KTH Electrical Engineering

Bandwidth and Storage Allocation for Operator-owned Content Management Systems

VALENTINO PACIFICI

Licentiate Thesis
Stockholm, Sweden 2014

TRITA-EE 2014:015
ISSN 1653-5146
ISBN 978-91-7595-088-4

KTH School of Electrical Engineering
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av licentiatexamen torsdagen den 24 april 2014 klockan 9.00 i sal Q2, KTH, Stockholm.

Tryck: Universitetsservice US AB

Abstract

The demand for Internet-based visual content delivery has increased significantly in recent years, triggered mainly by the widespread use of Internet enabled smartphones and portable devices, and by the availability of super HD content. As a consequence, live and on-demand video content has become the most important source of network traffic in mobile and fixed networks alike. In order to be able to efficiently deliver the increasing amount of video traffic, network operators have started to deploy caches and operator-owned CDNs. These solutions do not only reduce the amount of transit traffic of the operators but they may also improve the customers' quality of experience, through bringing the video content closer to customers. Nevertheless, their efficiency is determined by the algorithms and protocols used to allocate resources, both in terms of storage and bandwidth. The work in this thesis addresses the allocation of these two resources for operator-owned content management systems.

In the first part of the thesis we consider a cache maintained by a single network operator. We investigate how caching at a network operator affects the content distribution system as a whole, and consequently, the efficiency of content delivery. We propose a model of the decision process undertaken by a network operator that aims at optimizing the efficiency of a cache by actively managing its bandwidth. We design different algorithms that aim at approximating the optimal cache bandwidth allocation and we evaluate them through extensive simulations and experiments. We show that active cache bandwidth allocation can significantly increase traffic savings.

We then consider the potential interaction among caches maintained by different network operators. We consider the problem of selfish replication on a graph as a model of network operators that individually deploy replication systems, and try to leverage their peering agreements so as to minimize the traffic through their transit providers. We use game-theoretical tools to investigate the existence of stable and efficient allocations of content at the network operators. We show that selfish myopic updates of content allocations at different network operators lead the system to a stable state, and that the convergence speed depends on the underlying network topology. In addition, we show that interacting operator-owned caches can reach a stable content allocation without coordination, but coordination leads to more cost efficient content allocations.

Acknowledgments

I would like to thank my advisor György Dán for his guidance and all the fruitful discussions that provided me with fundamental insights into the problems I was going to address. I sincerely appreciate his patience as well as his dedication to push me forward, even when things did not turn out as expected. I would also like to thank professor Gunnar Karlsson for giving me the opportunity to become a member of LCN. Furthermore, I am happy to thank the members of LCN who maintain a joyful environment in the lab and break the monotony of everyday work.

I am thankful to all my friends, in Stockholm and abroad, who have encouraged, entertained and supported me through these years. In particular, I am very grateful to Selene who remembers me how it feels to look at the world from a non-engineering perspective. Last but not least, I would like to express my greatest gratitude to my loved ones; this journey would not have been possible without the constant support of my family.

Contents

Chapter 1

Introduction

1.1 Background

In recent years, the usage patterns of the Internet have increasingly shifted towards content generation, distribution and sharing. Real-time and video-on-demand (VoD) are widely consumed by Internet users and video content has become one of the most important sources of network traffic. In 2013, Netflix [1] was the leading downstream application in North America and together with YouTube accounted for over 50 percent of downstream traffic on fixed networks [2]. Cisco predicted a three-fold increase in IP traffic by 2017 [3], more than 80 percent of which is expected to be video traffic.

Due to the increasing demand for content, efficient content management is crucial to improve the performance and to reduce the cost of content delivery. Content providers are continuously deploying new infrastructure to face the growth of user demand for content. At the same time, content providers outsource content delivery to commercial content management systems, such as content delivery networks (CDNs). CDNs will deliver almost two-thirds of all video traffic by 2017 [3]. In 2011, the revenue of the CDN market accounted for $ 2 billion and it is predicted to reach $ 6 billion by 2015 [4].

1.2 Challenges

While the business of commercial content delivery thrives, network operators must face the increasing growth of Internet traffic without being part of the revenue-distribution mechanism of the content delivery market. Increased demand for traffic puts stress on their infrastructure and affects the quality of experience (QoE) of their subscribers. Furthermore, dynamic content workload and CDN content placement and redirection policies introduce new challenges for network operators trying to optimize their networks.

In response to these challenges, many service providers have deployed internal content management systems [5]–[8], with the objective of controlling and optimizing the delivery of content in their own network. By serving content from within their networks, operators aim at reducing their cost for traffic and increase the quality of experience of their subscribers.

Network operators deploy internal content management systems either by setting up their own infrastructure or by interacting with a CDN provider [86]. In the latter approach, the CDN provider can deploy its own servers and operate the CDN on behalf of the network operator; this approach is referred to as *managed CDN* [87]. Alternatively, the CDN provider licenses its software to the network operator, which is in charge of deploying and operating the CDN itself; this approach is called *licensed CDN* [88], [89].

Network operators designing and deploying their content management systems need to solve two problems. First, they need to characterize the workloads generated by their subscribers, to efficiently dimension and organize their storage sites. Second, they have to solve problems of content placement for optimal content delivery.

In this thesis we first consider a content management system deployed by a single network operator. We investigate how content delivery at a single network operator affects the content distribution system as a whole, in terms of costs and efficiency. We then consider the potential interaction among content management systems maintained by different network operators. We investigate the effects of interaction and coordination among network operators that leverage their settlement-free agreements.

1.3 Thesis Structure

The structure of this thesis is as follows. In Chapter 2, we introduce the performance metrics that network operators use to evaluate their content management systems and we describe different aspects of the problem of resource allocation. In Chapter 3, we investigate the problem of resource allocation in a stand alone content management system deployed by a single network operator. In Chapter 4, we extend our analysis by considering a network of content management systems deployed by multiple autonomous network operators and we focus on the effects of the interaction among peering network operators. Chapter 5 provides a summary of the papers collected in this thesis along with the thesis' author contributions to each paper. Chapter 6 concludes the thesis by summarizing the main findings and discussing potential directions for future research.

Chapter 2

Resource Allocation in Content Management Systems

Available bandwidth, storage capacity, content placement and routing of user requests in a content management system are arguably the key aspects of the resource allocation problem that affect cost and efficiency of content delivery. The storage capacity and the placement strategies of content items affect the share of user requests that are served within the operator's network. The routing of user requests and the bandwidth allocated to the content management system influence the QoE experienced by the subscribers. In order to quantify the effects of these aspects on the efficiency of content delivery, network operators can use various performance metrics.

2.1 Performance Metrics

A network operator aims at maximizing profit while providing a reasonable level of QoE to its subscribers. In the following we discuss two performance metrics that are useful to assess the performance of an operator-managed content management systems.

Cost

When deploying and operating a content management system, a network operator incurs capital (CapEx) and operational (OpEx) expenditures, and it is in its interest to efficiently dimension and utilize its infrastructure.

The storage and the bandwidth allocated to the content management system influence the costs of the network operator directly, upon provisioning, and indirectly, through affecting performance. The bandwidth at which content is served affects the subscribers' experience and in turn the future profit of the operator. The amount of storage allocated to the content management system determines

the share of content that can be served within the operator's network, and thereby affects the amount of inter-operator traffic generated by content retrieval. The cost of inter-operator traffic may vary depending on the business agreements stipulated with the neighboring network operators.

In today's Internet, tier-2 and 3 network operators connect to other networks in order to deliver their subscribers' traffic to destinations outside their footprint. Tier-2 and 3 network operators establish client-provider business relations (transit agreements) with one or more higher tier operators, in order to ensure global connectivity. In addition, network operators typically maintain peering agreements with adjacent autonomous systems, with the purpose of mutually reducing their transit traffic. The traffic exchanged between peering operators is usually not charged, unlike the traffic that low-tier network operators exchange with their transit providers. The cost for transit accounts for a share of the OpEX of tier-2 and 3 network operators that is difficult to quantify, since network operators usually do not publish the details of their operational costs. Nevertheless, it is in the interest of tier-2 and 3 network operators to minimize traffic through their transit providers, by leveraging their settlement-free agreements. This can be done through efficient algorithms of user-request routing and through agreements for cooperation between peering operators.

Quality of Experience

Over-the-top (OTT) content providers such as Netflix[1], Hulu [9], and Amazon [10], try to maintain customer satisfaction through increasing the quality of the offered content. 3D content has become commonplace, and super HD content has become available recently [1]. Higher quality of content results in increased bit-rates which in turn stress network operators' networks. Since for the network operators it is crucial to maintain an acceptable QoE for their subscribers, they need to find efficient solutions to the problem of increased traffic demands. Two main factors that influence the QoE of operators' subscribers are the quality of the delivered content and the startup delay at which the content is served. Bit-rate plays a central role in the delivery of video content, while latency is crucial for the delivery of small sized content chunks, as for the case of Web traffic. Several OTT content providers have highlighted the critical effects of perceived latency on user satisfaction and business metrics [11], [12]. Therefore, when deploying and operating their own content management systems, network operators control the delivery of content in their own networks and attempt to serve the most significant share of their content with low latency and high bit-rate.

