Stable Content-peering of Autonomous Systems in a Content-centric Network

Valentino Pacifici and György Dán
ACCESS Linnaeus Center, School of Electrical Engineering
KTH, Royal Institute of Technology, Stockholm, Sweden
E-mail: {pacifici,gyuri}@kth.se

***Abstract*—A future content-centric Internet would likely consist of autonomous systems (ASes) just like today's Internet. It would thus be a network of interacting cache networks, each of them optimized for local performance. To understand the influence of interactions between autonomous cache networks, in this paper we consider ASes that maintain peering agreements with each other for mutual benefit, and engage in content-level peering to leverage each others' cache contents. We propose a model of the interaction and the coordination between the caches managed by peering ASes. We address whether stable and efficient content-level peering can be implemented without explicit coordination between the neighboring ASes in order for the system to be stable. We show that content-level peering leads to stable cache configurations, and that avoiding simultaneous updates by peering ISPs provides faster and more cost efficient convergence to a stable configuration. We validate our analytical results using simulations on the measured peering topology of more than 600 ASes.**

I. Introduction

Recent proposals to re-design the Internet with the aim of facilitating content delivery share the common characteristic that caches are an integral part of the protocol stack [1], [2], [3]. In these content-centric networks users generate interest messages for content, which are forwarded until the content is found in a cache or the interest message reaches one of the content's custodians. The resulting network is often modeled as a network of interacting caches and several recent works aimed at optimizing its global performance [4], [5], [6].

Similar to the structure of today's Internet, a future content-centric network is likely to be a network of autonomous systems (AS). ASes are typically profit seeking entities and use an interior gateway protocol (IGP) for optimizing their internal routes. Nevertheless, they maintain client-provider and peering business relations with adjacent ASes [7], and they coordinate with each other using the Border Gateway Protocol (BGP), which allows them to exchange reachability information with their neighbors.

ASes are likely to play a similar role in a future content-centric Internet as they do today, and thus, instead of a single cache network dimensioned and managed for optimal global performance, the content-centric Internet will be a network of cache networks, each of them optimized for local performance. To make such a network of cache networks efficient, we need to understand the potential consequences of interaction between the individual cache networks in terms of stability and convergence of the cache contents, and the potential impact of coordination between the networks of caches.

In this work we consider a network of ASes that maintain peering agreements with each other for mutual benefit. The traffic exchanged between the peering AS is not charged, unlike the traffic that each AS exchanges with its transit provider. The ASes maintain their own cache networks, and they engage in *content-level peering* in order to leverage each others' cache contents, which in principle should enable them to decrease their transit traffic costs. The interaction between the caches could, however, lead to unforeseen instability and oscillations, as in the case of BGP. Thus, a fundamental question that one needs to answer is whether stable and efficient content-level peering can be implemented without explicit coordination between the neighboring cache networks, and if so, whether the system is going to be stable.

In this paper we address these questions by proposing a model of the interaction and the coordination between the caches managed by peering ASes. We show that, with or without coordination, content-peering leads to stable cache configurations. Finally, we investigate how the convergence speed and the cost efficiency of the reached cache configuration are affected by coordination. We illustrate the analytical results using simulations on the measured peering topology of more than 600 ASes.

II. System Model

We consider a set N of autonomous ISPs. Each ISP $i \in N$ is connected via peering links to some ISPs $j \in N$. We model the peering links among ISPs by an undirected graph $\mathcal{G} = (N, E)$, called the *peering graph*. We call $\mathcal{N}(i)$ the set of neighbors of ISP $i \in N$ in the peering graph, i.e. $\mathcal{N}(i) = \{j|(i,j) \in E\}$. Apart from the peering links, every ISP can have one or more transit links.

