Intra-AS Cooperative Caching for Content-Centric Networks

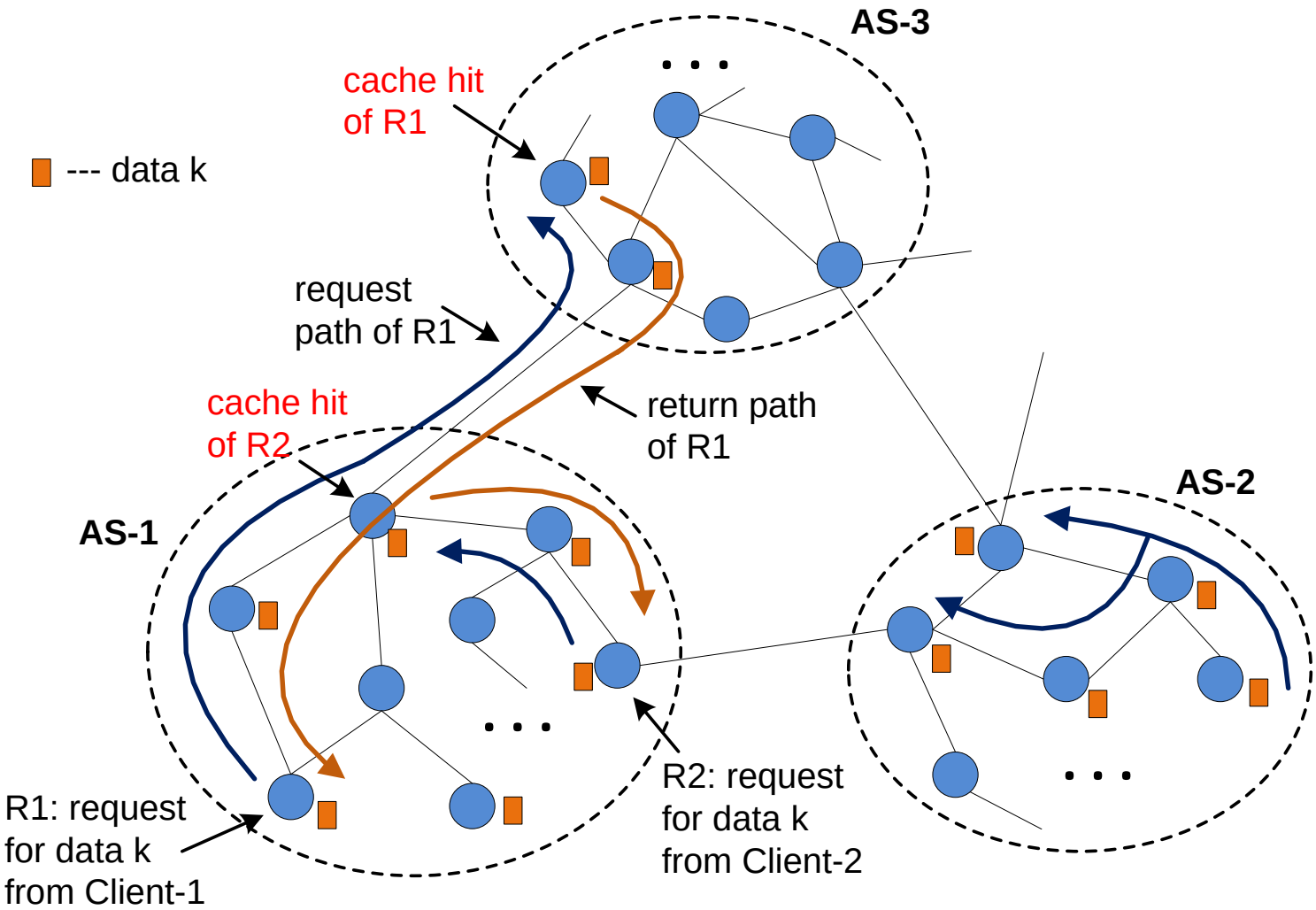
Min(Jason) WANG
jasonwangm@cse.ust.hk

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

August 12, 2013

Outline

Request-Response Scenario in CCN



Motivation for Redundancy Elimination

- Network traffic exhibits **high redundancy**
  - due to content popularity, often, the same content is accessed by many users.
- CCN enables individual nodes to reduce redundancy by managing a local cache
  - the central abstraction is the **named-data**.

Motivation for Redundancy Elimination

- Network traffic exhibits **high redundancy**
  - due to content popularity, often, the same content is accessed by many users.
- CCN enables individual nodes to reduce redundancy by managing a local cache
  - the central abstraction is the **named-data**.
- However, redundancy can freely appear across different nodes:
  - the default **ubiquitous LRU** caching scheme;
  - the support of **multi-path routing**.
- **Controlling the redundancy level** is critical to improving the systematic caching performance of CCN.

Motivation for Redundancy Elimination (II)

- **Fact 1:**
  - a CCN node's caching performance is highly related to its cache size, yet
  - the available caching resource is **rather limited**:
    - buffer memory of IP router ⇝ **content store**.

Motivation for Redundancy Elimination (II)

- **Fact 1:**
  - a CCN node's caching performance is highly related to its cache size, yet
  - the available caching resource is **rather limited**:
    - buffer memory of IP router ⇝ **content store**.
- **Fact 2:**
  - dominant video traffic surge on both wired and wireless network,
    - poses a burden on link bandwidth;
    - increases cross-traffic ⇝ **increased transit-cost**.
  - yet CCN is expected to **greatly offload** cross-AS traffic

Motivation for Redundancy Elimination (II)

- **Fact 1:**
  - a CCN node's caching performance is highly related to its cache size, yet
  - the available caching resource is **rather limited**:
    - buffer memory of IP router ⇝ **content store**.
- **Fact 2:**
  - dominant video traffic surge on both wired and wireless network,
    - poses a burden on link bandwidth;
    - increases cross-traffic ⇝ **increased transit-cost**.
  - yet CCN is expected to **greatly offload** cross-AS traffic
- These two facts dictate **a frugal usage of limited caching resources** within the AS,
  - storing valuable content only;
  - avoiding storage waste by reducing redundancy;

