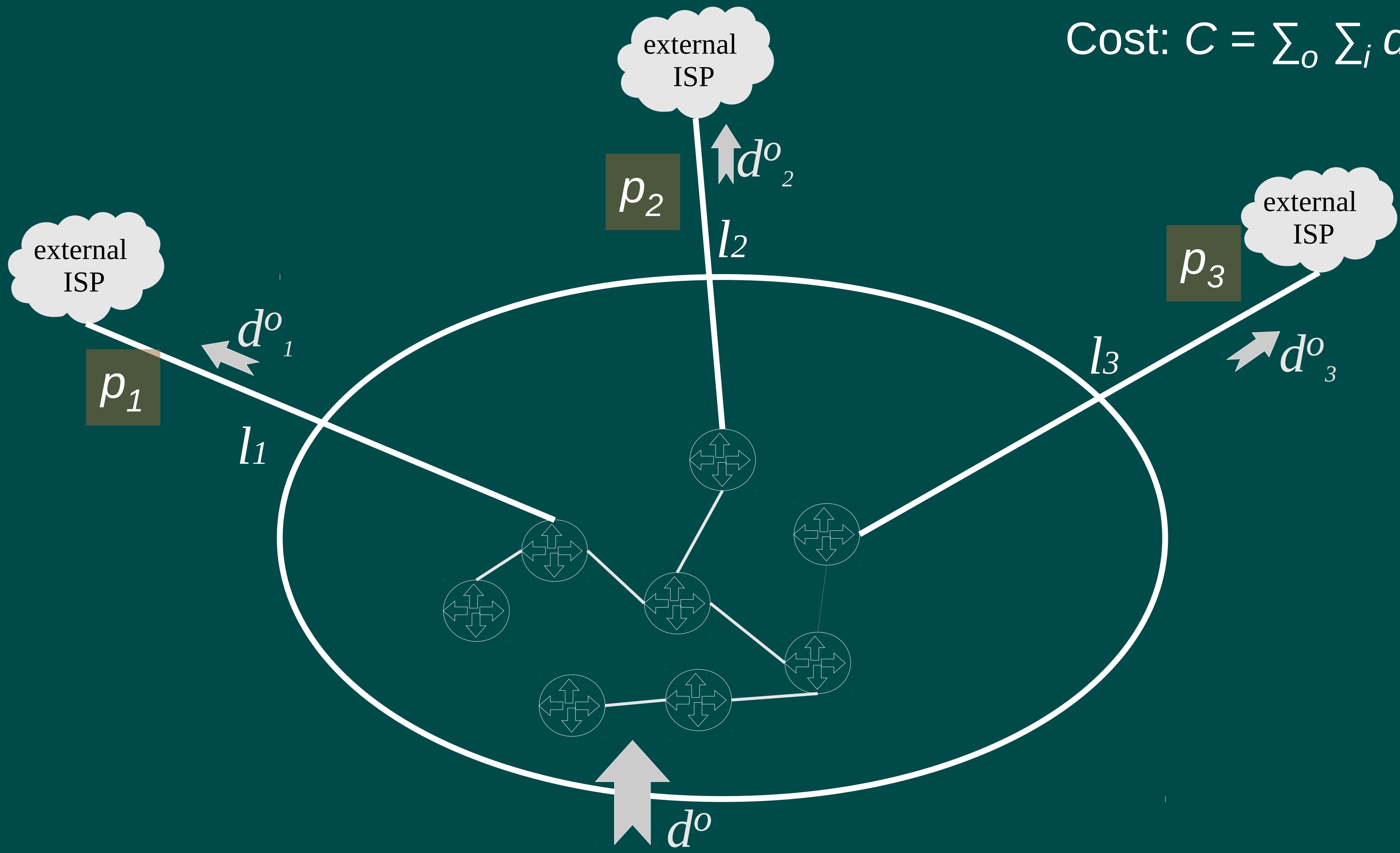

external ISP
external ISP
external ISP
d^o_2
l_2
d^o_1
l_1
l_3
d^o_3
d^o

Scenario


external ISP
external ISP
external ISP
$d^o{}_2$
p_2
l_2
p_3
$d^o{}_1$
l_3
$d^o{}_3$
p_1
l_1
d^o