A. Content Items and Caches

We denote the set of content items by $\mathcal{O}$. We follow common practice and consider that every item $o \in \mathcal{O}$ has unit size[8], [9], which is a reasonable simplification if content is divisible into unit-sized chunks. Each item $o \in \mathcal{O}$ is permanently stored at one or more *content custodians* in the network. A custodian in ISP i is a customer of ISP i that serves content to users. We denote by $\mathcal{H}_i$ the set of items kept by the custodians within

ISP i. Since the custodians are autonomous entities, ISP i cannot influence $\mathcal{H}_i$. Similar to other modeling works, we adopt the Independent Reference Model (IRM) [10], [8], [9] for the arrival process of interest messages for the items in $\mathcal{O}$ generated by the local users of the ISPs. Under the IRM, the probability that the next interest message at ISP i is for item o is independent of earlier events. An alternative definition of the IRM is that the inter-arrival time of interest messages for item o at ISP i follows an exponential distribution with distribution function $F_i^o(x) = 1 - e^{-w_i^o x}$, where $w_i^o \in \mathbb{R}_+$ is the average arrival intensity of interest messages for item o at ISP i.

Each ISP $i \in N$ maintains a network of content caches within its network, and jointly engineers the eviction policies of the caches, the routing of interest messages and the routing of contents via the caches to optimize performance. The set of items cached by ISP i is described by the set $\mathcal{C}_i \in \mathfrak{C}_i = \{\mathcal{C} \subset \mathcal{O} : |\mathcal{C}| = K_i\}$, where $K_i \in \mathbb{N}_+$ is the maximum number of items that ISP i can cache. A *summary cache* in each ISP keeps track of the configuration of the local caches and of the content stored in local custodians, it thus embodies the information about what content is available within ISP i. We call $\mathcal{L}_i = \mathcal{C}_i \cup \mathcal{H}_i$ the set of items available within ISP i.

We denote by $\alpha_i > 0$ the unit cost of retrieving an item from a local cache. We consider that retrieving an item from a peering ISP is not more costly than retrieving it locally. The assumption of equal local and peering cost is justified by the fact that in general, once a peering link has been established, there is no additional cost for traffic. The traffic on the transit link is charged by volume with unit cost γ_i, and we make the reasonable assumption that $\gamma_i > \alpha_i$.

B. Content-peering

Peering ISPs *synchronously exchange information* about the contents of their summary caches *periodically*, at the end of every time slot. Upon receiving an interest message for an item, ISP i consults its summary cache to see if the item is available locally. If it is, ISP i retrieves the item from a local cache. Otherwise, it consults its most recent copy of the summary caches of its peering ISPs $\mathcal{N}(i)$. In case a peering ISP $j \in \mathcal{N}(i)$ is caching the item, ISP i forwards the request to ISP j and fetches the content. If not, the interest message is sent to a transit ISP through a transit link.

Using the above notation, and denoting by $\mathcal{C}_{-i}$ the set of the cache configurations of every ISP other than ISP i, we can express the cost of ISP i to obtain item $o \in \mathcal{O}$ as

$$C_i^o(\mathcal{C}_i, \mathcal{C}_{-i}) = w_i^o \begin{cases} \alpha_i & \text{if } o \in \mathcal{L}_i \cup \mathcal{R}_i \\ \gamma_i & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathcal{R}_i = \bigcup_{j \in \mathcal{N}(i)} \mathcal{L}_j$ is the set of items ISP i can obtain from its peering ISPs.

C. Caching Policies and Cost Minimization

A content item o that is not available either locally or from a peering ISP is obtained through a transit link, and is a candidate for caching in ISP i. The cache eviction policy of ISP i determines if item o should be cached, and if so, which item $p \in \mathcal{C}_i$ should be evicted to minimize the expected future cost. There is a plethora of cache eviction policies for this purpose, such as Least recently used (LRU), Least frequently used (LFU), LRFU (we refer to [11] for a survey of some recent algorithms). We model the eviction decision as a comparison of the estimate $\overline{w}_i^o$ of the arrival intensity w_i^o for the item o to be cached and that for the items p in the cache, $\overline{w}_i^p$.