Dimensions of Redundancy Elimination

- **Spacial Dimension:**
  - **vertically**: control the number of copies along the return path
  - **horizontally**: control number of duplicates across neighbour nodes within the AS

Dimensions of Redundancy Elimination

- **Spacial Dimension:**
  - **vertically**: control the number of copies along the return path
  - **horizontally**: control number of duplicates across neighbour nodes within the AS

- **Temporal Dimension:**
  - **actively**: in real time before the data copy is brought into the cache
  - **passively**: on-demand offline after the data has been cached

Dimensions of Redundancy Elimination

- **Spacial Dimension:**
  - **vertically**: control the number of copies along the return path
  - **horizontally**: control number of duplicates across neighbour nodes within the AS

- **Temporal Dimension:**
  - **actively**: in real time before the data copy is brought into the cache
  - **passively**: on-demand offline after the data has been cached

- Previous works proposing new caching schemes for CCN inscribes in the **vertical-active** redundancy elimination (RE) category;

Dimensions of Redundancy Elimination

- **Spacial Dimension:**
  - **vertically**: control the number of copies along the return path
  - **horizontally**: control number of duplicates across neighbour nodes within the AS

- **Temporal Dimension:**
  - **actively**: in real time before the data copy is brought into the cache
  - **passively**: on-demand offline after the data has been cached

- Previous works proposing new caching schemes for CCN inscribes in the **vertical-active** redundancy elimination (RE) category;
- Our work adopts the **horizontal-passive** approach.

Outline

Notations

- $G = (N, E)$: a network managed by a single administrative authority
- S_i: the cache size of node i (in units of chunks)
- K_i: the set of cached items at node i
- N_i: the set of neighbouring nodes of i

Notations

- $G = (N, E)$: a network managed by a single administrative authority
- S_i: the cache size of node i (in units of chunks)
- K_i: the set of cached items at node i
- N_i: the set of neighbouring nodes of i

Cooperation Scope

1. node i can own the view of cached items of nodes in N_i and can utilize them to serve its locally-unsatisfied requests;
2. from node i's perspective, for each $k \in K_i$, if one copy of k exists in N_i, it can purge k to release one caching slot;

Notations

- $G = (N, E)$: a network managed by a single administrative authority
- S_i: the cache size of node i (in units of chunks)
- K_i: the set of cached items at node i
- N_i: the set of neighbouring nodes of i

Cooperation Scope

1. node i can own the view of cached items of nodes in N_i and can utilize them to serve its locally-unsatisfied requests;
2. from node i's perspective, for each $k \in K_i$, if one copy of k exists in N_i, it can purge k to release one caching slot;

- x_{ik}: whether node i should keep item k (1: yes; 0: no)
- $(S_i - \sum_{k \in K_i} x_{ik})$: the released slots from the cooperative RE

Problem Formulation

- **Cooperative Redundancy Elimination** (CRE):

$$\begin{aligned}
\text{max} \quad & \sum_{i \in N} w_i U_i(S_i - \sum_{k \in K_i} x_{ik}) \\
\text{s.t.} \quad & S_i - \sum_{k \in K_i} x_{ik} \geq 0, \ \forall i \in N \\
& x_{ik} + \sum_{j \in N_i, k \in K_j} x_{jk} \geq 1, \ \forall i \in N, k \in K_i \\
& x_{ik} \in \{0, 1\}, \ \forall i \in N, k \in K_i.
\end{aligned}$$

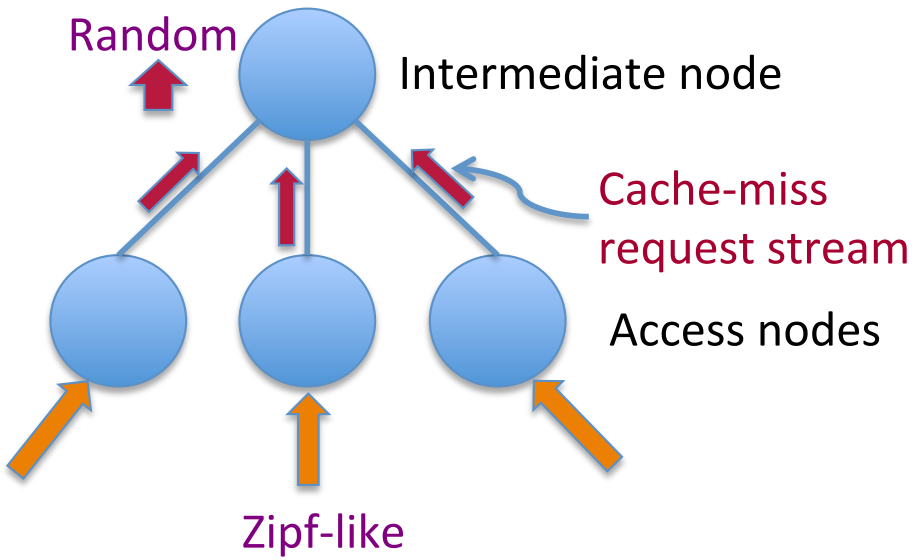
- $U_i(\cdot)$: quantifies the benefit achieved from RE
  - $U_i(v) = \ln(1 + v)$ to achieve the proportional fairness
- w_i: the weight of node i's utility

Modelling Issues (I)

- **Objective function**: sum of weighted utilities
  - Two types of nodes in the AS: access nodes and intermediate nodes.