2.2 The Resource Allocation Problem

When designing and deploying their content management systems, network operators need to allocate their resources intelligently, to provide satisfactory perfor-

mance at reasonable costs. The problem of resource allocation in a content management system has multiple interdependent aspects. We briefly describe them in the following.

Content Placement

Network operators can optimize content delivery by periodically updating the content allocated to their content management systems. They typically collect statistics of user accesses to content, with the purpose of estimating the expected future popularity of each content item among their subscribers [13], [14]. The information on the expected popularity of content items can then be used to optimize the placement of content in the storage of the content management system.

We distinguish between two processes of content placement in content management systems.

Caching: A *caching decision* is taken upon the retrieval of a content item requested by a subscriber but not available in the storage of the content management system. The network operator decides whether to store the content item, partially or in its entirety. The decision is taken according to the *cache admission policy* implemented in the system. If the decision is to store the content, the network operator has to decide also on the content that should be evicted to make room for the new item. The *cache eviction policy* serves this purpose.

Replication: When performing *pre-fetching* or *replication*, a network operator computes the set of content items to store in its content management system. The computation is usually done by solving an optimization problem based on the expected future popularity of the content among the subscribers and using the performance metrics described in Section 2.1. The content items in the optimal set are then retrieved from their origin and are stored in the content management system. Upon changes of expected content popularity or expected costs and latencies, the network operator recomputes the optimal set of content items and updates the set of allocated content by actively fetching new content items.

The complexity of the optimization problem and the size of the content items to be retrieved pose constraints on the amount of pre-fetching operations that can be performed in a give time span. For this reason, the time period between two subsequent pre-fetching operations is usually significantly longer compared to the time period between subsequent caching decisions.

Bandwidth Allocation

It is commonplace for network operators to actively manage the bandwidth allocated to various Internet services used by their subscribers. By using various techniques of traffic inspection [15]–[17], network operators can categorize user traffic and distinguish among traffic flows generated by different Internet services. Through assigning priorities to these flows, network operators can guarantee higher

bandwidth to profitable applications, like the delivery of pay-per-view content, providing better QoE to their premium customers.

Analogously to what they do with user traffic, network operators can actively allocate the bandwidth of their content management systems. By influencing the rate at which each single content item is served to the subscribers, network operators act on the subscribers' QoE and thereby influence the permanence and the future access patterns of users in the system. In some systems, the effects of these changes on the costs incurred by network operators can be relevant [18], [19]. Therefore, when optimizing the allocation of bandwidth to content items, network operators should take into account its direct and indirect effects on the performance metrics.

Storage Capacity Dimensioning

Network operators can decide upon the storage capacity allocated to their content management systems. Storage capacity dimensioning can be performed in the system design phase or dynamically, in response to shifting subscribers' demands or performance goals. When designing their content management systems, network operators can use statistics on the expected utilization of their system to solve the trade-off of storage cost versus returns in terms of performance metrics. The internal structure of the content management system often plays a central role in assessing the optimal storage capacity allocation. When dealing with multiple replica sites for example, network operators could opt for a multi-tier architecture. In such systems, the optimization of storage size would need to take into account the different performance requirements and costs for each tier. The optimal content allocation to replica sites would then depend on the storage capacity at each replica site. Given the tight coupling, there might be significant benefits of solving the problem of storage capacity allocation and cache allocation jointly.

Chapter 3

Stand-alone Content Management Systems

A network operator designing, deploying and operating its own content management system implements algorithms and protocols with the objective of optimizing content delivery in its own network. While aiming at maximizing its profits, the operator aims at decreasing its costs and maximizing the QoE of its subscribers. In the following we focus on the optimization performed by a single network operator, and neglect the potential effects of the operator's actions on the content delivery ongoing at neighboring network operators.

3.1 Content Placement

As discussed in Chapter 2, the set of content items stored in the content management system affects the efficiency of content delivery by influencing the amount of transit traffic and the user-perceived latency. A network operator aims at improving the performance of content delivery by storing the most popular content items close to its subscribers. It is known from results from disk and memory caching [20], [21] that optimal allocation decisions should be made so as to evict content that will not be accessed furthest ahead in the future. In practice however, the future access pattern is not available, and its estimation is complicated by the high variability of user access trends. In addition, content items may have different sizes and variable retrieval costs (i.e. miss penalties), which makes the problem of content allocation fundamentally different from traditional caching.

If network operators lack of an accurate prediction of future content popularity, they have to update the content allocation continuously, upon requests of subscribers for content items. There is a wide range of work that focuses on the design and evaluation of cache eviction policies [22], [23]. The simplest policies proposed in the literature try to capture and exploit the temporal locality of reference in

user-request streams. The Least Frequently Used (LFU) and Least Recently Used (LRU) eviction policies exploit two different forms of temporal locality, which are, respectively, locality due to popularity and locality due to recency of references. LFU keeps track of the frequency of user accesses to content items and discards the least often used item, while LRU keeps track of the recency of user accesses and discards the least recently used item.

LFU is optimal under a independent reference model [20], but it is oblivious to recency of access. Furthermore, its running time per request is logarithmic in the cache size. LRU is an approximation of the optimal replacement decision whenever the recent history of user accesses is an approximation of the future. LRU is simple to implement, and appears to perform considerably well, and thus it is widely used in Web servers, client applications and proxy servers [24]. Nevertheless, both LFU and LRU systems base their eviction on one single form of temporal locality of reference. In order to capture both recency and frequency of references, some hybrid policies like LRFU [25], LFU with Dinamic Aging (LFU-DA) [26], and ARC [27] have been proposed instead. Experiments showed that LFU-DA performs better than most existing algorithms in terms of hit rate [26].

The eviction policies discussed above base their decisions exclusively on temporal locality of reference, thereby not taking into account the retrieval cost of content items. However, in most systems, the cost for content retrieval varies among different items, depending of a variety of factors that include heterogeneous content size and traffic costs. The GreedyDual-Size (GDS) [28] eviction policy evicts content items with the smallest key value for a certain cost function. The key value is proportional to the cost of the content item which, in turn, depends on the goal the algorithm wants to achieve. GDS can be set to minimize the average latency or the overall traffic, and it has been shown to outperform most of the other replacement policies. One shortcoming of GDS is that it does not take into account how many times the content item has been accessed in the past. GDSF [26] and GDSP [29] are extension to GDS that consider also locality due to popularity.

Network operators may be capable of accurately predicting the future popularity of content items [13], [27], [30], [31]. Such accurate prediction would allow operators to make effective pre-fetching decisions and to update the storage allocation of their content management systems when their network infrastructure is not highly utilized. Pre-fetching can be done at different time scales depending on resource constraints. A large time period between consecutive pre-fetching operations would not allow tracking the variability of user access trends and thus to leverage the short term fluctuations in the content popularity. In [32] the authors investigate whether pre-fetching can be dynamically adapted to changing predicted content popularities. They consider hybrid content distribution systems that combine cloud and CDN based infrastructure, and show that dynamic allocation can provide significant gains compared to static allocation.

In some cases, network operators distribute the storage of their content man-

agement systems over multiple sites. A distributed storage system can push content closer to the network edge, and thereby improve the performance of content delivery. The placement of part of the storage sites close to the subscribers enables a better exploitation of locality. Multiple storage sites distributed across the subscriber population allow to leverage the variation of content popularity across heterogeneous subscribers communities. Furthermore, content placed closer to the subscribers can be accessed at lower latency and can reduce in-network traffic. The problem of determining the location of the storage sites in the network to maximize performance has been explored in [33], [34].

When trying to leverage the positive aspects of distributed content delivery, the placement of content items at different storage sites plays a central role. Network operators need intelligent content placement strategies that maximize the overall performance of their content management systems. Optimal content placement for arbitrary network structures has been investigated in [35], [36]. In large systems however, the calculation of the optimal content allocation can be computationally prohibitive. In such cases, network operators need to implement simple distributed algorithms that lead the system to a reasonable approximation of the optimal solution [35]–[37].

In the process of distributed content placement, the structure of the network connecting the storage sites may play a central role. Efficient content placement strategies may arise for specific network topologies. The problem of content placement in hierarchical networks has been explored in [38]–[40]. In particular,[38] showed that the optimal content allocation in hierarchical networks has a regular structure and provides better performance than in general network topologies.

3.2 Bandwidth Allocation

Bandwidth management is one of the tools that network operators can use to face increasing rate of traffic growth that, in turn, affects the cost of traffic and threatens the QoE of subscribers. By actively engineering the user traffic, network operators can guarantee throughput for the most important services or meet the QoE requirement of VIP subscribers, even in case of high network stress. Traffic engineering has been shown to provide significant benefits to network operators [41], and commercial solutions that allow categorization of traffic using deep packet inspection (DPI) are available [15]–[17].

However, the rapid and continuous emergence of new Internet applications and services constitutes a challenge for network operators trying to categorize user traffic. Furthermore, penalizing the most bandwidth and latency demanding applications like video-on-demand (VoD), peer-to-peer (P2P) and voice-over-IP (VOIP), degrades the QoE of the subscribers using these services that could potentially leave the network operator [42]. Despite this, today it is common among network operators to block or throttle VOIP and P2P traffic [43]. As an alternative approach, some network operators have deployed P2P caches to address the problem of in-

creased traffic demands from P2P [44], [45]. P2P caches decrease the amount of inter-ISP traffic by storing the most popular content in the operator's own network, saving the cost of content retrieval from external networks.