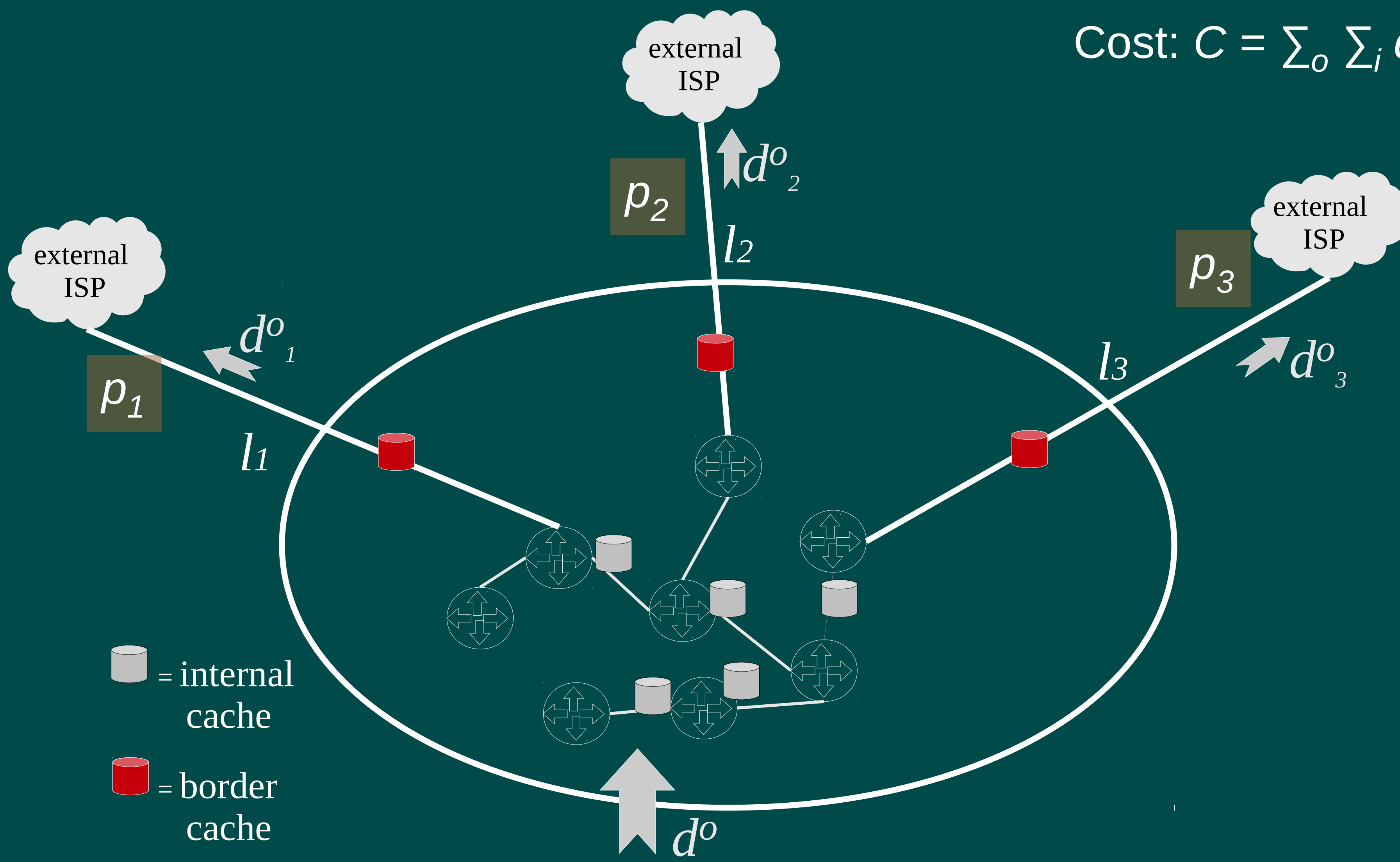
Scenario

Cost: $C = \sum_o \sum_i d^o{}_i \, p_i$



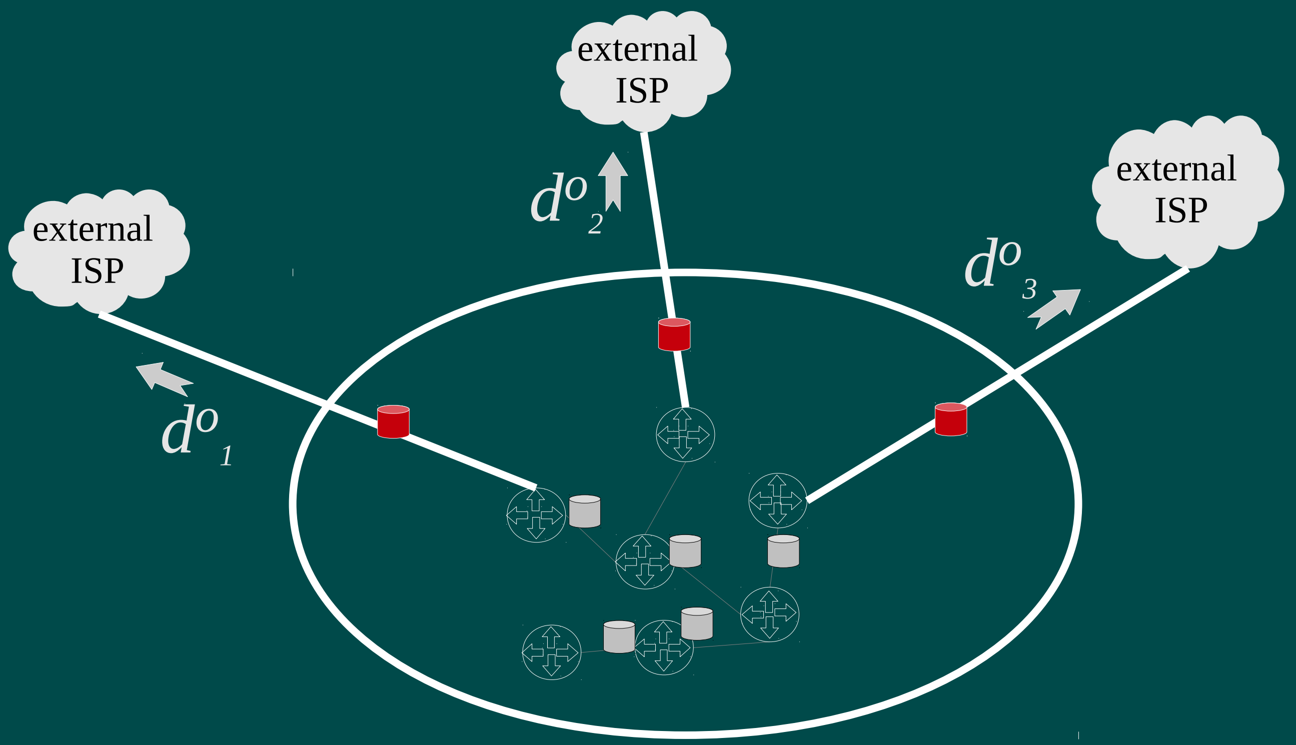
Scenario

Cost: $C = \sum_o \sum_i d^o{}_i \, p_i$



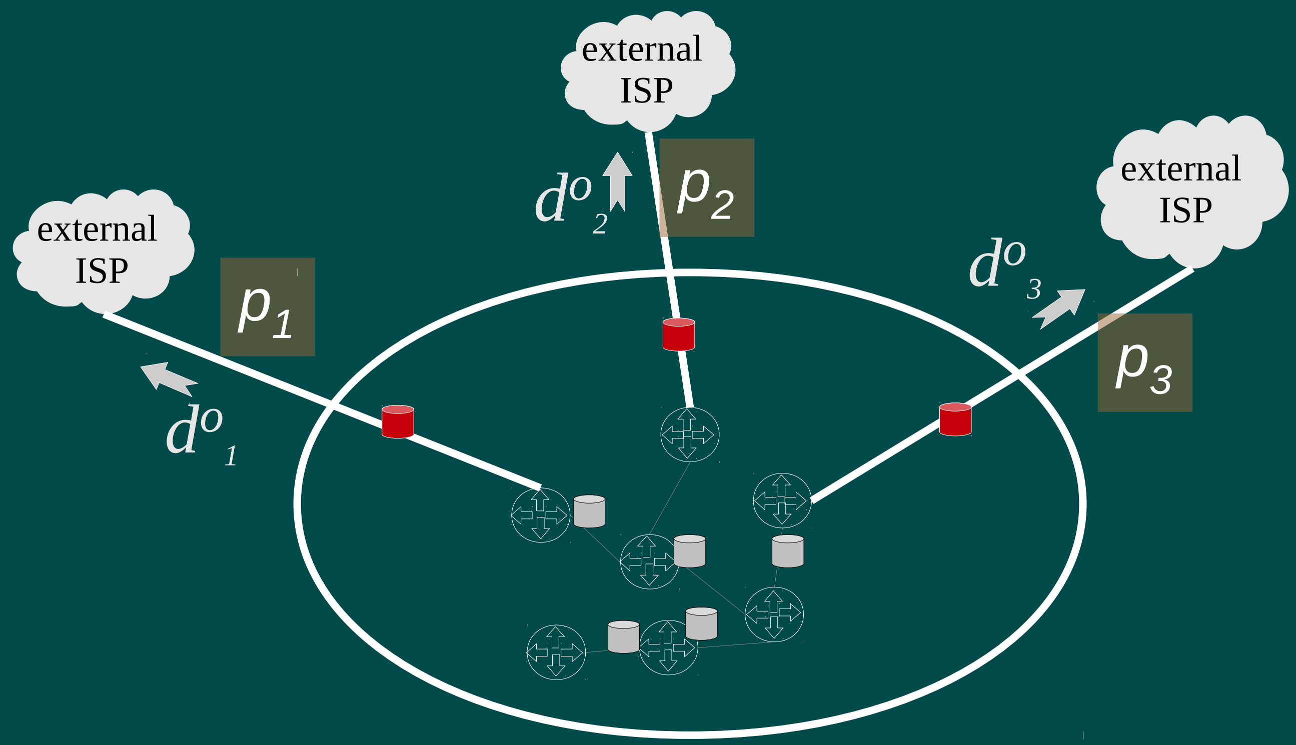
Optimization goals

At first glance: hit-ratio ↗ ➡ inter-domain traffic ↘ ➡ cost ↘



Optimization goals

At first glance: hit-ratio ↗ ➡ inter-domain traffic ↘ ➡ cost ↘



Price heterogeneity

Andrea Araldo

Optimization goals

At first glance: hit-ratio ↗ ⇒ inter-domain traffic ↘ ⇒ cost ↘

- *CLASSIC opt model*

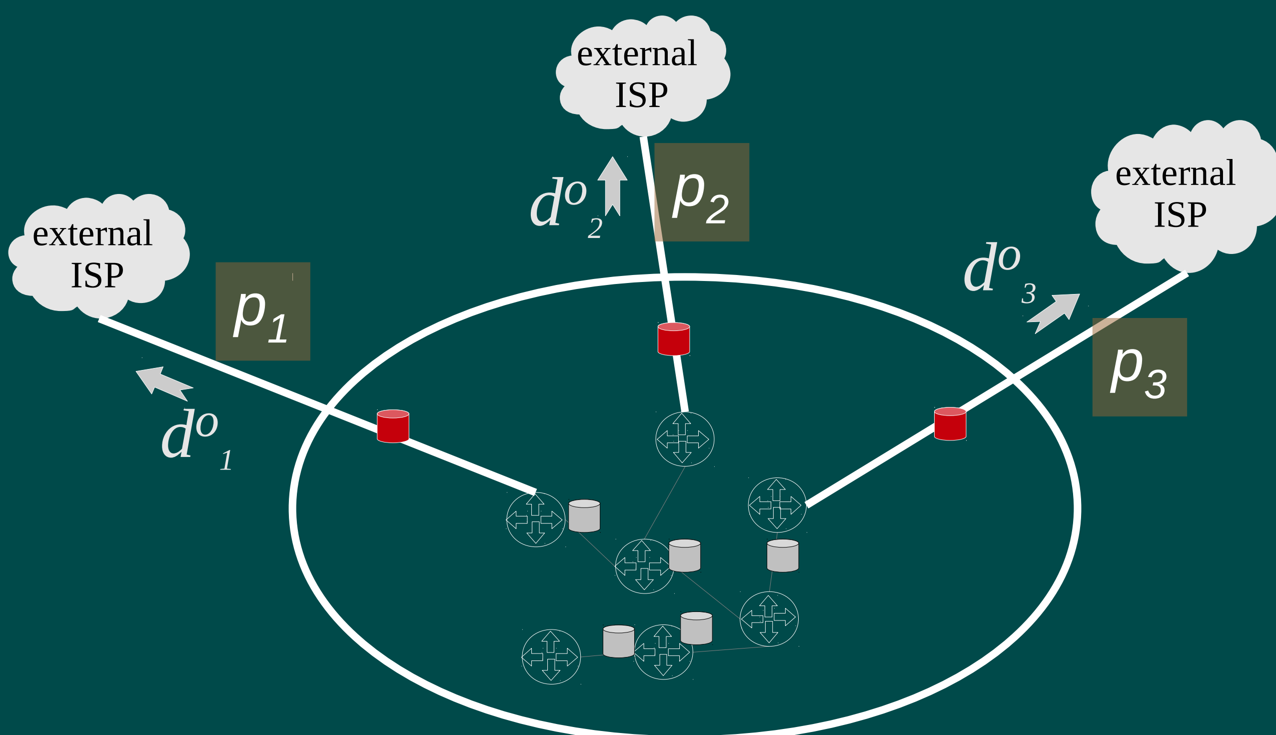
  Hit-ratio: $H = \dfrac{d^o - \sum_o \sum_i d^o_i}{d^o}$