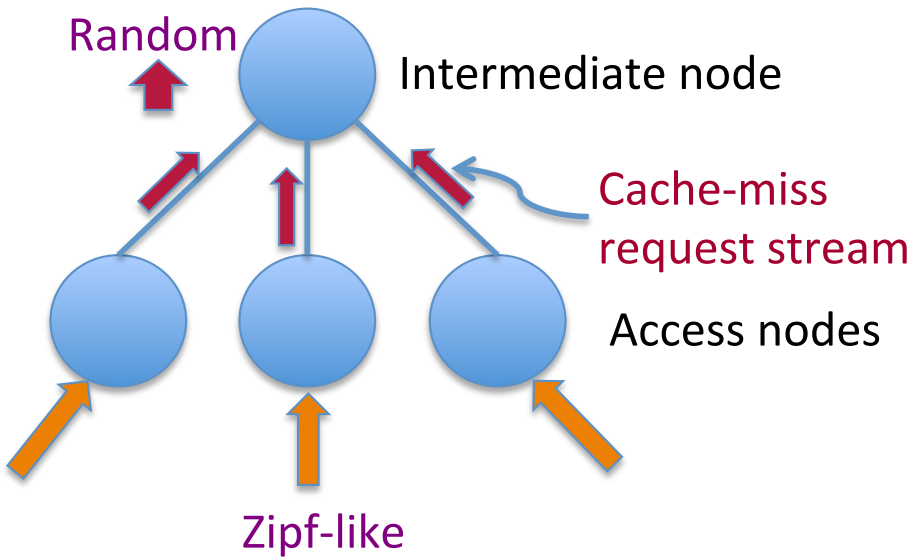
Modelling Issues (I)

- **Objective function**: sum of weighted utilities
  - Two types of nodes in the AS: access nodes and intermediate nodes.
  - Due to "cache filtering effect", the caching performance of access nodes plays the major role in systematic caching performance.



Modelling Issues (I)

- **Objective function**: sum of weighted utilities
  - Two types of nodes in the AS: access nodes and intermediate nodes.
  - Due to "cache filtering effect", the caching performance of access nodes plays the major role in systematic caching performance.
  - Value more the gain of access nodes: larger w_i;
  - Value less the gain of intermediate nodes: smaller w_i;



Modelling Issues (II)

- **Limited cooperation scope**
  - The accepted redundancy level reflects the trade-off between two goals
    - performance goal: minimizing the access latency;
    - cost goal: maximizing the diversity of cached items to reduce cross-traffic;

Modelling Issues (II)

- **Limited cooperation scope**
  - The accepted redundancy level reflects the trade-off between two goals
    - performance goal: minimizing the access latency;
    - cost goal: maximizing the diversity of cached items to reduce cross-traffic;
  - Lessons learned from previous study:
    - Acquiring optimal degree of duplication per se is challenging;
    - Often couples with network topology and needs to be handled together with request routing, rendering solutions non-scalable.

Modelling Issues (II)

- **Limited cooperation scope**
  - The accepted redundancy level reflects the trade-off between two goals
    - performance goal: minimizing the access latency;
    - cost goal: maximizing the diversity of cached items to reduce cross-traffic;
  - Lessons learned from previous study:
    - Acquiring optimal degree of duplication per se is challenging;
    - Often couples with network topology and needs to be handled together with request routing, rendering solutions non-scalable.
  - One-hop cooperation scope
    - eliminating duplicates in small groups controls overall redundancy level;
    - severs the dependence on request routing;
    - reduces the amount of signalling traffic needed for cooperation; yet,
    - only subtly prolongs access latency for evicted items;

Greedy Heuristic

- CRE is Non-linear Integer Programming and is NP-hard.
- With no fairness consideration and in absence of storage capacity constraint, CRE degenerates to solving a sequence of MDS problems.
  - each cached item k leads to an induced graph $G_k = (N, E_k)$;
  - calculate the MDS for each G_k;

Greedy Heuristic

- CRE is Non-linear Integer Programming and is NP-hard.
- With no fairness consideration and in absence of storage capacity constraint, CRE degenerates to solving a sequence of MDS problems.
  - each cached item k leads to an induced graph $G_k = (N, E_k)$;
  - calculate the MDS for each G_k;
- The greedy heuristic for MDS tends to pick nodes with high degrees.

  Greedy leads to a $\ln \Delta$-approx solution.

Greedy Heuristic

- CRE is Non-linear Integer Programming and is NP-hard.
- With no fairness consideration and in absence of storage capacity constraint, CRE degenerates to solving a sequence of MDS problems.
  - each cached item k leads to an induced graph $G_k = (N, E_k)$;
  - calculate the MDS for each G_k;
- The greedy heuristic for MDS tends to pick nodes with high degrees.

  Greedy leads to a $\ln \Delta$-approx solution.
- Two observations:
  1. Nodes with high degrees tend to be intermediate nodes that are of less importance to the systematic caching performance;
  2. A node with high degree locates in a better position for eliminating redundancy.

Greedy Heuristic

- CRE is Non-linear Integer Programming and is NP-hard.
- With no fairness consideration and in absence of storage capacity constraint, CRE degenerates to solving a sequence of MDS problems.
  - each cached item k leads to an induced graph $G_k = (N, E_k)$;
  - calculate the MDS for each G_k;
- The greedy heuristic for MDS tends to pick nodes with high degrees.

  Greedy leads to a $\ln \Delta$-approx solution.
- Two observations:
  1. Nodes with high degrees tend to be intermediate nodes that are of less importance to the systematic caching performance;
  2. A node with high degree locates in a better position for eliminating redundancy.

The greedy heuristic for MDS can be implemented in a distributed way [F. Kuhn PODC'03].

Intra-AS Cache Cooperation Framework (I)

- Each CCN node is taken care of by any specified cache management scheme, e.g., the default ubiquitous LRU.
- Ideally, the cache space of an AS should abound in popular items requested by its served clients.

Intra-AS Cache Cooperation Framework (I)

- Each CCN node is taken care of by any specified cache management scheme, e.g., the default ubiquitous LRU.
- Ideally, the cache space of an AS should abound in popular items requested by its served clients.

- Periodically, each CCN node exchanges with its one-hop neighbours the cache "summary".

Intra-AS Cache Cooperation Framework (I)

- Each CCN node is taken care of by any specified cache management scheme, e.g., the default ubiquitous LRU.
- Ideally, the cache space of an AS should abound in popular items requested by its served clients.

- Periodically, each CCN node exchanges with its one-hop neighbours the cache "summary".
  - "summary" encodes the information of currently cached items.
  - caveat: to avoid exposing short-lived items.