While bandwidth management of user traffic has been extensively investigated [41], [46], [47], the impact of the cache bandwidth and its management has received little attention in the literature. This is partially due to the fact that bandwidth management is not necessary in Web caching, as the amount of data served from the cache equals the traffic savings achieved by caching. Nonetheless, bandwidth management affects the subscribers' QoE and in turn influences their churn in the system. While this effect is negligible for Web caching, in the case of P2P-like systems bandwidth allocation affects the characteristics of the overlay, and the short and long term impact on the traffic cost may be relevant [18], [19].

Therefore, one important question is whether a network operator should allocate cache bandwidth among competing P2P overlays. Several works propose schemes for bandwidth allocation among multiple P2P overlays [48]–[50], but their purpose is to maximize the total download rate of the system in an attempt to minimize the download latencies perceived by the peers. The question of whether cache bandwidth allocation could be used by network operators to minimize their traffic costs remains open.

In this thesis we contribute to answer this question. In paper A, we model the decision of several network operators that perform active cache bandwidth allocation. We prove the existence of an optimal decision policy that only depends on the current state of the system. We propose different P2P cache bandwidth allocation policies that approximate the optimal policy and we demonstrate the importance of capturing the impact of the cache on the characteristics of the P2P overlays. By performing simulations and experiments, we show that P2P cache bandwidth should be actively managed so to achieve significant gains in terms of traffic cost, and we identify some of the primary factors that influence the gains.

3.3 Storage Capacity Dimensioning

Storage capacity directly influences the performance of a content management system. By increasing the storage capacity of their content management systems, network operators become able to serve a bigger share of the content requested by their subscribers from their internal networks, thereby reducing transit traffic and improving QoE. Nevertheless, storage capacity is limited by the costs for its purchasing, deployment and maintenance.

Storage capacity allocation in networked systems has received limited attention in the literature. Most of the work related to cache systems focuses on cache eviction policies, cache placement and routing of requests. The reason can be mostly attributed to the combination of wide availability of cheap storage and limited size of web objects that made the problem of cache dimensioning less relevant in practice, and assumptions such as unlimited caching storage became common.

Recent changes in the characteristics of content traffic prompted for a new analysis of the problem of storage capacity allocation. VoD systems have become commonplace [1] and caching of P2P content is already performed by network operators [44], [45]. The use of multiple replica sites close to the users to efficiently deliver content is now the norm [51]–[54]. Video content and P2P shared content can easily exhaust the capacity of a content management system, even under the current prices for storage. Therefore, it is important to characterize the necessary amount of storage in the content management system to meet constraints of QoE. This problem has been investigated in [55]–[57] where the authors solve the storage capacity allocation problem to meet some requirements on the delay perceived by the users. The performance benefits of jointly optimizing content placement and the storage capacity allocation have been investigated in [58].

Chapter 4

Network of Operator-owned Content Management Systems

Recent industry efforts aim at interconnecting the content management systems deployed by different network operators. Such interconnection could benefit network operators empowering them to leverage their peering links in order to decrease their transit traffic and the average latency perceived by the users. The success of networks of operator-owned content management systems depends on the one hand on developing appropriate interfaces and signaling systems, and on the other hand on the design of algorithms and protocols that perform well with respect to the performance metrics discussed in Section 2.1.

4.1 Cooperative Caching among Network Operators

In this section we consider a number of peering network operators that collaborate by serving each others' subscribers, in an attempt of capitalizing their settlement-free peering agreements. By doing so, each network operator tries to leverage the content stored in the caches of its peering operators, thereby reducing its transit traffic.

In the context of Web caching, several signaling protocols for inter-cache cooperation have been proposed [59]–[61]. The Internet Cache Protocol (ICP) [59] supports discovery and delivery of content stored in nearby caches. In [60], [61] instead, each cache maintains a potentially inaccurate summary of the content available at its neighbors.

In addition to the design of discovery or dissemination schemes, cooperative caching has to be analyzed to assess the gains in terms of efficiency of content delivery. Cooperative web caching has received a lot of research attention, and it has been shown that the gain from cooperation is only marginal and achieved under certain conditions [62], [63]. In particular, the authors of [63] show that the factors that mostly limit the gain from cooperation are the high dynamism of web

objects and the high overhead compared to the small object size. Furthermore, the ubiquitous local caches at each client significantly reduce the hit rates in cooperative web caching.

Even though none of the above limiting factors applies to P2P or Video caches, the case of cooperative caching in P2P and VoD systems has received little attention in the literature. Cooperative P2P caching was shown to lead to considerable improvements of the cache efficiency at the cost of a negligible overhead [64]–[66]. In particular, [66] shows that the improvements are significant even if the network operators deploying the P2P caches follow their selfish interests.

Distributed Selfish Replication

In lack of a central authority that would enforce the optimal content allocation, selfish nodes would not implement the optimal solution if they can individually fare better from deviating from it. As we discussed in Chapter 3, network operators, like selfish nodes in a replication group, would update their allocation of content to optimize their own performance metrics. In the case of interconnected content management systems, network operators would leverage their peering links by accessing the content allocated at their peers, so as to decrease their transit traffic and the average latency perceived by the users. Additionally, they would tailor their content allocation in response to the decisions on content allocations made by their peers. One important question in this case is whether there exist a stable allocation of content items to network operators. A similarly important question is whether the network operators would reach such stable allocation, if they perform selfish updates that optimize their content allocation with respect to their own performance metrics.

Several works have tried to answer these questions. The existence of a stable allocation was shown in the case of unit storage capacity and infinite number of content items [67], in the case of a complete peering graph and homogeneous payoff functions [68], [69], and in the case nodes can replicate a fraction of content items [70]. [71], [72] consider the case when the payoff functions are linear; [71] shows that cycles in sequential updates can exist in the case of directed peering graph.

It is not clear whether network operators would reach a stable allocation in the most general case of selfish distributed replication with node-specific, non-linear, payoff functions on an arbitrary peering graph topology. In Paper B we show that stable allocations of content items to network operators always exist and we give a bound on the complexity on calculating a stable allocation. We show that for non-complete peering graphs, network operators might cycle if they perform selfish updates, but they would reach a stable content allocation if the updates are done in random order. Finally, we consider a cost model in which peering network operators always have incentives to allocate disjoint sets of content items. We show that in such a model there are no cycles, even if the updates are performed simultaneously by network operators that belong to an independent set of the peering graph.

4.2 Autonomous Cache Networks in CCNs

In order to overcome the structural limitations of the host-to-host paradigm of the current Internet, several works have proposed a clean-slate design of the Internet architecture, where content delivery is the key function. Content-centric networking is a network model oriented to content, in which content items are addressable and the basic form of interaction is host-to-content [73]–[75].

In a content-centric network (CCN) caches are part of the protocol stack. Content items are stored in caches spread throughout the network, and routing is content-aware. When a user asks for a content item in a CCN, the request is routed through a set of caches in the network. If the content is found in one of these caches, then it is retrieved and served, otherwise the request reaches one of the content custodians from where it is ultimately served. Content custodians store content items permanently; they are content producers or servers.

The advantages of such an architecture over a host-to-host paradigm are clear. Due to the availability of storage sites throughout the network, the most popular content can be efficiently pushed close to the network's edge. Furthermore, content is intrinsically redundant in the network, and it can be placed to keep the overall network traffic minimal. In addition, caching of content items is not performed at the application layer like in today's Internet, but at the network layer. This approach drastically benefits the speed of the cache lookups and the content retrieval in case of cache hit. Finally, security is built into the network at the network layer.

Nevertheless, content-centric networking comes with a wide range of challenges that need to be addressed. As the design of CCNs lies outside the scope of this thesis, we limit ourselves to a brief overview of the most interesting problems and point to the related research work.

As the users of a CCN are oblivious to the content location, routing of user requests is performed by content name. Therefore, the naming mechanism adopted in a CCN influences the design of the routing scheme and thereby the efficiency of content delivery. In order to avoid excessive growth of routing tables, content naming should be scalable and allow aggregation. Furthermore, content naming should be persistent, as user requests for the same name at different points in time are expected to retrieve the same content item. As the location of content items is not unique nor fixed and the ownership of content might change, it is not trivial to provide naming persistence and aggregation capabilities. Several research works have addressed the problems of content naming [73], [74] and name-based routing [76], [77]. The design of intelligent content placement schemes for CCNs is also challenging. The mechanism of automatic caching supported in CCNs [73] inherently presents problems of caching redundancy. Moreover, due to the mesh topology of caches in a CCN, existing caching eviction policies are not guaranteed to achieve good performance. The design and evaluation of efficient caching algorithm for CCNs has been addressed in [78].