  max H

- *COST-AWARE opt model*

  Cost: $C = \sum_o \sum_i d^o_i \, p_i$

  min C



Price heterogeneity

Constraints

$$\rho_o = F\left(\mathcal{Q}, P, \{x_{n,o} | n \in \mathcal{N}\}\right) \tag{1}$$

$$d^{out}_{core,o} = (1 - \rho_o) d_o \tag{2}$$

$$d^{out}_{core,o} = \sum_{l \in \mathcal{L}} d^{in}_{l,o} \tag{3}$$

$$d^{in}_{l,o} = 0, \forall l \in \mathcal{L} \setminus \mathcal{L}_o \tag{4}$$

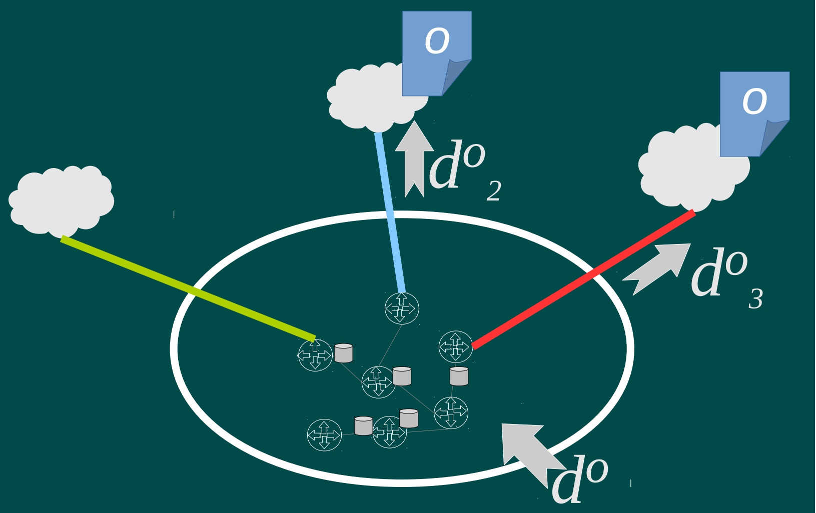
$$d^{out}_{l,o} = (1 - x_{l,o}) \cdot d^{in}_{l,o} \tag{5}$$

$$\sum_{o \in \mathcal{O}} x_{l,o} = cs_l, \forall l \in \mathcal{L} \tag{6}$$

$$\sum_{o \in \mathcal{O}} x_{n,o} = cs_n, \forall n \in \mathcal{N} \tag{7}$$

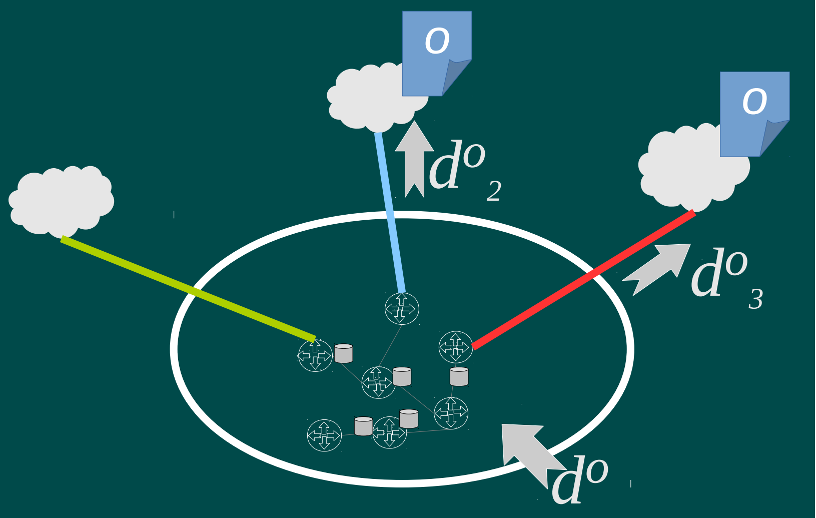
$$\sum_{l \in \mathcal{L}} cs_l + \sum_{n \in \mathcal{N}} cs_{n,o} \leq C_{tot}. \tag{8}$$

Greedy algorithms



Greedy algorithms

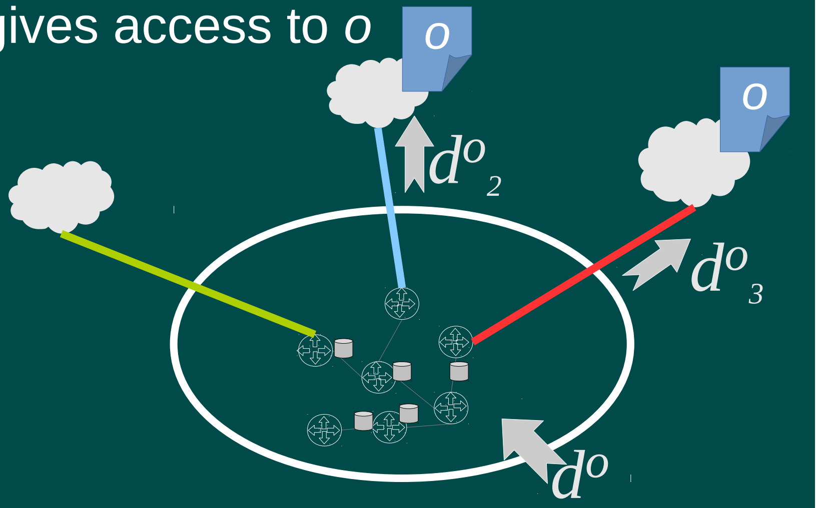
1. Where should we forward each request for object *o*?



Greedy algorithms

1. Where should we forward each request for object *o*?

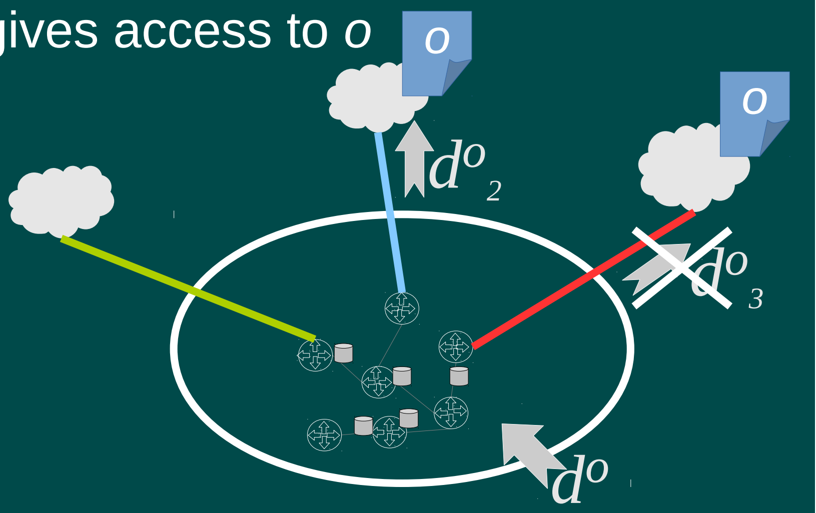
– Forward them only to the cheapest link l_o that gives access to *o*



Greedy algorithms

1. Where should we forward each request for object *o*?

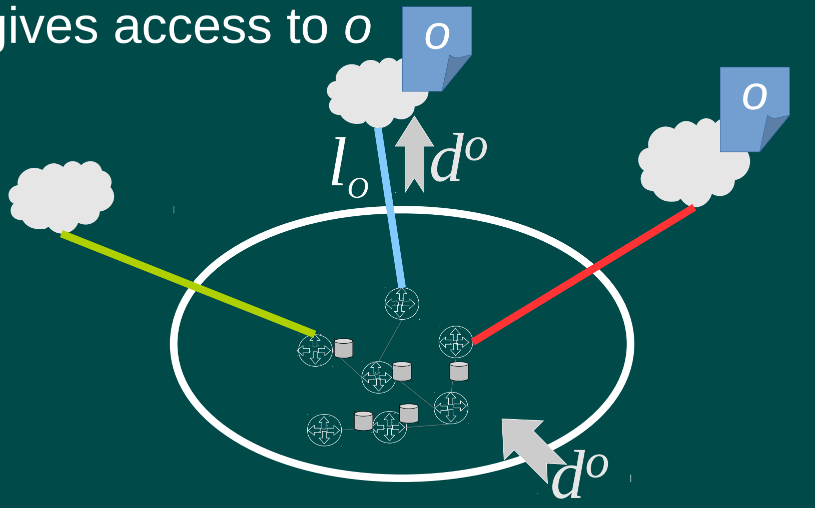
– Forward them only to the cheapest link l_o that gives access to *o*