Intra-AS Cache Cooperation Framework (II)

- A fresh round of cooperative redundancy elimination will be launched,

Intra-AS Cache Cooperation Framework (II)

- A fresh round of cooperative redundancy elimination will be launched,
  - when the "redundancy monitoring metric" indicates the potential benefits of doing so based on collected summaries.

Intra-AS Cache Cooperation Framework (II)

- A fresh round of cooperative redundancy elimination will be launched,
  - when the "redundancy monitoring metric" indicates the potential benefits of doing so based on collected summaries.

- If a request cannot be served locally, probe the neighbours to see whether an early cache hit can happen, in light of exchanged summaries.

Intra-AS Cache Cooperation Framework (II)

- A fresh round of cooperative redundancy elimination will be launched,
  - when the "redundancy monitoring metric" indicates the potential benefits of doing so based on collected summaries.

- If a request cannot be served locally, probe the neighbours to see whether an early cache hit can happen, in light of exchanged summaries.
  - in view of the existence of false-positive probe (due to stale summary),
  - probing action is confined with one-hop neighbours,
  - and only at the first hop node;

Intra-AS Cache Cooperation Framework (II)

- A fresh round of cooperative redundancy elimination will be launched,
  - when the "redundancy monitoring metric" indicates the potential benefits of doing so based on collected summaries.

- If a request cannot be served locally, probe the neighbours to see whether an early cache hit can happen, in light of exchanged summaries.
  - in view of the existence of false-positive probe (due to stale summary),
  - probing action is confined with one-hop neighbours,
  - and only at the first hop node;

Cache Cooperation + Cooperative Redundancy Elimination

Experimental Setup (I)

- Network topologies:

Topology	Nodes	Edges	Max degree
AS 1755	172	381	15
AS 3967	201	434	15
Brite1	200	404	16
Brite2	200	404	11

Experimental Setup (I)

- Network topologies:

Topology	Nodes	Edges	Max degree
AS 1755	172	381	15
AS 3967	201	434	15
Brite1	200	404	16
Brite2	200	404	11

- Workloads: generated using ProWGen.

	Zipf-slope	Object size	One-timers	Unique objects
Trace-1	[0.90, 0.96]	[5, 20]	[60, 70]%	[35, 40]%
Trace-2	[0.70, 0.76]	[5, 20]	[60, 70]%	[35, 40]%

Experimental Setup (II)

- CCN simulator:
  - implemented on top of OmNet++;
  - incorporate three basic components: CS, PIT, and FIB;
  - support per-chunk request, chunk-based caching, and request aggregation;
  - assume the shortest-path routing;

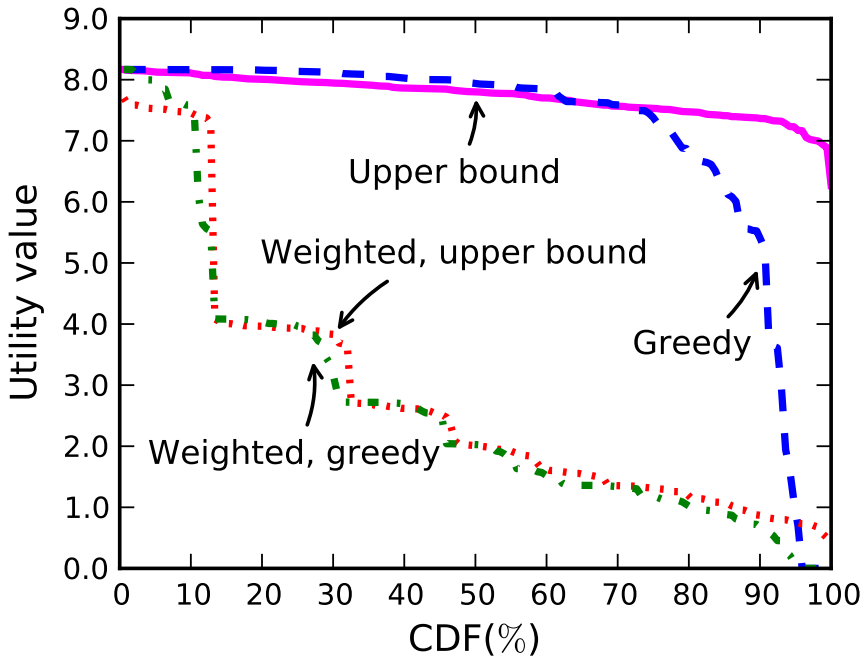
- The arrival process of requests (6,000) for objects is Poissonian.
- A request for object translates into a sequence of chunk requests;
  - the sending window: 1

Efficiency of the Greedy Heuristic

- We compare with an "upper bound" of CRE:
  - relaxing Integer constraint of x_{ik}
- Derive a decentralized algorithm to relaxed-version by using non-linear Gauss-Seidel procedure and Dual decomposition.