Perfect information: Under perfect information $\overline{w}_i^o = w_i^o$, and only the items with highest costs $C_i^o(\mathcal{C}_i, \mathcal{C}_{-i})$ are cached.

Imperfect information: Under imperfect information $\overline{w}_i^o$ is a random variable with mean w_i^o, and we assume that the probability of misestimation decreases exponentially with the difference in arrival intensities, that is, for $w_i^o > w_i^p$ we have

$$P(\overline{w}_i^o < \overline{w}_i^p) \propto \epsilon e^{-\frac{1}{\beta}(w_i^o - w_i^p)}. \tag{2}$$

This assumption is reasonable for both the LRU and the LFU cache eviction policies. Under LRU the cache miss rate was shown to be an exponentially decreasing function of the item popularity [9]. Under a perfect LFU policy, if we denote the interval over which the request frequencies are calculated by τ, then $\overline{w}_i^p$ follows a Poisson distribution with parameter $w_i^p \tau$. Hence the difference $k = \overline{w}_i^o \tau - \overline{w}_i^p \tau$ of two estimates follows the Skellam distribution [12] and the probability of misestimation decreases exponentially in $w_i^o - w_i^p$ for $\tau > 0$.

III. Results

We start the analysis by considering the case of perfect information, that is, when the cache eviction policies are not prone to misestimation.

The key question we ask is whether the profit-maximizing behavior of the individual ISPs would allow the emergence of an equilibrium allocation of items. If an equilibrium cannot be reached then, as a consequence of coordination, every ISP could evict and fetch the same items repeatedly over transit connections, thereby increasing their traffic costs compared to no content-peering. Ideally, for a stationary arrival of interest messages the cache contents should stabilize in an equilibrium state that satisfies the ISPs' interest of traffic cost minimization. In the following we propose two distributed algorithms that avoid such inefficient updates and allow the system to reach an equilibrium allocation of items from which no ISP has an interest to deviate. For the proofs of all the theorems that follow, we refer to [13].

An equilibrium allocation of items $\mathcal{C}^* \in \times_{i \in N} \mathfrak{C}_i$ corresponds to a pure strategy Nash equilibrium of the strategic game $< N, (\mathfrak{C}_i)_{i \in N}, (C_i)_{i \in N} >$, in which each ISP i aims to minimize its own cost $C_i = \sum_{o \in \mathcal{O}} C_i^o$. The cache allocation $\mathcal{C}^*$ is a pure strategy Nash equilibrium if no single ISP can decrease its cost by deviating from it, that is

$$\forall i \in N, \forall \mathcal{C}_i \in \mathfrak{C}_i : C_i(\mathcal{C}_i^*, \mathcal{C}_{-i}^*) \leq C_i(\mathcal{C}_i, \mathcal{C}_{-i}^*) \tag{3}$$

A. Cache-or-Wait (CoW) Algorithm

Before we describe the Cache-or-Wait (CoW) algorithm, let us recall the notion of an independent set.

Definition 1. We call a set $\mathcal{I} \subseteq N$ an *independent set* of the peering graph $\mathcal{G}$ if it does not contain peering ISPs. Formally

$$\forall i, j \in \mathcal{I}, j \notin \mathcal{N}(i).$$

We denote by $\mathfrak{I}$ the set of all the independent sets of the peering graph $\mathcal{G}$. Consider a sequence of time slots t and a sequence of independent sets $\mathcal{I}_1, \mathcal{I}_2, \ldots \in \mathfrak{I}$ indexed by t, such that for every time slot $t \geq 1$ and every ISP $i \in N$ there is always a time slot $t' > t$ such that $i \in \mathcal{I}_{t'}$. At each time slot t we allow every ISP $i \in \mathcal{I}_t$ to update the set of its cached content $\mathcal{C}_i$. ISP $i \in \mathcal{I}_t$ can decide to insert in its cache the items that are requested by one or more of its local users during time slot t but were not cached at the beginning of the time slot. At the same time, ISPs $j \notin \mathcal{I}_t$ are not allowed to update the set of their cached contents. The pseudocode of the CoW algorithm for every time slot $t \geq 1$ is then the following:

- Pick $\mathcal{I}_t$.
- Allow ISPs $i \in \mathcal{I}_t$ to change their cached items from $\mathcal{C}_i(t-1)$ to $\mathcal{C}_i(t)$,
- For all $j \notin \mathcal{I}_t$, $\mathcal{C}_j(t) = \mathcal{C}_j(t-1)$.
- At the end of the time slot inform the ISPs $j \in \mathcal{N}(i)$ about the new cache contents $\mathcal{C}_i(t)$

Fig. 1. Pseudo-code of the Cache-or-Wait (CoW) Algorithm

What we are interested in is whether ISPs following the CoW algorithm would reach an equilibrium allocation from which none of them would like to deviate. If CoW reaches such an allocation, then it terminates, and no other cache update will take place. In the following we provide a sufficient condition for CoW to terminate in a finite number of steps. We call the condition *efficiency*, and the condition concerns the changes that an ISP can make to its cache configuration.

Definition 2. Consider the updated cache configuration $\mathcal{C}_i(t)$ of ISP $i \in \mathcal{I}_t$ immediately after time slot t. Define the evicted set as $E_i(t) = \mathcal{C}_i(t-1) \setminus \mathcal{C}_i(t)$ and the inserted set as $I_i(t) = \mathcal{C}_i(t) \setminus \mathcal{C}_i(t-1)$. $\mathcal{C}_i(t)$ is an *efficient update* if for any $o \in I_i(t)$ and any $p \in E_i(t)$

$$C_i^o(\mathcal{C}(t)) + C_i^p(\mathcal{C}(t)) < C_i^o(\mathcal{C}(t-1)) + C_i^p(\mathcal{C}(t-1)) \quad (4)$$

Given that the ISPs are profit maximizing entities, it is natural to restrict the changes in the cache configuration to changes that actually lead to lower cost. The *efficiency* condition is sufficient for CoW to converge, as stated by the following

Theorem 1. *If every ISP performs efficient updates then the function* $\Psi : \times_i(\mathfrak{C}_i) \to \mathbb{R}$ *defined as*

$$\Psi(\mathcal{C}) = \sum_{i \in N} C_i(\varnothing, \mathcal{C}_{-i}) - C_i(\mathcal{C}_i, \mathcal{C}_{-i}) \quad (5)$$

increases strictly upon every update and CoW *terminates in an equilibrium allocation after a finite number of updates.*

Thus, a network of ISPs in which only non-peering ISPs perform efficient updates simultaneously at every time slot reaches an equilibrium allocation after a finite number of updates.

B. Cache-no-Wait (CnW) Algorithm

A significant shortcoming of CoW is that in slot t it disallows ISPs $j \notin \mathcal{I}_t$ to perform an update. This restriction would provide little incentive for ISPs to adhere to the algorithm. In the following we therefore investigate what happens if every ISP in the system is allowed to perform an efficient update during every time slot. The pseudo-code of the CnW algorithm for time slot $t \geq 0$ looks as follows.

- Every ISP $i \in N$ is allowed to change its cached items from $\mathcal{C}_i(t-1)$ to $\mathcal{C}_i(t)$.
- At the end of the time slot ISP i informs the ISPs $j \in \mathcal{N}(i)$ about the new cache contents $\mathcal{C}_i(t)$

Fig. 2. Pseudo-code of the Cache-no-Wait (CnW) Algorithm

Theorem 2. *If every ISP performs only efficient updates,* CnW *terminates in an equilibrium allocation with probability 1.*