Greedy algorithms

1. Where should we forward each request for object *o*?

– Forward them only to the cheapest link l_o that gives access to *o*

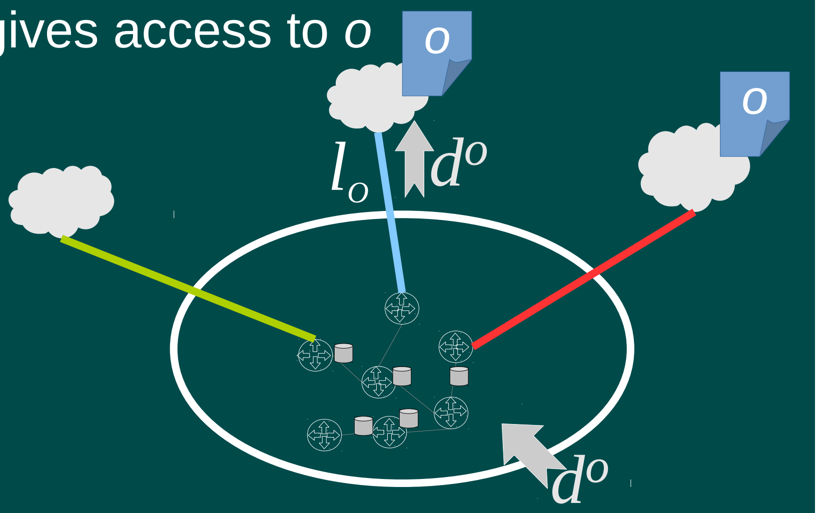


Greedy algorithms

1. Where should we forward each request for object *o*?

- Forward them only to the cheapest link l_o that gives access to *o*
- Compute the cost of each object

  $c_o \triangleq d^o \cdot p_{l_o}$

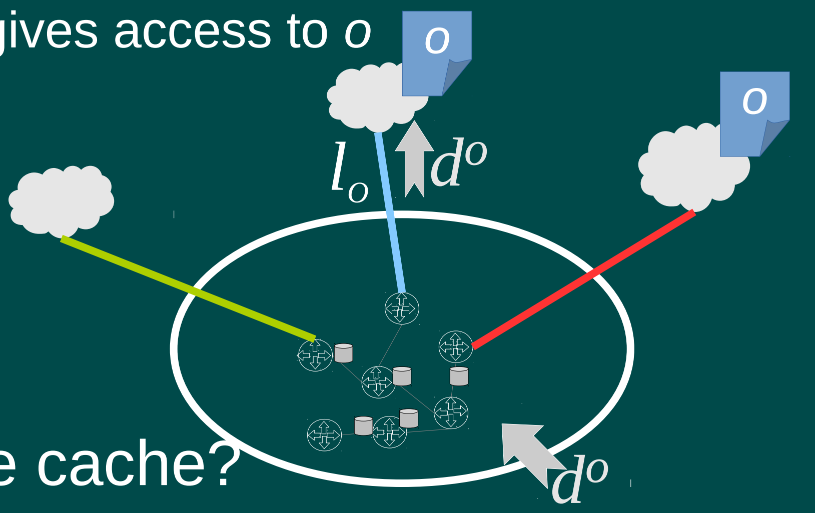


Greedy algorithms

1. Where should we forward each request for object o?

- Forward them only to the cheapest link l_o that gives access to o
- Compute the cost of each object

  $c_o \triangleq d^o \cdot p_{l_o}$



2. Which objects should we cache?

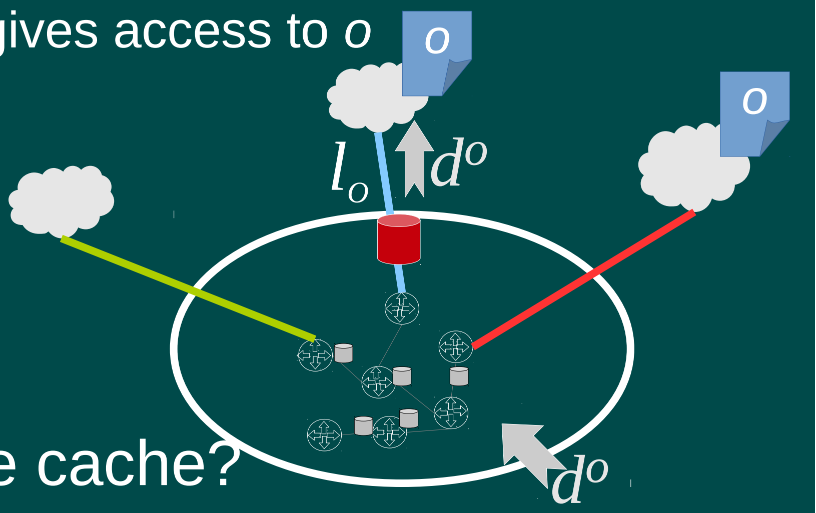
- If your cache budget is C_{tot}, cache the C_{tot} objects with highest cost $c_o = d_o \cdot p_{lo}$

Greedy algorithms

1. Where should we forward each request for object *o*?

- Forward them only to the cheapest link l_o that gives access to *o*
- Compute the cost of each object

  $c_o \triangleq d^o \cdot p_{l_o}$



2. Which objects should we cache?

- If your cache budget is C_{tot}, cache the C_{tot} objects with highest cost $c_o = d_o \cdot p_{lo}$

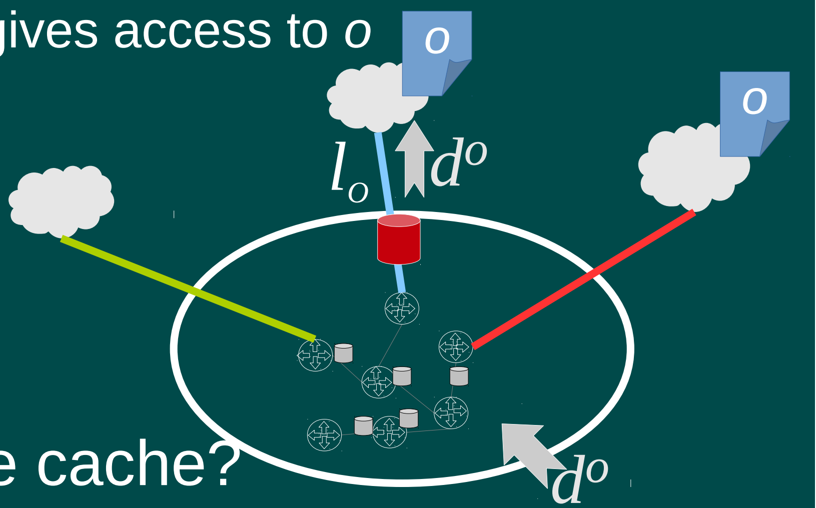
3. Where should we cache o?

- Cache *o* in the border cache of the associated link l_o

Greedy algorithms

1. Where should we forward each request for object *o*?

- Forward them only to the cheapest link l_o that gives access to *o*
- Compute the cost of each object
  $c_o \triangleq d^o \cdot p_{l_o}$



2. Which objects should we cache?

- If your cache budget is C_{tot}, cache the C_{tot} objects with highest cost
  $c_o = d_o \cdot p_{lo}$

3. Where should we cache o?

- Cache *o* in the border cache of the associated link l_o

COST-AWARE solution

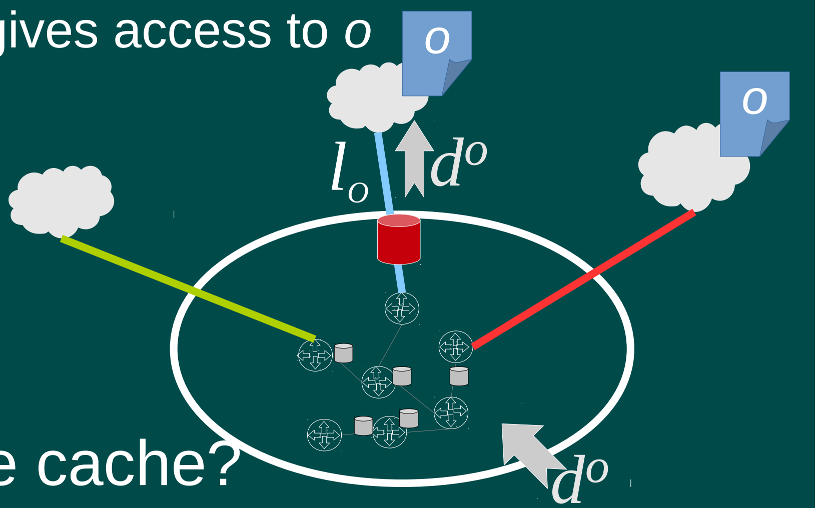
Greedy algorithms

Forwarding:

1. Where should we forward each request for object *o*?

- Forward them only to the cheapest link l_o that gives access to *o*
- Compute the cost of each object

$$c_o \triangleq d^o \cdot p_{l_o}$$



Object placement:

2. Which objects should we cache?

- If your cache budget is C_{tot}, cache the C_{tot} objects with highest cost $c_o = d_o \cdot p_{lo}$

Cache sizing:

3. Where should we cache o?

- Cache *o* in the border cache of the associated link l_o

COST-AWARE solution

Greedy algorithms

Forwarding:

1. Where should we forward each request for object *o*?

- Forward them only to the cheapest link l_o that gives access to *o*
- Compute the cost of each object
  $c_o \triangleq d^o \cdot p_{l_o}$

Object placement:

2. Which objects should we cache?

- If your cache budget is C_{tot}, cache the C_{tot} objects with highest cost $\boxed{c_o = d_o \cdot p_{lo}}$
- Cache the C_{tot} most popular objects, i.e. the ones with highest $\boxed{d^o}$

Cache sizing:

3. Where should we cache o?

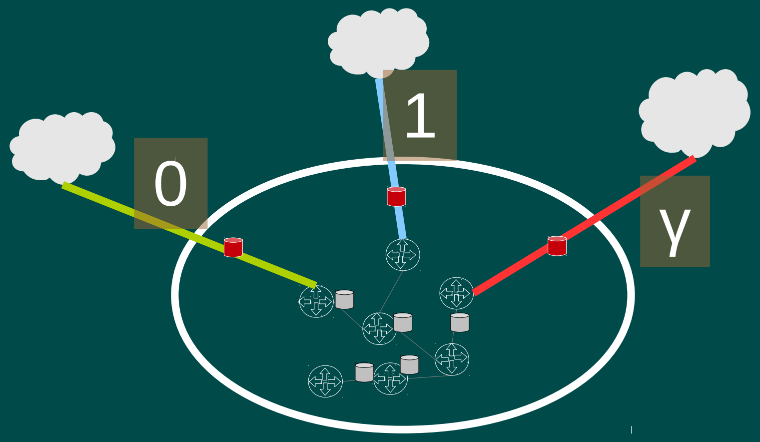
- Cache *o* in the border cache of the associated link l_o

COST-AWARE solution

CLASSIC solution

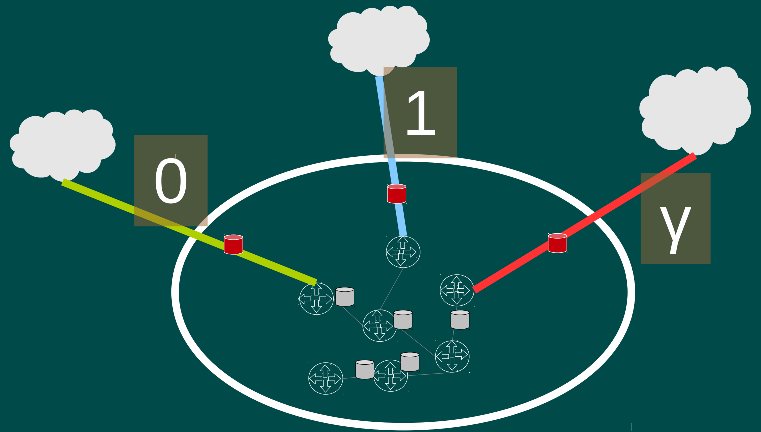
Parameters

- Realistic catalog: 10^7 objects
- Zipf popularity
- 3 external links



- Objects are randomly distributed on external links (40 seeds)
- For each configuration, we compute:
  - $H_{CLASSIC}$, $C_{CLASSIC}$, $H_{COST\text{-}AWARE}$, $C_{COST\text{-}AWARE}$

Cost savings

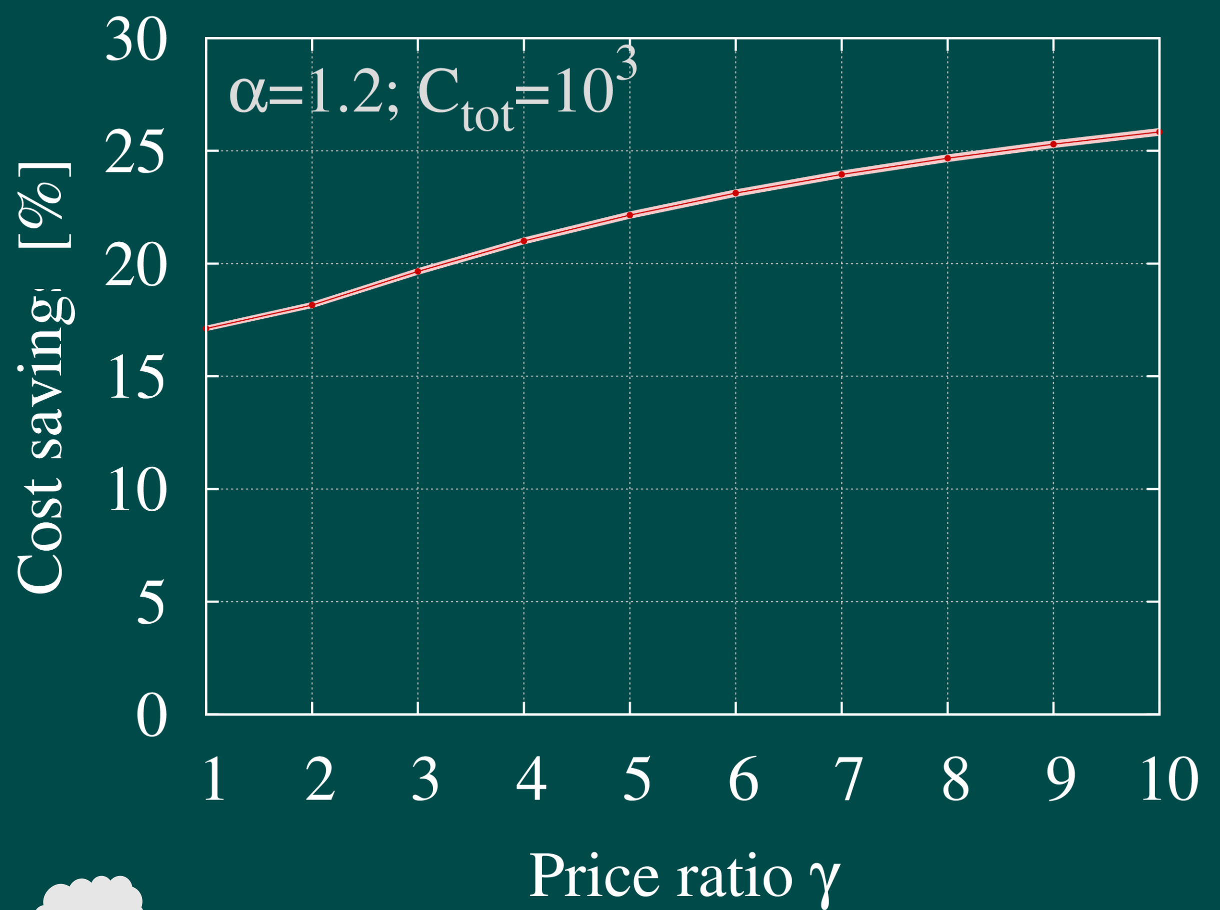


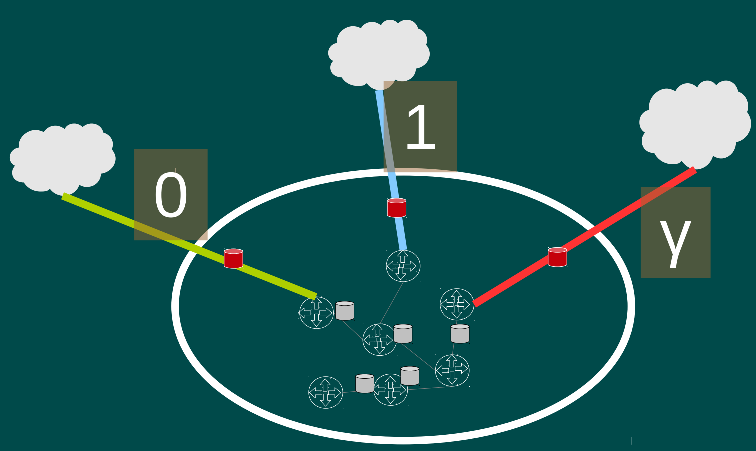
$$\text{Cost saving} = \frac{C_{CLASSIC} - C_{COST\text{-}AWARE}}{C_{CLASSIC}}$$