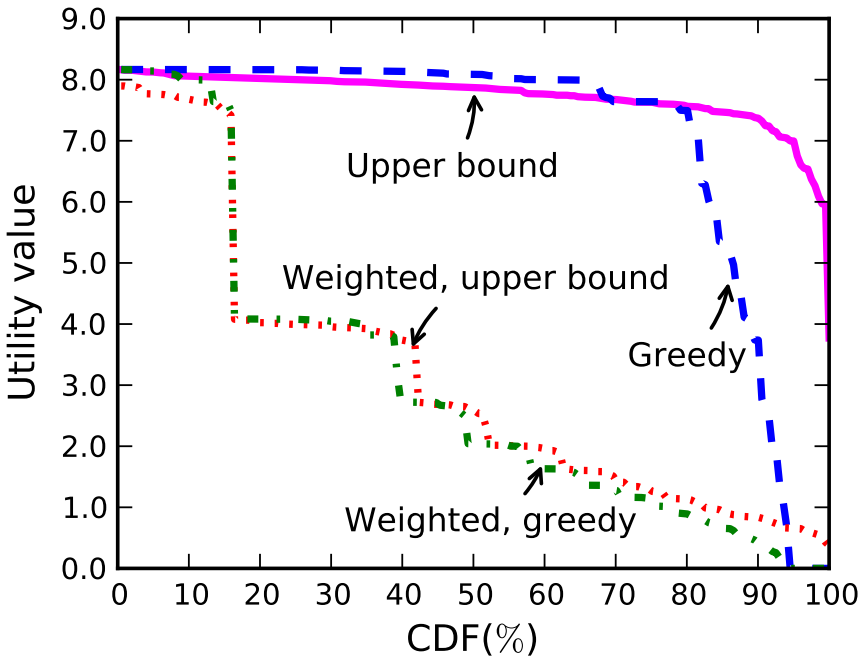
Efficiency of the Greedy Heuristic

- We compare with an "upper bound" of CRE:
  - relaxing Integer constraint of x_{ik}
- Derive a decentralized algorithm to relaxed-version by using non-linear Gauss-Seidel procedure and Dual decomposition.
- AS 1755:



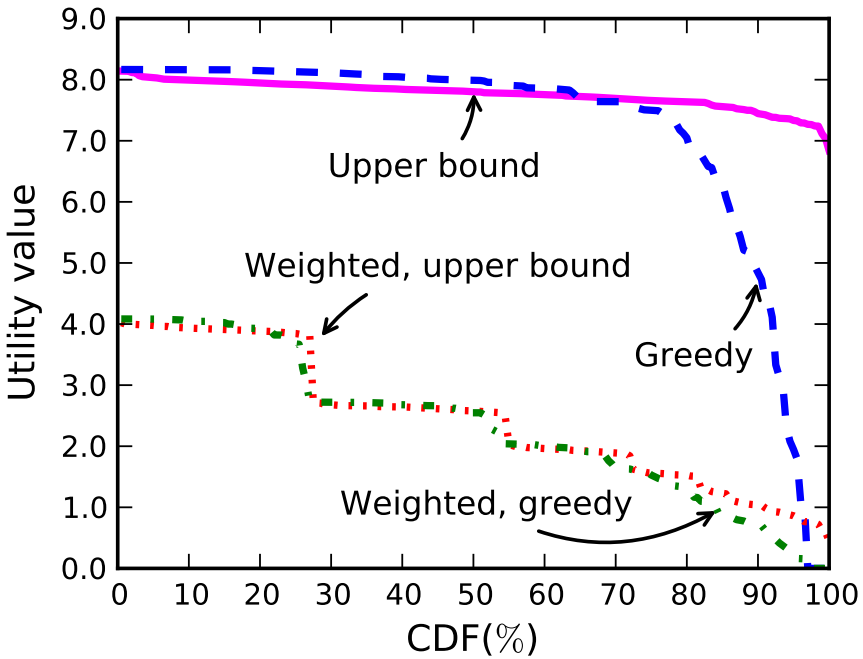
Efficiency of the Greedy Heuristic

- We compare with an "upper bound" of CRE:
  - relaxing Integer constraint of x_{ik}
- Derive a decentralized algorithm to relaxed-version by using non-linear Gauss-Seidel procedure and Dual decomposition.
- AS 3967:



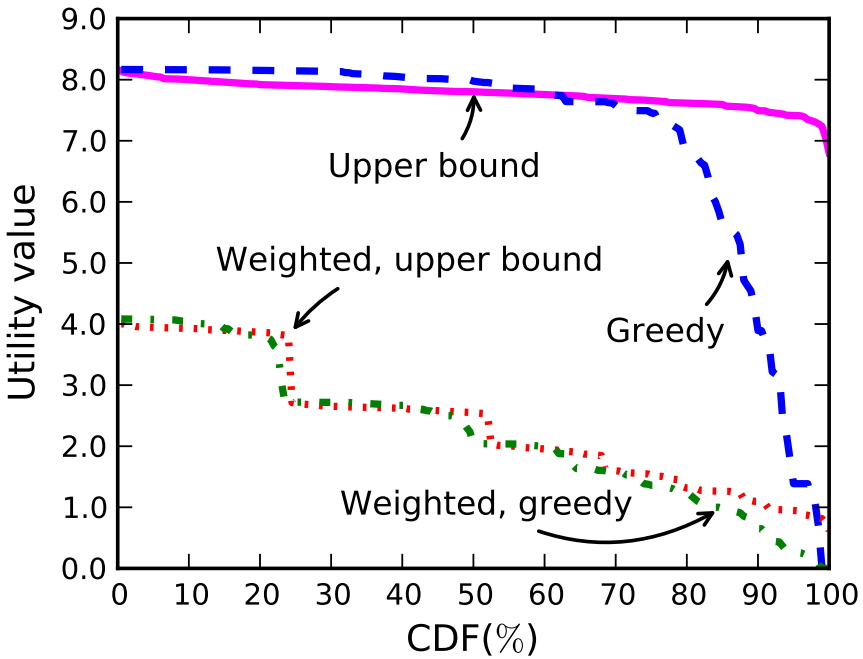
Efficiency of the Greedy Heuristic

- We compare with an "upper bound" of CRE:
  - relaxing Integer constraint of x_{ik}
- Derive a decentralized algorithm to relaxed-version by using non-linear Gauss-Seidel procedure and Dual decomposition.
- Brite1:



Efficiency of the Greedy Heuristic

- We compare with an "upper bound" of CRE:
  - relaxing Integer constraint of x_{ik}
- Derive a decentralized algorithm to relaxed-version by using non-linear Gauss-Seidel procedure and Dual decomposition.
- Brite2:



Core Benefits of Cooperation Scheme (I)