Autonomous networks of caches in a CCN

Similar to the structure of today's Internet, a future content-centric network is likely to be a network of autonomous operators, each of them maintaining and managing the infrastructure within its network. Each network operator's primary concerns would be its profit and the QoE of its subscribers. Therefore, the routing and the content allocation within each network would be optimized for local performance. Each network operator would have incentives to design the routing mechanism for its subscribers' requests to reach primarily the content replicas cached within its network. In addition, instead of forwarding the requests that cannot be served locally to a transit provider, the operator would try to leverage the content cached within its peering networks. Therefore, in a cache network of autonomous operators, each operator would likely implement cache eviction policies that attempt to minimize its traffic costs. Each caching decision at a network operator would then depend on the content cached at its peering networks.

An open question is whether the interaction between autonomous peering networks of caches in a CCN would lead to unforeseen instability and oscillation of content allocations. In Paper C we model the interaction and the coordination between caches managed by peering network operators. We prove that network operators do not need to coordinate in order to reach a stable allocation of content items to caches. Furthermore, we show that, in order to achieve fast convergence to a stable allocation, network operators establishing peering agreements should not simultaneously update the set of cached content items.

4.3 Interconnected Content Distribution Networks

As OTT content providers believe that the key to maintaining customer loyalty and increasing sales is to provide better QoE, the traffic generated by digital video content delivery is anticipated to show tremendous growth over the coming years. To face the new challenges brought by raising demands for digital content and in order to maintain a competitive QoE for their users, it is often more convenient for the content providers to outsource content delivery to commercial CDNs.

Through multiple replicas and efficient routing of users to replicas [79], commercial CDNs provide better performance than a system based on a single delivery server [80], [81]. In addition, they provide dynamically scaling bandwidth to cope with the variability of customer demands. Overall, they offer a relatively low delivery cost compared to the costs incurred by the content providers that deploy their own infrastructure.

Outsourcing content delivery to a single CDN does not usually provide content providers with enough footprint to serve all their customers. Therefore, users that are not included in the CDN footprint may experience a degraded QoE. As a result, content providers stipulate agreements with multiple CDNs, in an attempt to combine their footprints and thereby reach all of their customers with a guaranteed QoE.

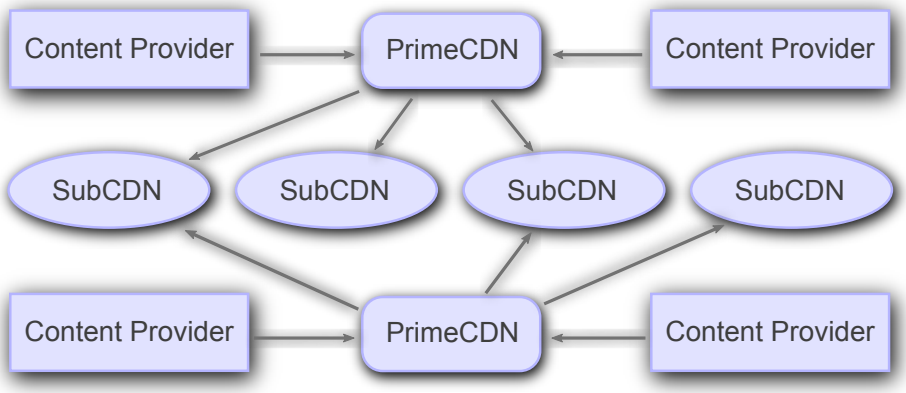

Figure 4.1: A schematics showing an example of business agreements among content providers, primeCDNs and subCDNs. The arrows show the direction of the money flows.

However, in the process of choosing multiple content delivery networks, content providers have to face the complexity of negotiating business and technical arrangements with multiple parties. Once establishing the deals, content providers need to deploy ad-hoc systems to perform analytic, reporting and control over the delivery of content to the users in the footprints of the various CDNs.

For this reason, several content providers together with few commercial CDNs have expressed the need of mechanisms for CDN brokerage. A *CDN broker* [82] or *primeCDN* [4], [83] is an intermediary between the content provider and multiple commercial CDNs. A CDN broker would be in charge of aggregating the offers of several *sub-CDNs*, i.e. wholesale content delivery networks not involved in the negotiations with the content providers. The function of the CDN broker would be to select the sub-CDNs so as to provide cost-effective delivery and optimal geographical coverage of the content providers' customers. The CDN broker would resell the offers of the sub-CDNs as a packaged service to the content providers, thereby offering hosting and delivery services, together with analytical tools to monitor the process of content delivery. In Figure 4.3 we provide an example of business agreements between content providers, primeCDNs and sub-CDNs.

As today's commercial CDNs are closed systems, with proprietary interfaces and protocols, adapting the different interfaces to provide content delivery under the same framework could be a daunting task for a content provider. Therefore, outsourcing content delivery to many independently operated CDNs may be prohibitive, even in presence of an aggregating service such a CDN broker.

CDN interconnection would represent a solution to the shortcomings described above. In addition, it would bring several advantages to other players in the content

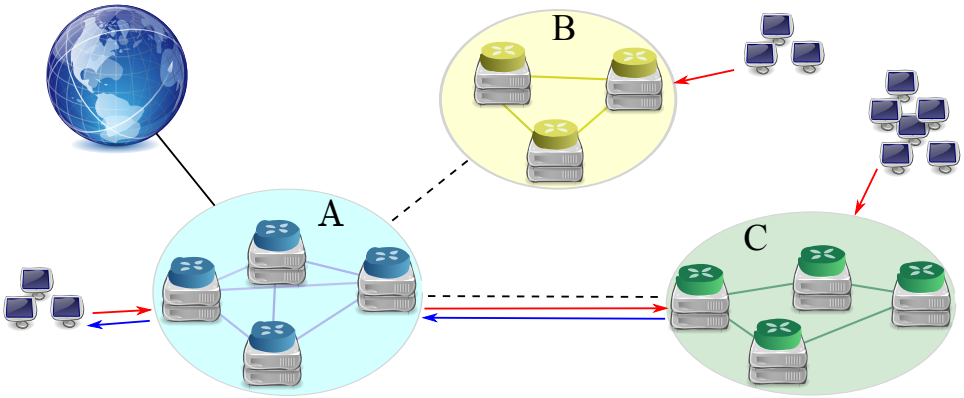

Figure 4.2: An example of routing of requests in a network of operator-deployed CDNs. The dashed lines are peering links while the continuous line shows a transit link. In the figure, the request from one of the subscribers of operator A is served from the CDN owned by operator C, through a peering link.

delivery market. Users would be able to access a wider range of content with CDN level performance and CDNs would benefit from an extended footprint and improve their resilience to flash crowd effects.

The success of CDN interconnection depends, on the one hand, on developing appropriate interfaces and protocols, and on the other hand on investigating the effects of such interconnection on content placement and request routing.
The problem of designing interfaces and mechanisms for CDN interconnection, such as content meta-data exchange, logging information exchange and request routing, have received significant attention [84]. A standard for CDN interconnection is under development in the IETF CDNi working group [85].

Interconnected Operator-owned CDNs

Peering network operators deploying and operating their *network CDNs* (NCDNs) (often referred to as *Telco CDNs*, or *Carrier CDNs*) would likely connect their CDNs through the interfaces and protocols designed in [84], [85], in an attempt to leverage their settlement-free agreements. Each network operator would then route part of the content requests from its subscribers through its peering links in an attempt to serve them through its peering CDNs. An example of routing of request in a network of operator-deployed CDNs is shown in Figure 4.2.

As network operators would be able to access the content items allocated at the CDNs of their peering network operators, they would optimize their content allocation depending on the content that they can access from their peers. Therefore, one important aspect that could determine the success of autonomous interconnected CDNs is the convergence of the system to an efficient and stable global content

allocation of items to the various CDNs. The questions of whether such stable allocation exists or whether the optimizations of every autonomous network operator would lead the systems to such global content allocation are still open.

Chapter 5

Summary of original work

Paper A: Cache Bandwidth Allocation for P2P File Sharing Systems to Minimize Inter-ISP Traffic

Valentino Pacifici and Frank Lehrieder and György Dán

submitted to IEEE/ACM Transacions on Networking (TON), second revision.

The original version of the paper appeared in Proc. of IEEE INFOCOM, 2012.

Summary: In this paper we investigate the problem of bandwidth allocation in ISP-deployed P2P caches. We consider ISPs that actively allocate cache bandwidth among the overlays in an attempt to maximize the inter-ISPs traffic savings of the cache. We formulate the P2P cache bandwidth allocation problem as a Markov decision process and we show the existence of an optimal stationary allocation policy. We show that the complexity of finding the optimal policy is prohibitive even for a moderate number of ISPs and we propose two approximations to the optimal policy. We perform simulations and experiments and show that cache bandwidth allocation can lead to significant savings in inter-ISP traffic. Based on the insights obtained during the numerical evaluation of the approximate policies, we propose a simple, priority-based, active allocation policy that performed well both in simulations and in experiments with BitTorrent clients on Planet-lab.

Contribution: The author of this thesis developed the Planet-lab framework to run the BitTorrent overlays, implemented a distributed on-line traffic estimator of inter-ISP traffic and performed the experiment-based performance evaluation of the cache bandwidth allocation policies. The second author of the paper implemented and carried out the simulations, and analyzed the resulting data. The third author developed the analytical model, proved the analytical results and designed the cache bandwidth allocation policies. The article was written in collaboration with the third author.

Paper B: Convergence in Player-Specific Graphical Resource Allocation Games

Valentino Pacifici and György Dán

in IEEE Journal on Selected Areas in Communications, December 2012.