Cost savings

Price diversity ↗

⇒ Savings ↗



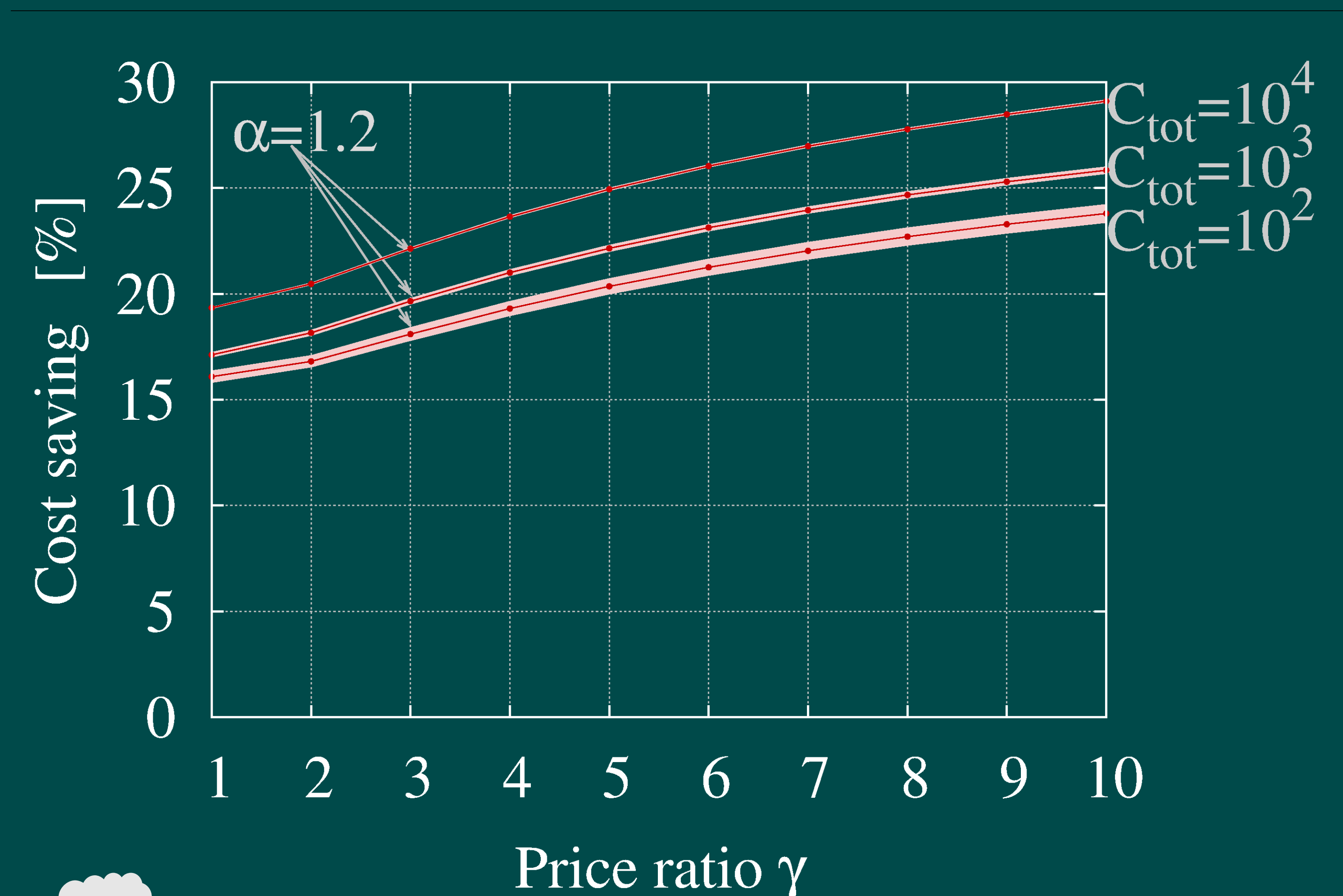


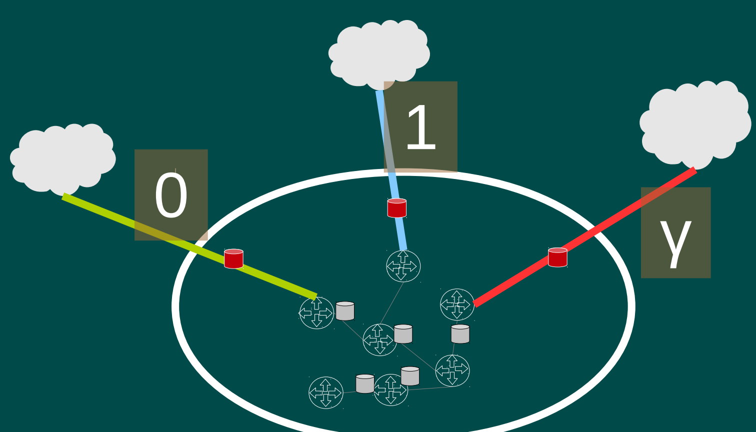
$$\text{Cost saving} = \frac{C_{CLASSIC} - C_{COST\text{-}AWARE}}{C_{CLASSIC}}$$

Cost savings

Price diversity ↗
➡ Savings ↗

Cache Budget ↗
➡ Savings ↗





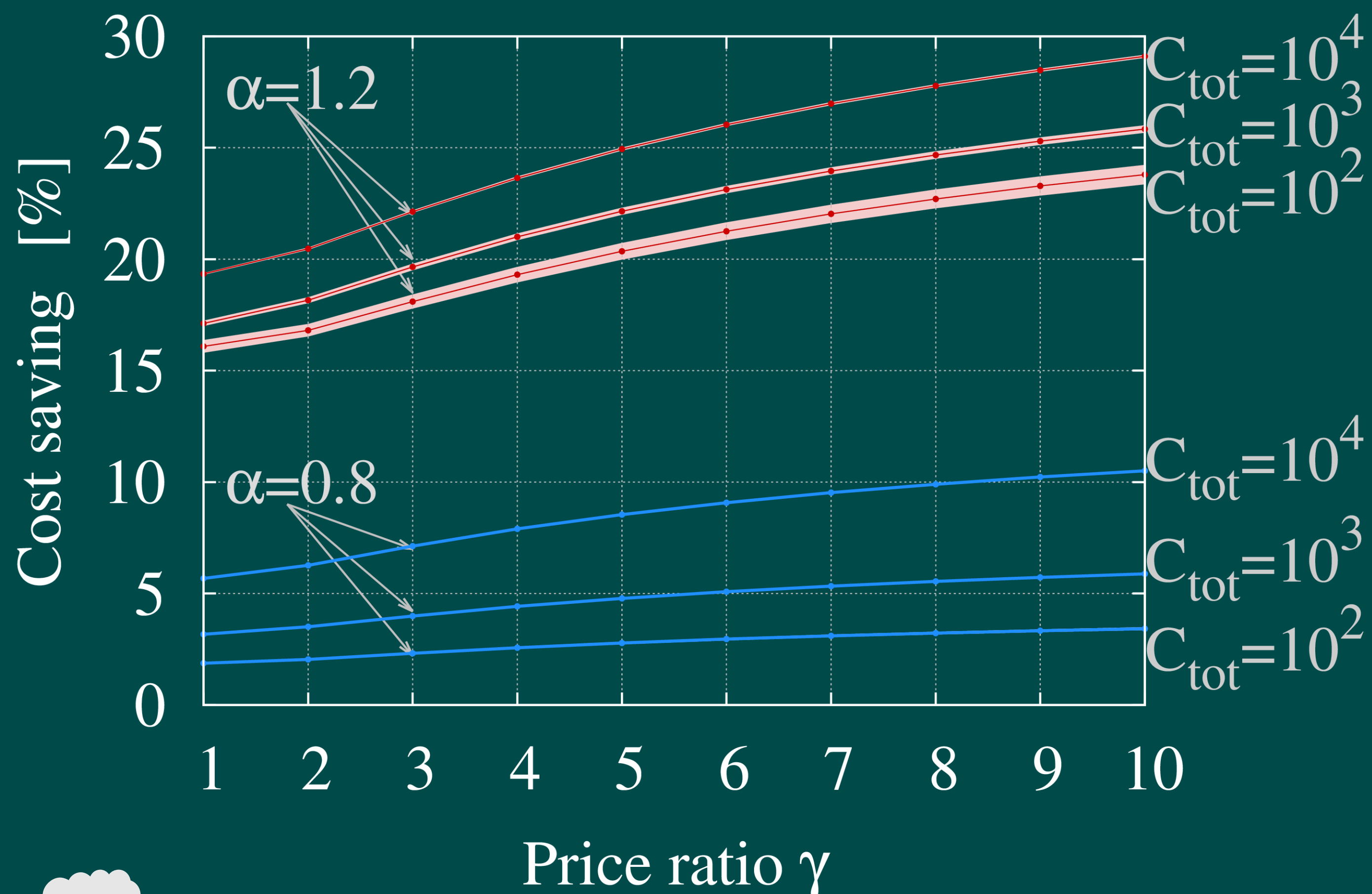
$$\text{Cost saving} = \frac{C_{CLASSIC} - C_{COST\text{-}AWARE}}{C_{CLASSIC}}$$