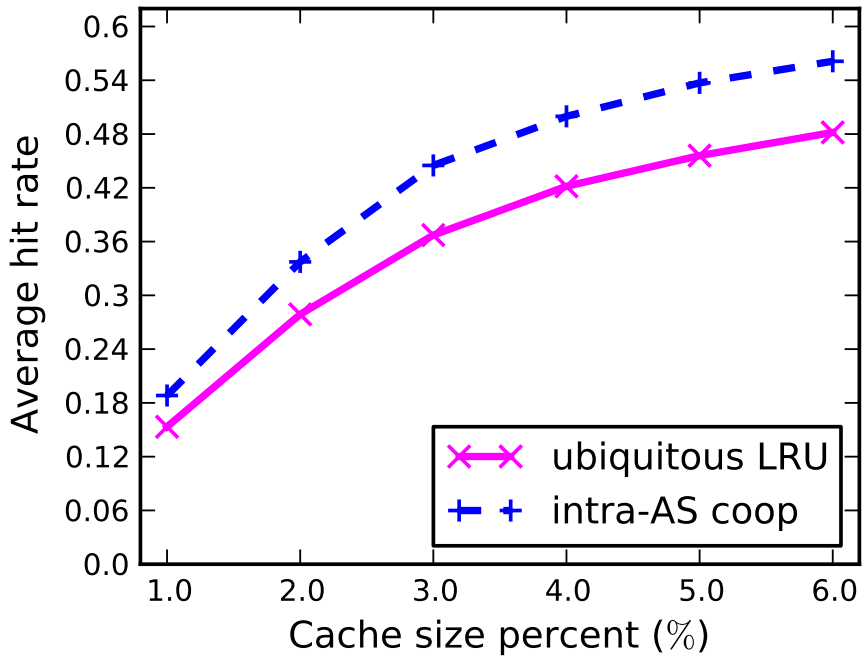
- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.

Core Benefits of Cooperation Scheme (I)

- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.
- Report results on Trace-1, since similar results are observed for Trace-2.

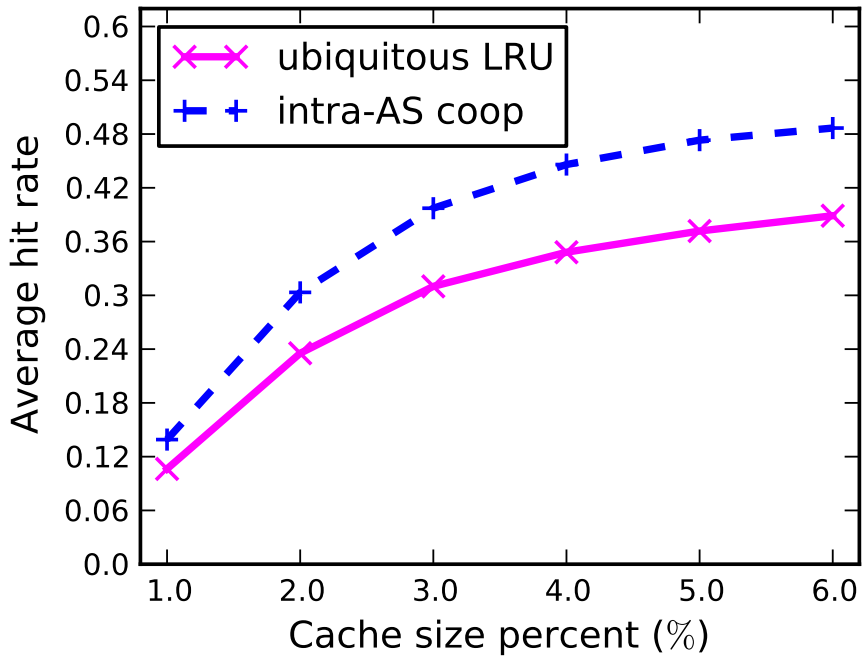
Core Benefits of Cooperation Scheme (I)

- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.
- AS 1755:



Core Benefits of Cooperation Scheme (I)

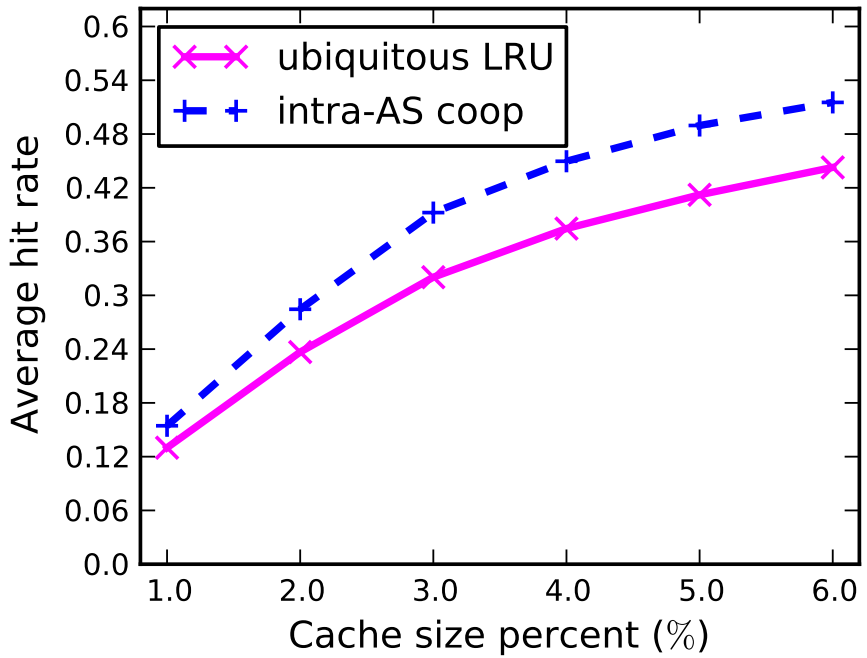
- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.
- AS 3967:



Core Benefits of Cooperation Scheme (I)