C. Numerical Results

In the following we show simulation results to illustrate the analytical results for CoW and CnW. Figures 3 and 4 show the average number of iterations and the average time the algorithms CoW and CnW need to terminate as a function of the time slot duration Δ, respectively. We report results for three different peering graphs. The CAIDA graph is based on the Internet AS-level peering topology in the CAIDA dataset [14]. The dataset contains 36878 ASes and 103485 transit and peering links between ASes as identified in [15]. The CAIDA graph is the largest connected component of peering ASes in the data set, and consists of 616 ISPs with measured average node degree of 9.66. The Erdős-Rényi (ER) and Barabási-Albert (BA) random graphs have the same number of vertexes and the same average node degree as the CAIDA graph. In ER graphs the degree distribution of vertices is binomial, while in BA graphs the degree distribution follows a power law. For the CoW algorithm, we used the Welsh-Powell algorithm to find a coloring [16] of the peering graph. We used $\alpha_i = 1$, $\gamma_i = 10$ and cache capacity $K_i = 10$ at every ISP. Each ISP receives interest messages for $|\mathcal{O}| = 3000$ items. The arrival intensities w_i^o follow Zipf's law with exponent 1, and for all $i \in N$ it holds $\sum_{o \in \mathcal{O}} w_i^o = 1$. Each data point in the figures is the average of the results obtained from 40 simulations.

Figure 3 shows that the number of iterations the CoW algorithm needs to reach an equilibrium allocation monotonically decreases with the time slot length. The longer the time slots, the more interest messages the ISPs receive within a time slot. This enables the ISPs to insert more highly popular objects per iteration. Furthermore, since only ISPs in an independent set can make updates at each iteration, simultaneous cache updates cannot occur. Consistently, the total time needed for the CoW algorithm to converge, shown in Figure 4, remains constant independent of the slot length Δ.

The CnW algorithm exhibits significantly different behavior for long time slots, as the number of iterations needed to terminate increases compared to the CoW algorithm. This happens

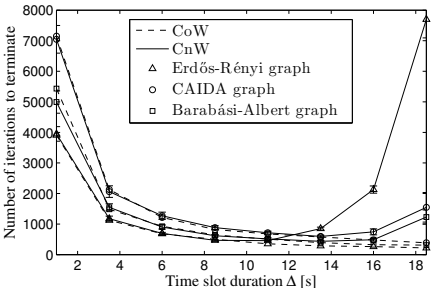

Fig. 3. Average number of iterations needed to reach an equilibrium allocation as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

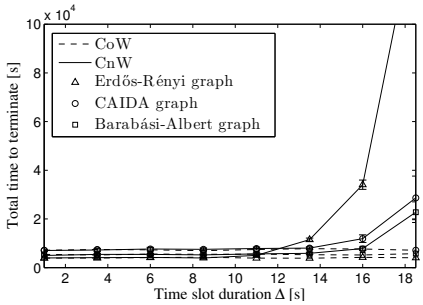

Fig. 4. Average time needed to terminate as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

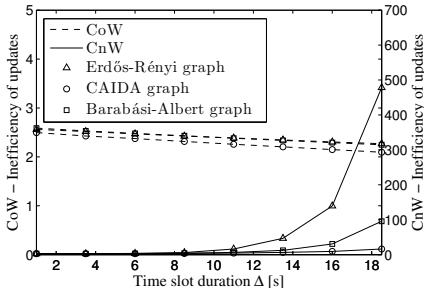

Fig. 5. Average inefficiency as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

because using the CnW algorithm a higher number of arrivals per time slot leads to a higher number of simultaneous updates, which disturb convergence. Figure 3 shows that simultaneous updates are most likely to occur in ER graphs. In BA graphs simultaneous updates would occur mainly among the few nodes with high degree, and since most ISPs have low node degree, the CnW algorithm would converge faster than on ER graphs. For the same reason, for small time slots when simultaneous updates are unlikely to occur, both the CoW and CnW algorithms perform best on the Erdős-Rényi random graph. From Figure 4 we notice that, as expected, the time for the CnW algorithm to terminate starts to increase with high values of the slot length. This increase is fast for the ER graph due to the higher occurrence of simultaneous updates, as we discussed above.