Summary: Resource allocation games are a model of the problem of content placement in a network of operator-owned content management systems. In this paper we consider resource allocation games played over a graph: the payoff that an allocated resource yields to a player is amortized if any of its neighbors allocates the same resource. We address the question whether in resource allocation games there exist equilibrium allocations of resources, from which no player has incentive to deviate unilaterally. We prove that Nash equilibria always exist for arbitrary graph topologies, and we compute a bound on the complexity of computing an equilibrium. We then consider the problem of reaching an equilibrium when the players update their allocation to maximize their payoffs, one at a time. We show that, for complete graph topologies, the players reach an equilibrium in a finite number of asynchronous myopic updates. For non-complete graph topologies however, the asynchronous myopic updates of the players might lead to cycles. Nevertheless, we show that if the updates are performed in random order the players would reach a Nash equilibrium. We then consider the case when the players have receive no payoff from allocating resources allocated by their neighbors and we show that there are no cycles. Finally, we relax the requirements of asynchronous updates and we propose an efficient algorithm to reach an equilibrium over arbitrary graph topology and we illustrate its performance through simulations.

Contribution: The author of this thesis developed the analytical model in collaboration with the second author of the paper, proved the analytical results concerning the most general payoff function, carried out the simulations and analyzed the resulting data. The article was written in collaboration with the second author.

Paper C: Content-peering Dynamics of Autonomous Caches in a Content-centric Network

Valentino Pacifici and György Dán

in Proc. of IEEE International Conference on Computer Communications (INFOCOM), 2013.

Summary: In this paper we consider multiple autonomous systems (ASes) in a content-centric network that maintain peering agreements with each other. In order to leverage each other's cache contents to decrease their transit traffic costs, the ASes engage in content-level peering. We develop a model of the interaction and the coordination between the caches managed by peering ASes. We use the

model to investigate whether ASes need to coordinate in order to achieve stable and efficient cache allocations. We show that coordination is not necessary in order to reach stable cache allocations. Nevertheless, avoiding simultaneous cache evictions by peering ASes leads to fast and more efficient convergence to a stable allocation. Furthermore, we show that even if the estimates of the content popularities available to the ASes are inaccurate, content peering is likely to lead to cost efficient cache allocations. Finally we show that it is possible to obtain insight into the structure of the most likely cache allocations.

Contribution: The author of this thesis developed the model in collaboration with the second author of the paper, proved the analytical results for the case of perfect information, implemented and carried out the simulations and analyzed the resulting data. The proofs for the case of imperfect information were carried out in collaboration with the second author of the paper. The article was written in collaboration with the second author.

Publications not included in the thesis

1. V. Pacifici and G. Dan, "Stable Content-peering of Autonomous Systems in a Content-centric Network", in *Proc. of Swedish National Computer Networking Workshop (SNCNW)*, 2013, pp. 1–4

2. V. Pacifici, F. Lehrieder, and G. Dán, "Cache Capacity Allocation for BitTorrent-Like Systems to Minimize Inter-ISP Traffic", in *Proc. of IEEE INFOCOM*, 2012, pp. 1512–1520

3. V. Pacifici and G. Dan, "A Game Theoretic Analysis of Selfish Content Replication on Graphs", *Poster presented at IEEE INFOCOM*, 2012

4. V. Pacifici and G. Dán, "Selfish Content Replication on Graphs", in *Proc. of the 23rd International Teletraffic Congress*, 2011, pp. 119–126

5. V. Pacifici, F. Lehrieder, and G. Dan, "On the Benefits of P2P Cache Capacity Allocation", *Poster presented at the International Teletraffic Congress (ITC)*, pp. 312–313, 2011

6. V. Pacifici and G. Dan, "A Game Theoretic Analysis of Selfish Content Replication on Graphs", in *Proc. of Swedish National Computer Networking Workshop (SNCNW)*, 2011, pp. 1–4

Chapter 6

Conclusions and Future Work

In this thesis, we considered network operators that deploy and operate content management systems in order to improve the efficiency of content delivery within their networks. We evaluated the performance of content delivery in terms of cost for traffic and QoE of operators' subscribers. In the following, we summarize the main contributions of this thesis and we outline some possible directions for future work.

First, we addressed the problem of bandwidth allocation in a stand alone content management system. We considered a network operator that deploys a P2P cache in an attempt of reducing the inter-operator traffic and we investigated the impact of active cache bandwidth allocation on the inter-operator traffic savings of the cache. We demonstrated the importance of capturing the cache bandwidth impact on the characteristics of the P2P overlays and we showed that the savings yielded by cache bandwidth allocation can be significant.

Second, we considered a network of operator-owned content management systems, and analyzed the effect of the interaction of the content placement decisions at different operators. We considered a model of the problem of cooperative caching between autonomous network operators that establish peering agreements with each other. We investigated the existence of stable allocations of content items from which no operator has incentive to deviate unilaterally. Furthermore, we addressed the question of whether network operators would reach a stable allocation if they selfishly and myopically update their content allocation.

Finally, we considered network operators that manage their network of caches in a CCN. We addressed the question of whether the interaction between autonomous peering cache networks would lead to unforeseen instability of content allocation. We proposed a model of the interaction and the coordination between autonomous peering caches and we investigated whether peering network operators need to coordinate in order to reach a stable global allocation of items to caches.

There are still challenges to be addressed in the area of content allocation in networks of operator-owned content management systems. For instance, in the context of interconnected network CDNs, the effects of the interaction among peering network operators need to be investigated. The cost model assumed in such scenario would differ from any cost model considered in this thesis. When interconnecting network CDNs, it would be reasonable to assume that a network operator incurs different traffic costs from retrieving content from different peering CDNs. Similarly, the average latency at which content is served may differ depending on which CDN the content is retrieved from. The effect of such more general cost model on the existence of stable content allocations is unknown. In the case stable allocation from which no operator has incentive to deviate unilaterally do not exist, it might be necessary to design compensation mechanism that lead the system to a stable state.

Bibliography

[1] *Netflix*. [Online]. Available: `http://www.netflix.com`.

[2] Sandvine, "Global Internet Phenomena Report", Tech. Rep.

[3] Cisco, "Visual Networking Index: Forecast and Methodology", 2012.

[4] M. Latouche, J. Defour, T. Regner, T. Verspecht, and F. Le Faucheur, "The CDN Federation - Solutions for SPs and Content Providers To Scale a Great Customer Experience", Cisco, Tech. Rep. October, 2012, pp. 1–11.

[5] *AT&T CDN Service*. [Online]. Available: `http://www.business.att.com/enterprise/Service/hosting-services/content-delivery/distribution/`.

[6] *HP SpeedVideo CDN*. [Online]. Available: `http://www.hp.com/go/cdn`.

[7] *Level3 Content Delivery Network*. [Online]. Available: `http://www.level3.com/en/products-and-services/data-and-internet/cdn-content-delivery-network/`.

[8] *Alcatel-Lucent Velocix CDN*. [Online]. Available: `http://resources.alcatel-lucent.com/?cid=165199`.

[9] *Hulu*. [Online]. Available: `http://www.hulu.com/`.

[10] *Amazon*. [Online]. Available: `http://www.amazon.com/`.

[11] E. Schurman and J. Brutlag, "The User and Business Impact of Server Delays, Additional Bytes, and HTTP Chunking in Web Search", Amazon and Google, Tech. Rep., 2009.

[12] R. Kohavi and R. Longbotham, "Online Experiments: Lessons Learned", *IEEE Computer*, vol. 40, no. 9, pp. 85–87, 2007.

[13] G. Gursun, M. Crovella, and I. Matta, "Describing and forecasting video access patterns", in *Proc. of IEEE INFOCOM Mini Conference*, 2011, pp. 16–20.

[14] G. Szabo and B. A. Huberman, "Predicting the popularity of online content", *Communications of the ACM*, vol. 53, no. 8, p. 80, 2010.

[15] Huawei, *SingleEPC*. [Online]. Available: `http://www.huawei.com/`.

[16] *Ipoque*. [Online]. Available: `http://www.ipoque.com/`.

[17] Radware, *AppDirector*. [Online]. Available: `http://www.radware.com/`.

[18] F. Lehrieder, G. Dán, T. Hoßfeld, S. Oechsner, and V. Singeorzan, "The impact of caching on BitTorrent-like peer-to-peer systems", in *Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P)*, Aug. 2010.

[19] F. Lehrieder, G. Dan, T. Hossfeld, S. Oechsner, and V. Singeorzan, "Caching for BitTorrent-Like P2P Systems: A Simple Fluid Model and Its Implications", *IEEE/ACM Trans. Netw.*, vol. 20, no. 4, pp. 1176–1189, 2012.

[20] A. V. Aho, P. J. Denning, and J. D. Ullman, "Principles of Optimal Page Replacement", *Journal of the ACM*, vol. 18, no. 1, pp. 80–93, 1971.

[21] O. I. Aven, E. G. Coffman Jr., and Y. A. Kogan, *Stochastic Analysis of Computer Storage*. 1987.

[22] O. Bahat and A. M. Makowski, "Optimal replacement policies for non-uniform cache objects with optional eviction", in *Proc. of IEEE INFOCOM*, 2003, pp. 427–437.