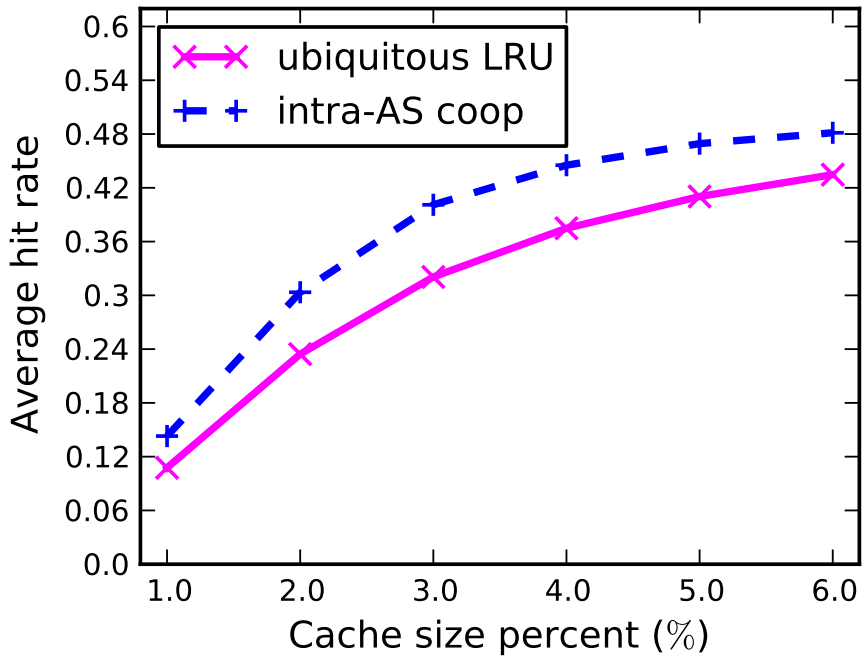
- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.

- Brite1:



Core Benefits of Cooperation Scheme (I)

- Cache hit rate:
  - a cache hit: a request either locally served or successfully fetched from neighbours.
- Brite2:



Core Benefits of Cooperation Scheme (II)

- Bandwidth saving:
  - unsatisfied requests by caches within the AS have to fetch data from original content server and this yields the cross-traffic.
- The amount of cross-traffic (in GB): one chunk is 10KB

Core Benefits of Cooperation Scheme (II)

- Bandwidth saving:
  - unsatisfied requests by caches within the AS have to fetch data from original content server and this yields the cross-traffic.
- The amount of cross-traffic (in GB): one chunk is 10KB

AS	Method	1%	2%	3%	4%	5%	6%
1755	ubi-LRU	4.06	2.70	1.57	0.69	0.34	0.19
	AS-coop	3.91	2.47	1.25	0.47	0.21	0.12
3967	ubi-LRU	5.33	3.62	1.97	0.83	0.37	0.20
	AS-coop	5.13	3.15	1.40	0.42	0.18	0.10
Brite1	ubi-LRU	4.73	3.38	2.06	1.16	0.70	0.45
	AS-coop	4.61	3.19	1.77	0.93	0.53	0.31
Brite2	ubi-LRU	4.86	3.08	1.44	0.54	0.24	0.12
	AS-coop	4.67	2.70	1.00	0.33	0.15	0.11

Core Benefits of Cooperation Scheme (III)

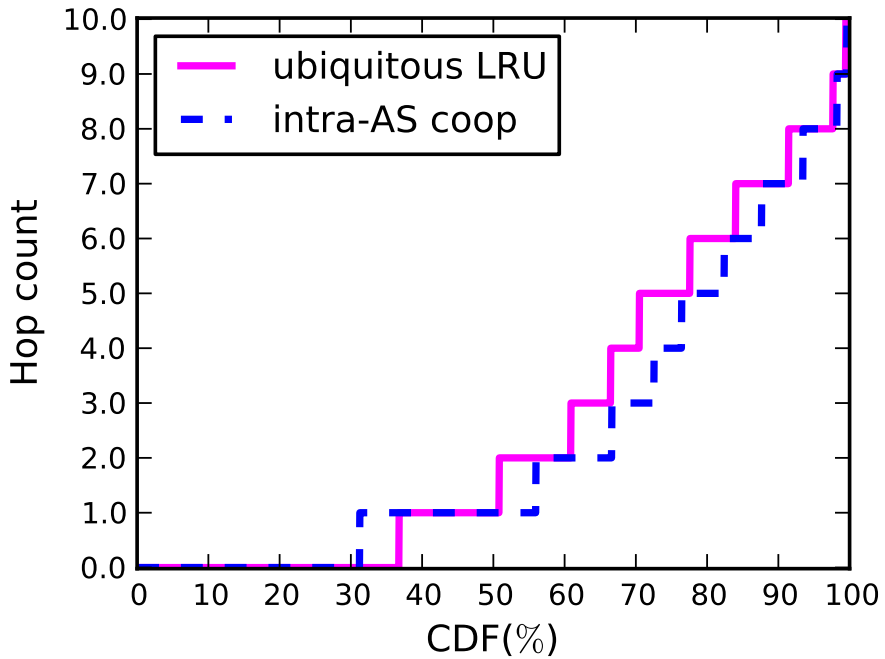
- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;

Core Benefits of Cooperation Scheme (III)

- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;
- Below reports the CDF of hop counts when the cache size is 3%.

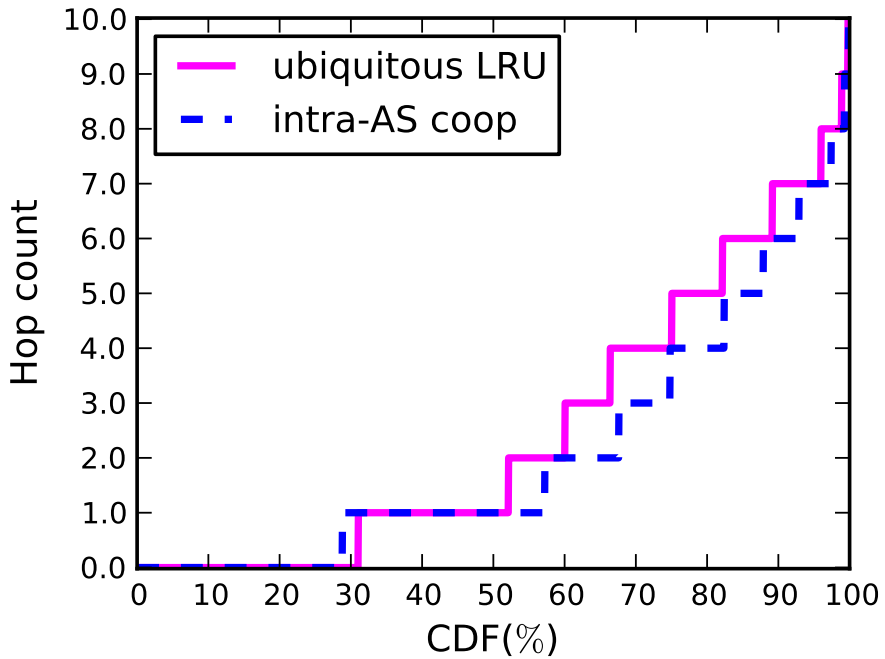
Core Benefits of Cooperation Scheme (III)

- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;
- AS 1755:



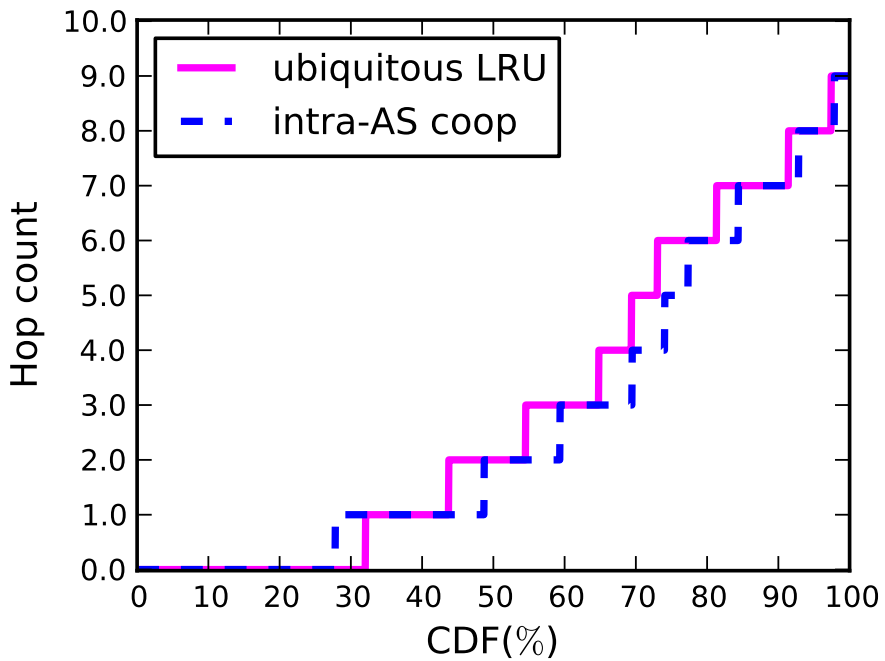
Core Benefits of Cooperation Scheme (III)

- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;
- AS 3967:



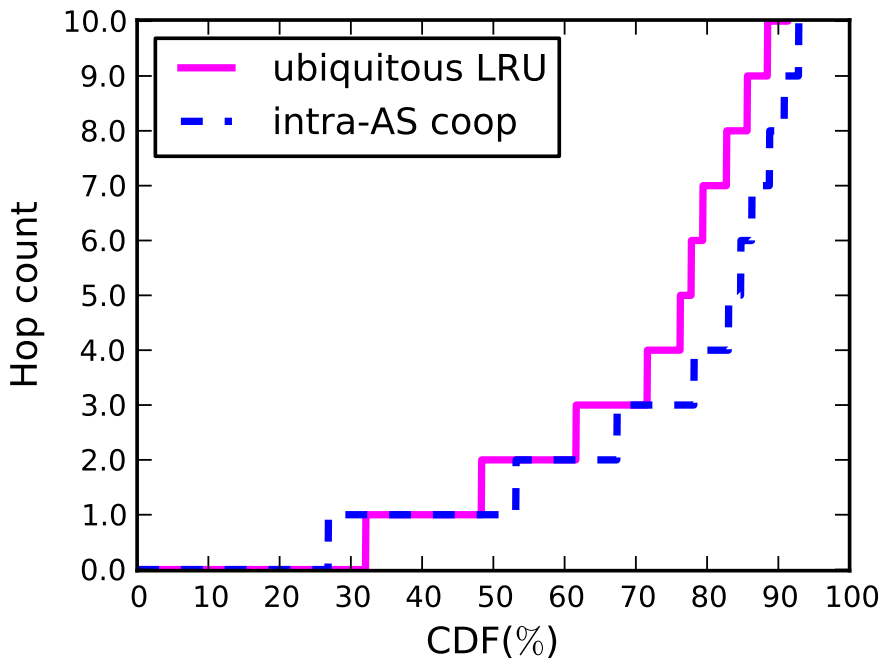
Core Benefits of Cooperation Scheme (III)

- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;
- Brite1:



Core Benefits of Cooperation Scheme (III)

- Average hop count of requests:
  - intuitively, cache cooperation tends to generate more internal traffic;
  - measure the distribution of the hop count with which each chunk-request is served;
- Brite2:



Outline

Conclusion

- We studied the caching problem in CCN following a different approach, advocating cooperative redundancy elimination after data have been cached.

Conclusion

- We studied the caching problem in CCN following a different approach, advocating cooperative redundancy elimination after data have been cached.

- Two benefits reaped from intra-AS cache cooperation scheme:
  - caching slots released from RE can be used to cache other popular items;
  - a broader view of cached items at neighbours helps serve locally unsatisfied requests;

Conclusion

- We studied the caching problem in CCN following a different approach, advocating cooperative redundancy elimination after data have been cached.

- Two benefits reaped from intra-AS cache cooperation scheme:
  - caching slots released from RE can be used to cache other popular items;
  - a broader view of cached items at neighbours helps serve locally unsatisfied requests;

- **Initial results**:
  - greedy is efficiency in eliminating redundancy;
  - intra-AS cache-coop improves caching performance of access routers;
  - and reduces the AS cross-traffic without overloading internal links.

Thank you very much!