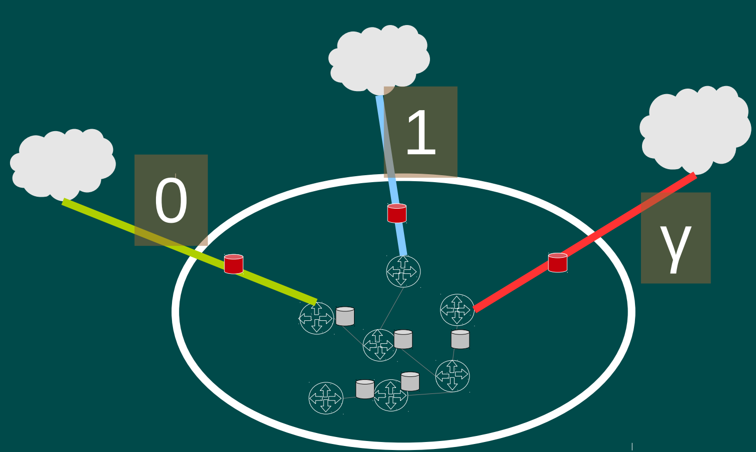
Cost savings

Price diversity ↗
⇨ Savings ↗

Cache Budget ↗
⇨ Savings ↗

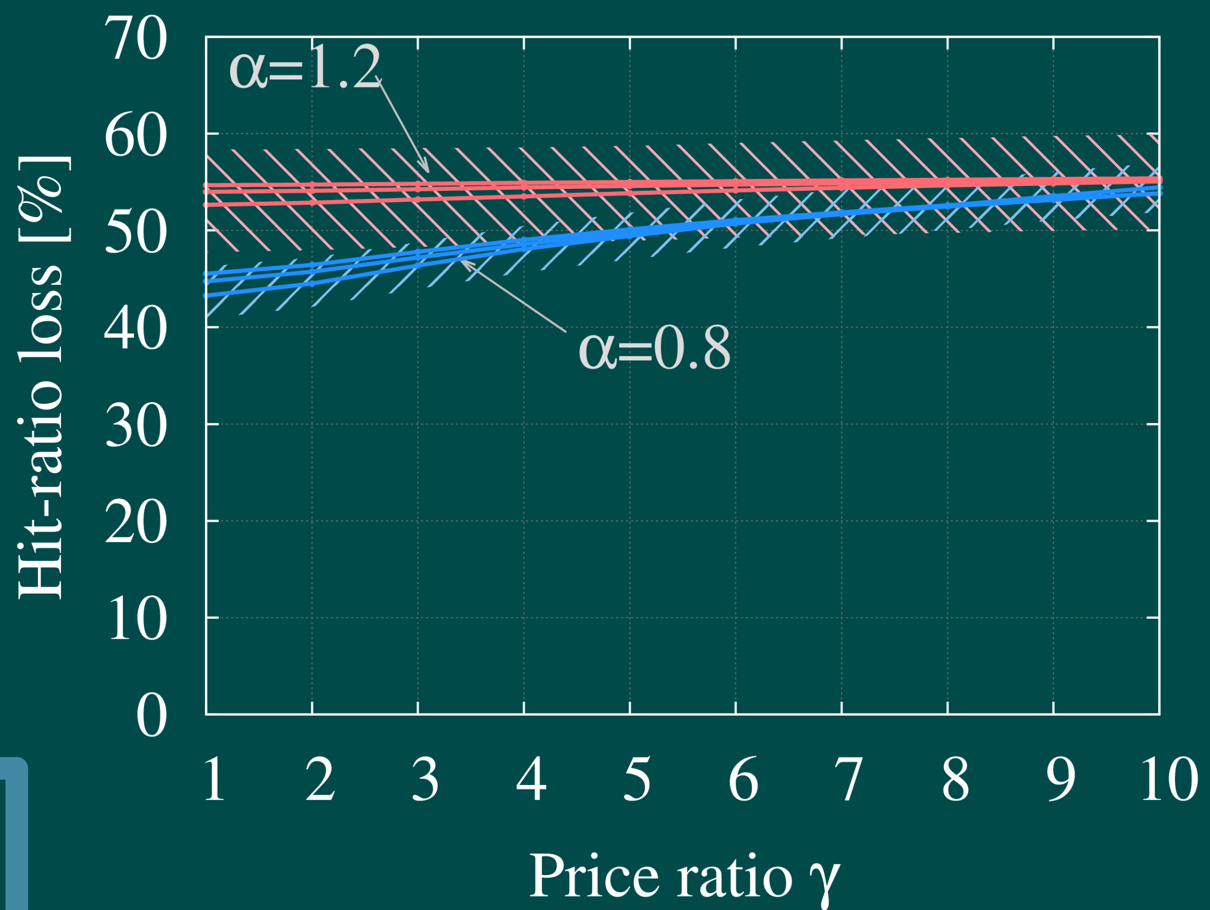
Skew ↗
⇨ Savings ↗





$$\text{Cost saving} = \frac{C_{CLASSIC} - C_{COST\text{-}AWARE}}{C_{CLASSIC}}$$

Hit-ratio loss

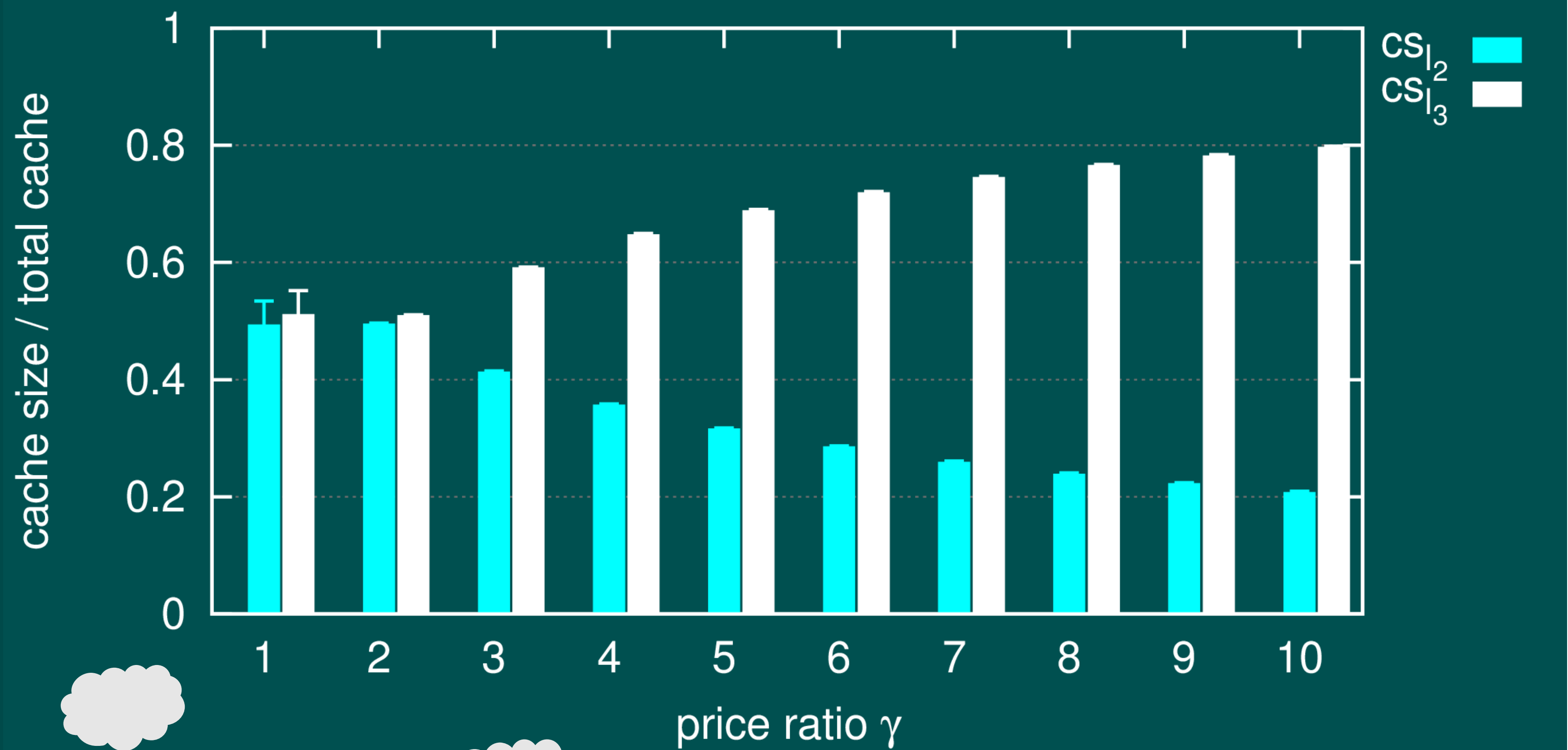


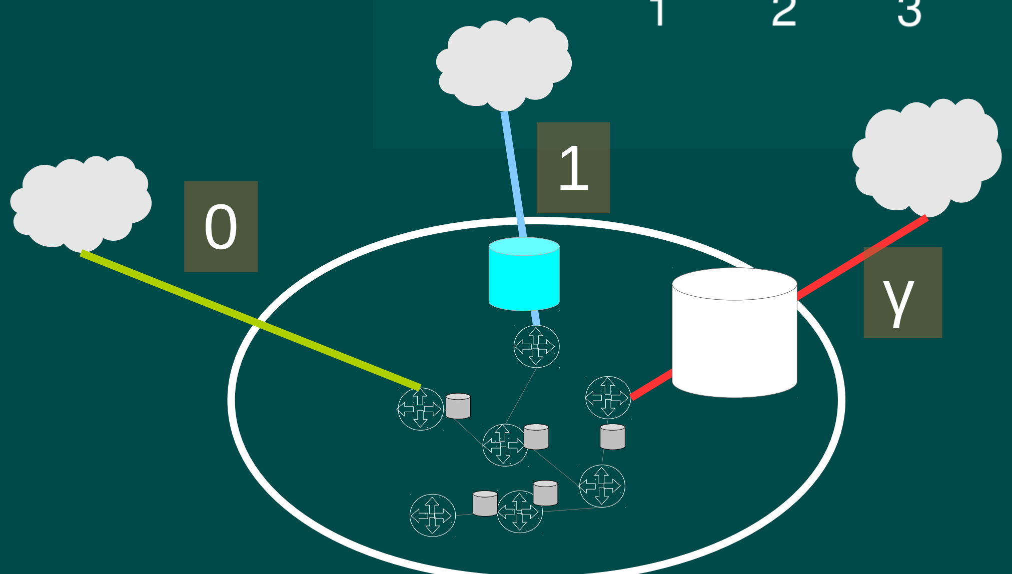
To attain cost savings, an ISP should sacrifice cache efficiency

$$\text{Hit-ratio loss} = \frac{H_{CLASSIC} - H_{COST\text{-}AWARE}}{H_{CLASSIC}}$$