[23] S. Jin and A. Bestavros, "Temporal Locality in Web Request Streams: Sources", 1999.

[24] *Squid Internet Object Cache*. [Online]. Available: `http://www.squid-cache.org/`.

[25] D. Lee, J. Choi, J.-H. Kim, S. H. Noh, S. L. Min, Y. Cho, and C. S. Kim, "LRFU: a spectrum of policies that subsumes the least recently used and least frequently used policies", *IEEE Transactions on Computers*, vol. 50, no. 12, pp. 1352–1361, 2001.

[26] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin, "Evaluating content management techniques for Web proxy caches", in *Proc. of Internet Server Performance*, 1999, pp. 1–9.

[27] N. Megiddo and D. S. Modha, "Arc: A self-tuning, low overhead replacement cache", in *Proc. of USENIX FAST*, 2003, pp. 115–130.

[28] S. Irani, "Cost-Aware WWW Proxy Caching Algorithms", in *Proc. of USENIX on Internet Technologies and Systems*, 1997, pp. 193–206.

[29] S. Jin and A. Bestavros, "Popularity-Aware GreedyDual-Size Web Proxy Caching Algorithms", in *Proc. of Distributed Computing Systems*, 2000, pp. 245–261.

[30] D. Niu, Z. Liu, B. Li, and S. Zhao, "Demand forecast and performance prediction in peer-assisted on-demand streaming systems", *Proc. of IEEE INFOCOM*, pp. 421–425, 2011.

[31] D. Niu, C. Feng, and B. Li, "Pricing cloud bandwidth reservations under demand uncertainty", in *Proc. of ACM SIGMETRICS Performance Evaluation Review*, 2012, pp. 151–162.

[32] G. Dan and N. Carlsson, "Dynamic Content Allocation for Cloud-assisted Service of Periodic Workloads", in *Proc. of IEEE INFOCOM*, 2014, pp. 1–9.

[33] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of web server replicas", in *Proc. of IEEE INFOCOM*, 2001, pp. 1587–1596.

[34] E. Cronin, S. Jamin, A. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the Internet", *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1369–1382, 2002.

[35] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, "Optimal content placement for a large-scale VoD system", in *Proc. of Co-NEXT*, 2010, pp. 1–12.

[36] I. Baev, R. Rajaraman, and C. Swamy, "Approximation Algorithms for Data Placement Problems", *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1411–1429, 2008.

[37] B. Awerbuch, Y. Bartal, and A. Fiat, "Distributed Paging for General Networks", *Journal of Algorithms*, vol. 28, no. 1, pp. 67–104, 1998.

[38] S. Borst, V. Gupta, and A. Walid, "Distributed Caching Algorithms for Content Distribution Networks", in *Proc. of IEEE INFOCOM*, 2010, pp. 1478–1486.

[39] H. Che, Z. Wang, and Y. Tung, "Analysis and design of hierarchical Web caching systems", in *Proc. of IEEE INFOCOM*, vol. 3, 2001, pp. 1416–1424.

[40] M. R. Korupolu, C. Plaxton, and R. Rajaraman, "Placement Algorithms for Hierarchical Cooperative Caching", *Journal of Algorithms*, vol. 38, no. 1, pp. 260–302, 2001.

[41] B. Srivastava, Shekhar Krithikaivasan, C. Beard, D. Medhi, W. Alanqar, and A. Nagarajan, "Benefits of Traffic Engineering using QoS Routing Schemes and Network Controls", in *Proc. of Computer Communications*, 2002, pp. 271–275.

[42] P. Eckersley, F. von Lohmann, and S. Schoen, *Packet forgery by ISPs: a report on the Comcast affair*, White paper, Nov. 2007.

[43] BEREC, "A view of traffic management and other practices resulting in restrictions to the open Internet in Europe", Tech. Rep., 2012, pp. 1–39.

[44] OverCache P2P, *Http://www.oversi.com*.

[45] PeerApp UltraBand, *Http://www.peerapp.com*.

[46] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and Principles of Internet Traffic Engineering", Internet Engineering Task Force (IETF), RFC, 2002. [Online]. Available: `http://www.ietf.org/rfc/rfc3272.txt`.

[47] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols", *IEEE Communications Magazine*, vol. 40, no. 10, pp. 118–124, 2002.

[48] R. S. Peterson and E. G. Sirer, "Antfarm : Efficient Content Distribution with Managed Swarms", in *Proc. of NSDI*, 2009, pp. 107–122.

[49] R. S. Peterson, B. Wong, and E. G. Sirer, "A Content Propagation Metric for Efficient Content Distribution", in *Proc. of ACM SIGCOMM*, vol. 41, New York, New York, USA, 2011, pp. 326–337.

[50] A. Sharma, A. Venkataramani, and A. A. Rocha, "Pros & Cons of Model-based Bandwidth Control for Client-assisted Content Delivery", *CoRR*, vol. abs/1209.5, 2012. arXiv: `arXiv:1209.5651v1`.

[51] *CDNetworks*. [Online]. Available: `http://www.cdnetworks.com/`.

[52] *Akamai*. [Online]. Available: `http://www.akamai.com`.

[53] *CloudFlare*. [Online]. Available: `http://www.cloudflare.com/`.

[54] *Amazon CloudFront*. [Online]. Available: `http://aws.amazon.com/cloudfront/`.

[55] S.-H. Chan and F. Tobagi, "Modeling and dimensioning hierarchical storage systems for low-delay video services", *IEEE Transactions on Computers*, vol. 52, no. 7, pp. 907–919, 2003.

[56] A. Merchant and B. Sengupta, "Hierarchical storage servers for video on demand: feasibility, design and sizing", in *Proc. of GLOBECOM*, vol. 1, 1996, pp. 272–278.

[57] T. Kelly and D. Reeves, "Optimal Web cache sizing: Scalable methods for exact solutions", *Computer Communications*, vol. 24, pp. 163–173, 2001.

[58] N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis, "On the Optimization of Storage Capacity Allocation for Content Distribution", *Computer Networks*, vol. 47, no. 3, pp. 409–428, 2005.

[59] D. Wessels and K. Claffy, "Internet Cache Protocol (ICP), version 2", Internet Engineering Task Force (IETF), RFC, 1997, pp. 1–8. [Online]. Available: `https://www.ietf.org/rfc/rfc2186.txt`.

[60] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area Web cache sharing protocol", *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000.

[61] A. Rousskov and D. Wessels, "Cache Digests", in *Proc. of Computer Networks and ISDN Systems*, 1998, pp. 22–41.

[62] S. G. Dykes and K. A. Robbins, "Limitations and Benefits of Cooperative Proxy Caching", *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1290–1304, 2002.

[63] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy, "On the scale and performance of cooperative Web proxy caching", in *Proc. of ACM Symposium on Operating Systems Principles (SOSP)*, vol. 33, 1999, pp. 16–31.

[64] G. Dan, "Cooperative caching and relaying strategies for peer-to-peer content delivery", in *Proc. of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2008, pp. 1–16.

[65] M. Hefeeda and B. Noorizadeh, "On the Benefits of Cooperative Proxy Caching for Peer-to-Peer Traffic", *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 7, pp. 998–1010, 2010.

[66] G. Dán, "Cache-to-cache: Could ISPs cooperate to decrease peer-to-peer content distribution costs?", *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 9, pp. 1469–1482, 2011.

[67] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C. H. Papadimitriou, and J. Kubiatowicz, "Selfish caching in distributed systems: a game-theoretic analysis", in *Proc. of ACM symposium on Principles of Distributed Computing (PODC)*, 2004, pp. 21–30.

[68] I. Milchtaich, "Congestion games with player-specific payoff functions", *Games and Economic Behavior*, vol. 13, no. 1, pp. 111–124, 1996.

[69] N. Laoutaris, O. Telelis, V. Zissimopoulos, and I. Stavrakakis, "Distributed selfish replication", *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 12, pp. 1401–1413, 2006.

[70] G. Dán, T. Hoß feld, S. Oechsner, P. Cholda, R. Stankiewicz, I. Papafili, and G. D. Stamoulis, "Interaction Patterns between P2P Content Distribution Systems and ISPs", *IEEE Communications Magazine*, vol. 49, no. 5, pp. 220–230, 2011.

[71] V. Bilò, A. Fanelli, M. Flammini, and L. Moscardelli, "Graphical Congestion Games.", in *Proc. of WINE*, vol. 5385, 2008, pp. 70–81.

[72] D. Fotakis, V. Gkatzelis, A. C. Kaporis, and P. G. Spirakis, "The Impact of Social Ignorance on Weighted Congestion Games", in *Proc. of WINE*, 2009, pp. 316–327.

[73] 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.

[74] 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, 2007, pp. 181–192.

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

[76] A. Carzaniga, M. J. Rutherford, and A. L. Wolf, "A Routing Scheme for Content-Based Networking", in *Proc. of IEEE INFOCOM*, 2004, pp. 1–11.

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

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

[79] V. Valancius, B. Ravi, N. Feamster, and A. C. Snoeren, "Quantifying the benefits of joint content and network routing", in *Proc. of ACM SIGMETRICS Performance Evaluation Review*, 2013, pp. 243–254.