Figure 5 shows the number of items inserted in cache (potentially several times) for the two algorithms until termination divided by the minimum number of items needed to be inserted to reach the same equilibrium. We refer to this quantity as the *inefficiency of updates*. While the inefficiency of the CoW algorithm decreases slowly with the time slot length, that of the CnW algorithm shows a fast increase for high values of Δ, in particular for the ER and the BA graphs, which can be attributed to the simultaneous updates under CnW. These results show that although CnW would be more appealing as it allows ISPs to update their cache contents all the time, CoW terminates significantly faster and is more efficient.

D. The Case of Imperfect Information

Under imperfect information the estimation of the item popularities is imperfect, thus the system can not settle in any single equilibrium or stable allocation, unlike in the case of perfect information. Nevertheless, the cache allocations that are most likely to occur are not arbitrary. In [13] we model the evolution of the cache allocation under imperfect information by a regular perturbed Markov process, and we show that the stochastically stable states under imperfect information are a subset of the equilibrium allocations under perfect information.

IV. Conclusion

We proposed a model of the interactions between the caches managed by peering ASes in a content-centric network. We used the model to investigate whether peering ASes need to coordinate in order to achieve stable and efficient cache allocations in the case of content-level peering. We showed that irrespective of whether the ISPs coordinate, the cache allocations of the ISPs engaged in content-level peering will reach a stable state. If fast convergence to a stable allocation is important too then synchronization is needed to avoid simultaneous cache evictions by peering ISPs. Interesting open questions are how peer-specific costs influence the stability of cache allocations and the convergence properties of the system, as well as the design of pricing schemes to incentivize peering ISPs to synchronize.

References

[1] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proc. of ACM SIGCOMM*, vol. 37, no. 4, 2007, pp. 181–192.

[2] C. Dannewitz, "NetInf: An Information-Centric Design for the Future Internet," in *Proc. of GI/TG KuVS Work. on The Future Internet*, 2009.

[3] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. of ACM CoNEXT*, 2009.

[4] D. Rossi and G. Rossini, "On sizing CCN content stores by exploiting topological information," in *Proc. of IEEE Infocom, NOMEN Workshop*, 2012, pp. 280–285.

[5] E. J. Rosensweig and J. Kurose, "Breadcrumbs: Efficient, Best-Effort Content Location in Cache Networks," in *Proc. of IEEE Infocom*, 2009, pp. 2631–2635.

[6] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks," in *ICN workshop*, 2012, pp. 1–6.

[7] P. Faratin, D. Clark, P. Gilmore, S. Bauer, A. Berger, and W. Lehr, "Complexity of Internet Interconnections : Technology , Incentives and Implications for Policy," in *Proc. of Telecommunications Policy Research Conference*, 2007, pp. 1–31.

[8] E. J. Rosensweig, J. Kurose, and D. Towsley, "Approximate Models for General Cache Networks," in *Proc. of IEEE INFOCOM*, 2010, pp. 1–9.

[9] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance," in *Proc. of the 24th International Teletraffic Congress*, 2012.

[10] E. G. Coffman Jr. and P. J. Denning, *Operating Systems Theory*, 1973.

[11] N. Megiddo and D. Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache," in *Proc. of FAST*, 2003, pp. 115 – 130.

[12] J. G. Skellam, "The frequency distribution of the difference between two Poisson variates belonging to different populations." *Journal Of The Royal Statistical Society*, vol. 109, no. 3, p. 296, 1946.

[13] V. Pacifici and G. Dán, "Content-peering Dynamics of Autonomous Caches in a Content-centric Network," in *Proc. of IEEE Infocom*, 2013.

[14] "CAIDA. Automated Autonomous System (AS) ranking." [Online]. Available: http://as-rank.caida.org/data/

[15] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, K. Claffy, and G. Riley, "AS relationships: inference and validation," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 29–40, 2007.

[16] D. J. A. Welsh and M. B. Powell, "An upper bound for the chromatic number of a graph and its application to timetabling problems," *The Computer Journal*, vol. 10, no. 1, pp. 85–86, 1967.