Cache sizing

$\alpha = 1.2;$
$C_{tot} = 10^3$





Cost-aware decision policy

Is cost-awareness implementable in a cache?

Yes!

A. Araldo, D. Rossi, F. Martignon,
"***Design and Evaluation of Cost-aware Information Centric Routers***",
in ACM SIGCOMM Conference on Information-Centric Networking (ICN), Paris, 2014

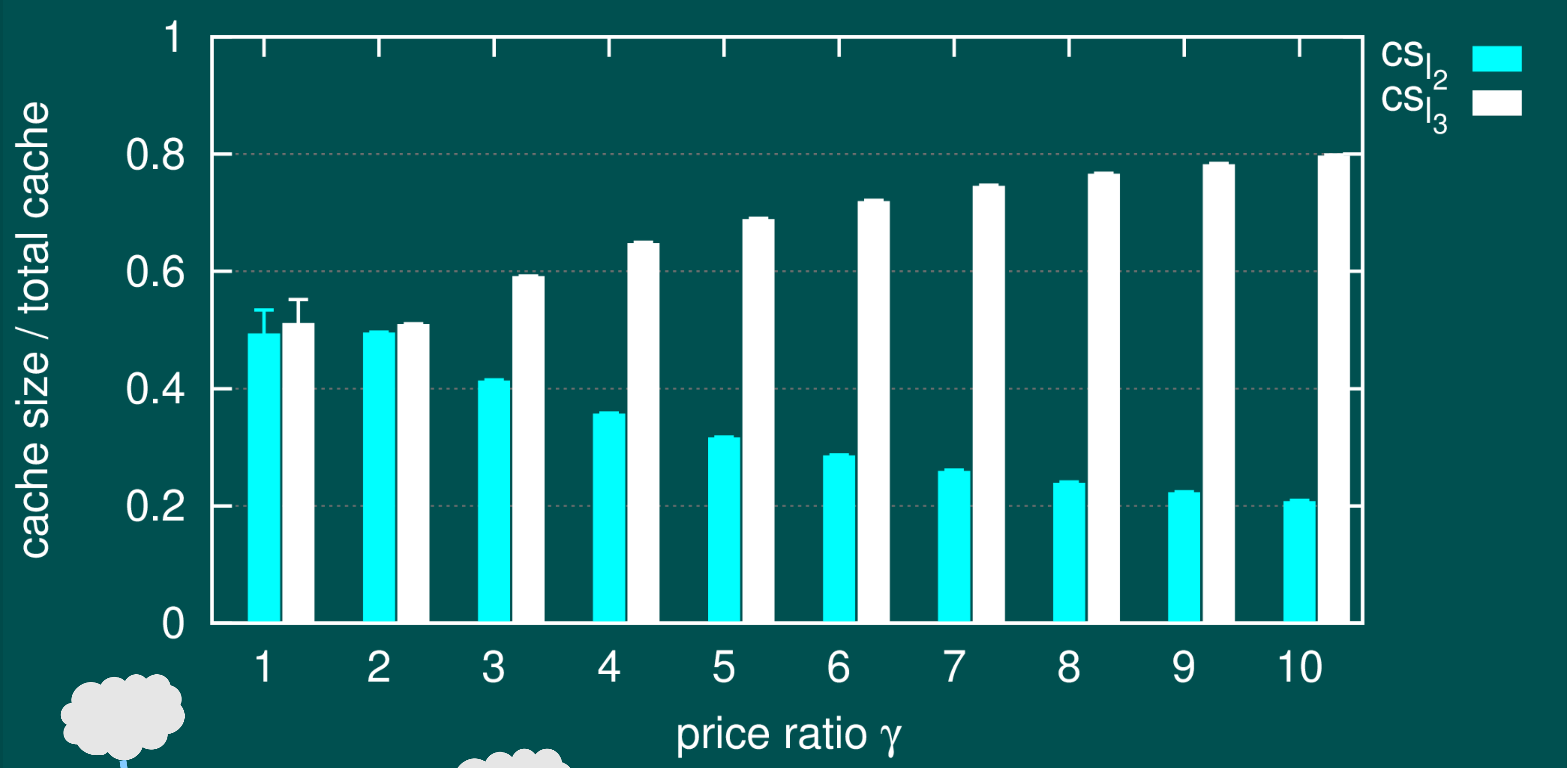
Conclusion

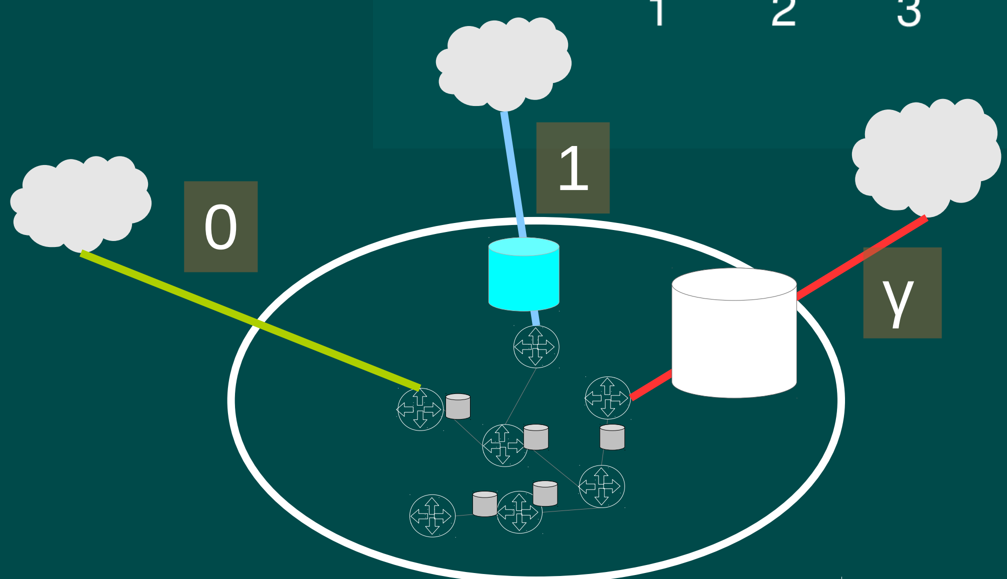
- We proposed two optimization models: CLASSIC vs. COST-AWARE
  - Forwarding, object placement, cache sizing
- We proposed two greedy algorithms
  - We analytically prove their optimality
- Key findings
  - Classic caching is cost-ineffective
  - We need Cost-Aware caching to take into account price heterogeneity
  - It brings sizable savings (~25%)

Backup

Cache sizing details

α = 1.2;
$C_{tot} = 10^3$





- Why with γ=2, the caches on the cheap and expensive links are almost the same?
  - Cache sizing does not depend only on the link prices but also on how many objects we retrieve through each link.
  - In case an object is reachable through l_2 and l_3, the greedy algorithm imposes to use l_2
  - Therefore, more objects are retrieved through l_2 and the cache of l_2 is quite big, even if the price is half the price of l_3
- Why is the variability of the cache size is so high for γ=1?
  - For γ=1 the price has no influence, only the object distribution (which link gives access to each object) matters. Since it changes from a run to another, it brings a large variability to cache sizing results
  - For higher values of γ, also the link price (that does not change from a run to another) impacts the solution and attenuates the variability of the object distribution

Greedy algorithm time complexity

- Polynomial time algorithm
  - Forwarding
    - Find the cheapest link for all the objects
      $O(|L|\cdot|O|)$
      - $|L|$: number of links
      - $|O|$: number of objects
    - Compute the price of each object
      $O(|O|)$
  - Object placement
    - Find the C_{tot} most expensive objects It is a
      $O(|O| \cdot \log |O|)$
  - Cache sizing
    - For each link, count how many selected objects are behind it
      - $O(|L| \cdot |O|)$