[80] A.-M. K. Pathan and R. Buyya, "A Taxonomy and Survey of Content Delivery Networks", The University of Melbourne, Australia, Tech. Rep., 2007, pp. 1–44.

[81] E. Bertrand, E. Stephan, T. Burbridge, P. Eardley, K. Ma, and G. Watson, "Use Cases for Content Delivery Network Interconnection", Internet Engineering Task Force (IETF), RFC, 2012. [Online]. Available: `http://tools.ietf.org/html/rfc6770`.

[82] Y. Le Louedec, G. Bertrand, C. Goutard, N. Amann, C. Hawinkel, D. De Vleeschauwer, V. Bonneau, S. Nakajima, S. Ropert, S. Lubrano, G. Fontaine, S. Girieud, M. Leiba, C. Hellge, F. Kozamernik, and B. Tullemans, "Final requirements for Open Content Aware Networks", FP7 - Open ContEnt Aware Networks (OCEAN), Deliverable D2.2, 2013.

[83] S. Puopolo, M. Latouche, F. Le Faucheur, and J. Defour, "Content Delivery Network (CDN) Federations - How SPs Can Win the Battle for Content-Hungry Consumers", Cisco, Tech. Rep., 2011, pp. 1–9.

[84] L. Peterson and B. Davie, "Framework for CDN Interconnection", Internet Engineering Task Force (IETF), RFC, 2013. [Online]. Available: `http://datatracker.ietf.org/doc/draft-ietf-cdni-framework/`.

[85] "Content Delivery Networks Interconnection", Internet Engineering Task Force (IETF), RFC, 2013. [Online]. Available: `http://datatracker.ietf.org/wg/cdni/`.

[86] AT&T Press Room, *Akamai and AT&T Forge Global Strategic Alliance to Provide Content Delivery Network Solutions*, 2012. [Online]. Available: `http://www.att.com/gen/press-room?pid=23598%5C&cdvn=news%5C&newsarticleid=35788%5C&mapcode=`.

[87] Akamai, *Aura Managed CDN*. [Online]. Available: `http://www.akamai.com/html/solutions/aura%5C_managed%5C_cdn.html`.

[88] EdgeCast, *Licensed CDN*. [Online]. Available: `http://www.edgecast.com/solutions/licensed-cdn/`.

[89] Akamai, *Aura Licensed CDN*. [Online]. Available: `http://www.akamai.com/html/solutions/aura%5C_licensed%5C_cdn.html`.

[90] V. Pacifici and G. Dan, "Stable Content-peering of Autonomous Systems in a Content-centric Network", in *Proc. of Swedish National Computer Networking Workshop (SNCNW)*, 2013, pp. 1–4.

[91] V. Pacifici, F. Lehrieder, and G. Dán, "Cache Capacity Allocation for BitTorrent-Like Systems to Minimize Inter-ISP Traffic", in *Proc. of IEEE INFOCOM*, 2012, pp. 1512–1520.

[92] V. Pacifici and G. Dan, "A Game Theoretic Analysis of Selfish Content Replication on Graphs", *Poster presented at IEEE INFOCOM*, 2012.

[93] V. Pacifici and G. Dán, "Selfish Content Replication on Graphs", in *Proc. of the 23rd International Teletraffic Congress*, 2011, pp. 119–126.

[94] V. Pacifici, F. Lehrieder, and G. Dan, "On the Benefits of P2P Cache Capacity Allocation", *Poster presented at the International Teletraffic Congress (ITC)*, pp. 312–313, 2011.

[95] V. Pacifici and G. Dan, "A Game Theoretic Analysis of Selfish Content Replication on Graphs", in *Proc. of Swedish National Computer Networking Workshop (SNCNW)*, 2011, pp. 1–4.

Paper **A**

Cache Bandwidth Allocation for P2P File Sharing Systems to Minimize Inter-ISP Traffic

Valentino Pacifici and Frank Lehrieder and György Dán

submitted to IEEE/ACM Transacions on Networking (TON), second revision.

Cache Bandwidth Allocation for P2P File Sharing Systems to Minimize Inter-ISP Traffic

Valentino Pacifici*, Frank Lehrieder⋆, György Dán*

*School of Electrical Engineering,
KTH Royal Institute of Technology, Stockholm, Sweden

⋆Institute of Computer Science,
University of Würzbur, Würzburg, Germany

Abstract

Many Internet service providers (ISPs) have deployed peer-to-peer (P2P) caches in their networks in order to decrease costly inter-ISP traffic. A P2P cache stores parts of the most popular contents locally, and if possible serves the requests of local peers to decrease the inter-ISP traffic. Traditionally, P2P cache resource management focuses on managing the storage resource of the cache so as to maximize the inter-ISP traffic savings. In this paper we show that, when there are many overlays competing for the upload bandwidth of a P2P cache, then in order to maximize the inter-ISP traffic savings the cache's upload bandwidth should be actively allocated among the overlays. We formulate the problem of P2P cache bandwidth allocation as a Markov decision process, and propose three approximations to the optimal cache bandwidth allocation policy. We use extensive simulations and experiments to evaluate the performance of the proposed policies, and show that the bandwidth allocation policy that prioritizes swarms with a small ratio of local peers can improve the inter-ISP traffic savings in BitTorrent-like P2P systems by up to 30 to 60 percent.

1 Introduction

The number of peer-to-peer applications has increased significantly in recent years, and so has the amount of Internet traffic generated by peer-to-peer (P2P) applications. P2P traffic accounts for up to 70% of the total network traffic, depending on geographical location [1], and is a significant source of inter-ISP traffic. Inter-ISP traffic can be a source of revenue for tier-1 ISPs, but it is a source of transit traffic costs for ISPs at the lower levels of the ISP hierarchy, e.g., for tier-2 and tier-3 ISPs. Some ISPs have attempted to limit their costs due to P2P applications by throttling P2P traffic [2]. Nevertheless, the users of P2P applications constitute a significant share of the ISPs' customer base, and hence a solution

that negatively affects the performance of P2P applications can result in a decrease of an ISP's revenues on the long term.

Recent research efforts have tried to decrease the amount of inter-ISP P2P traffic by introducing locality-awareness in the neighbor-selection policies of popular P2P applications, like BitTorrent [3–6]. Locality information can be provided by the ISPs [3–5] or can be obtained via measurements [6], and is used to prioritize nearby peers to distant ones when exchanging data. Through exchanging data primarily with nearby peers a P2P application can improve the locality of its traffic, and hence, can decrease inter-ISP traffic. Nevertheless, locality-aware neighbor selection can deteriorate the performance and the robustness of a P2P application [7].

To address the problem of increased inter-ISP traffic, many ISPs have deployed P2P caches [8, 9]. P2P caches, similar to web proxy caches, decrease the amount of inter-ISP traffic by storing the most popular contents in the ISP's own network, so that they do not have to be downloaded from peers in other ISPs' networks. According to measurement studies 30 to 80 percent of P2P traffic is cacheable [10, 11]. Nevertheless, the actual efficiency of a cache depends on two main factors. First, the amount of storage, which determines the share of the contents that can be kept in cache. Second, the available bandwidth of the cache, which determines the rate at which data can be served by the cache, if the data are in storage.

The goal of cache storage management is to maximize the probability that data are found in the cache when requested. The algorithms for cache storage management, called cache eviction policies, in the case of P2P caches differ significantly from those in the case of web proxy caching. Web objects are typically small, and consequently eviction policies can replace entire contents at once [12]. Objects in P2P systems are nevertheless typically too big to be replaced at once, so that eviction policies for P2P caches have to allow partial caching of contents [10, 11]. By allowing partial caching, P2P eviction policies can achieve within 10 to 20 percent of the optimal offline eviction policy [10, 11].

The impact of the cache bandwidth and its management has received little attention, even though cache bandwidth can be costly, as caches are often priced based on their bandwidth [8, 9]. In the case of web proxy caching bandwidth management is not necessary, because the incoming inter-ISP traffic saving equals the amount of data served from the cache. In the case of a P2P cache the inter-ISP traffic saving is, however, not only determined by how much data the cache serves but also by the characteristics of the overlay to which the data is served [13].

The fundamental question we address in this paper is whether given a limited amount of P2P cache bandwidth, the bandwidth can be actively managed such as to minimize the amount of inter-ISP traffic. We make three important contributions to answer this question. First, we provide a mathematical formulation of the cache bandwidth allocation problem, and show the existence of a stationary optimal policy. Second, we use the proposed mathematical model and insights from [13, 14] to derive three allocation policies to approximate the optimal policy. Third, through simulations and through experiments on Planet-lab we show that by actively allocating the upload bandwidth between different overlays the inter-ISP traffic savings due to P2P caches can be improved significantly. We identify the

heterogeneity of the ratio of local peers in the swarms as the key factor that determines the potential traffic savings.

The rest of the paper is organized as follows. In Section 2 we review the related work. In Section 3 we model the system and its evolution using a Markov jump process. In Section 4 we formulate the problem of cache bandwidth allocation and show the existence of an optimal cache bandwidth allocation policy. Section 5 describes three policies to approximate the optimal policy. In Section 6 we use simulation and experiment results to quantify the potential of the proposed bandwidth allocation policies and to provide insight into the characteristics of an optimal policy. Section 7 concludes the paper.

2 Related work

The solutions for ISP-friendly P2P application design proposed in the literature fall into three main categories: peer-driven, ISP-driven and caching [15]. Peer-driven solutions adapt the neighbor selection strategy of the peers by relying on measurements of latency [16], on autonomous system (AS) topology map information [5] or on third-party infrastructures like content delivery networks [6]. Motivated by the difficulty of inferring the ISPs' interests based on measurements [3, 4] investigated the use of ISP-provided information to influence peer selection. All these works make P2P systems more ISP-friendly by influencing the overlay construction, and are complementary to P2P caching.

Caching of P2P contents has been the subject of several works. Most works focused on the achievable cache hit ratios [17, 18], and on the efficiency of various cache eviction policies [10, 11]. Our work is orthogonal to the works on cache eviction policies, as we assume the existence of a cache eviction policy, and we consider the impact of allocating the cache's upload bandwidth between competing overlays on the amount of inter-ISP traffic generated by the overlays.

Cache upload bandwidth management for P2P video streaming systems was considered in [19, 20] in order to decrease the ISPs' incoming transit traffic. In the case of streaming the download rate of peers is determined by the video rate, and the received rate does not influence the peers' behavior. This makes the problem of cache bandwidth allocation for streaming systems significantly different from the problem considered in this paper. We do not only consider the impact of the cache upload rate on the instantaneous inter-ISP traffic, but also its impact on the system dynamic.

Close to our work is [21] where the authors studied the impact of different bandwidth reservation schemes between two overlays via simulations. They concluded that the impact of cache bandwidth allocation was minor, which can be attributed to the inefficiency of the cache bandwidth utilization under the considered schemes. Compared to [21] in this paper we give a mathematical formulation of the problem of cache bandwidth allocation, use analytical models of the swarm dynamics and the inter-ISP traffic to give insight into the characteristics of an optimal allocation policy, and use simulations and experiments to demonstrate the inter-ISP traffic savings achievable through cache bandwidth allocation.

In [22–24] the authors proposed schemes for bandwidth allocation among multiple

swarms in P2P systems. In these works the initial seeder is the bandwidth allocator that attempts to maximize the total download rate of the system so as to minimize the download latencies of the peers. The most fundamental difference between our work and [22–24] is that we aim at minimizing the amount of inter-ISP traffic generated by the overlays. [22, 23] consider managed swarms, while in our work caching is performed transparently to the peers. In [23] the authors assume peers belonging to multiple swarms. The bandwidth allocation among swarms is implemented by these peers as they follow the prioritization scheme suggested by the coordinator. More similar to our work is [24], where the authors implement a simple model-based controller for server bandwidth in BitTorrent systems. Due to the large amount of data needed to parametrize the model, the authors question the practicality of their approach [24].

Our work relies on the analytical models of the system dynamics of BitTorrent-like systems in [13, 25–29]. These works used a Markovian model of the system dynamics of BitTorrent-like systems to model the service capacity and the scalability [25, 26], to evaluate the impact of peer upload rate allocation between two classes of peers [27], to assist the dimensioning of server assisted hybrid P2P content distribution [29], and to evaluate the impact of caches on the swarm dynamics and on the amount of inter-ISP traffic for a single overlay [13]. Our work differs significantly from these works, as we consider multiple overlays and use the fluid model of the system dynamics to get insight into the characteristics of an optimal P2P cache bandwidth allocation policy.

In our work we model the cache bandwidth allocation problem in P2P systems as a Markov Decision Process (MDP). MDPs were used in [30–32] to analyze schemes for incentivizing fair resource reciprocation and for discouraging free riding in P2P systems. Compared to [31–33], in our work we use a MDP to prove the existence of an optimal cache allocation policy.

3 System Model

In the following we describe our model of a multi-swarm file-sharing system and our model of cache bandwidth allocation. The model captures the effect of the cache bandwidth allocation on the evolution of the system.

We consider a set $\mathcal{I} = \{1, \ldots, I\}$ of ISPs, and a set of swarms $\mathcal{S} = \{1, \ldots, S\}$, whose peers are spread over the ISPs. Peers are either leechers, which download and upload simultaneously, or seeds, which upload only. Leechers arrive to swarm s according to a Poisson process with intensity λ_s, the arrival rate of leechers in ISP i is $\lambda_{i,s}$. The Poisson process can be a reasonable approximation of the arrival process over short periods of time [34], even if the arrival rate of peers varies over the lifetime of a swarm. We model the leechers' impatience by the abort rate θ. A leecher departs at this rate before downloading the entire content. Seeds depart from the swarm at rate γ, so that a seed stays on average $1/\gamma$ time in the swarm. The upload rate of peers is denoted by μ and their download rate by c. We focus on the case when $\mu < c$. For simplicity we consider that all files have the same size, and thus, μ and c can be normalized by the file size. Finally, we assume that leechers

can use a share η of their upload rate due to partial content availability. This model of swarm dynamics was used in [13, 25, 26, 28, 29].

We denote by $X_{i,s}(t)$ the number of leechers in ISP i in swarm s at time t, and by $Y_{i,s}(t)$ the number of seeds in ISP i in swarm s at time t. $X_{i,s}(t)$ and $Y_{i,s}(t)$ take values in the countably infinite state space $\mathbb{N}_0$. As a shorthand we introduce $Z_{i,s}(t) = (X_{i,s}(t), Y_{i,s}(t))$ and $Z_s(t) = (Z_{i,s}(t))_{i\in\mathcal{I}}$. Finally, we denote the state of the swarms by $Z(t) = (Z_s(t))_{s\in\mathcal{S}}$.

Seeds and leechers in ISP i can upload and download data to and from peers in any ISP $j \in \mathcal{I}$. We define the publicly available upload rate $u^P_{i,s}(t)$ as the available upload rate located in ISP i that can be used by leechers of swarm s in any ISP. This quantity tantamounts the upload rate of the leechers and the seeds $u^P_{i,s}(t) = \mu(\eta X_{i,s}(t) + Y_{i,s}(t))$. A leecher cannot download from itself, therefore the publicly available upload rate in ISP i to a local leecher of swarm s is $u^{PL}_{i,s}(t) = \max[0, \mu(\eta(X_{i,s}(t) - 1) + Y_{i,s}(t)]$.

3.1 P2P Cache Bandwidth Allocation Policies

The ISPs, as they are located in the lower layers of the ISP hierarchy, are interested in decreasing the inter-ISP traffic generated by the peers. In order to decrease its inter-ISP traffic, ISP $i \in \mathcal{I}$ maintains a cache with upload *bandwidth capacity* $K_i < \infty$, which acts as an ISP managed super peer [8]. The abstraction of a P2P cache as a source of upload bandwidth is motivated by that P2P caches are often priced by their maximum upload rates. Since every ISP's goal is to decrease its own incoming inter-ISP traffic, it is reasonable to assume that the cache operated by ISP i only serves leechers in ISP i.

ISP i can implement an *active* cache bandwidth allocation policy to control the amount of cache bandwidth $\kappa_{i,s}(t)$ available to leechers in ISP i belonging to swarm s. We denote the cache bandwidth allocation of ISP i at time t by the vector $\kappa_i(t) = (\kappa_{i,1}(t), \ldots, \kappa_{i,S}(t))$, and the set of feasible cache bandwidth allocations of ISP i by $\mathcal{K}_i = \{\kappa_i | \sum_{s\in\mathcal{S}} \kappa_{i,s} \leq K_i\} \subseteq [0, K_i]^{|\mathcal{S}|}$. We also make the reasonable assumption that $\kappa_{i,s}(t) > 0$ for a swarm s only if the corresponding file is at least partially cached at ISP i at time t.

Given the set $\mathcal{K}_i$ of feasible cache bandwidth allocations for ISP i, a cache bandwidth allocation *policy* π defines $\kappa_i(t)$ as a function of the system's history up to time t, i.e., $(Z(u))_{u<t}$, and past cache allocations $(\kappa_i(u))_{u<t}$. We denote the set of all cache bandwidth allocation policies by Π.

3.2 Caching and System Dynamics

Consider a policy π implemented by ISP i. We model the evolution of the swarms' state by an $I \times S \times 2$ dimensional continuous-time Markov jump process $\mathcal{Z}^\pi = \{Z(t), t \geq 0\}$, which is a collection of S coupled $I \times 2$ dimensional continuous-time Markov jump processes $\mathcal{Z}^\pi_s = \{Z_s(t), t \geq 0\}$.

Consider now a swarm $s \in \mathcal{S}$ under policy π, and denote the transition intensity from state z_s to state z'_s by $q^\pi_{z_s,z'_s}$. Denote by e_i the I dimensional vector whose i^{th} component is 1. The transition intensities from state $z_s = (x_s, y_s)$ are $q^\pi_{z_s,(x_s+e_i,y_s)} = \lambda_{i,s}$ (leecher arrival), $q^\pi_{z_s,(x_s-e_i,y_s)} = \theta x_{i,s}$ (leecher abort), and $q^\pi_{z_s,(x_s,y_s-e_i)} = \gamma y_{i,s}$ (seed departure). The transition

Parameter	Definition
$\mathcal{I}$, $\mathcal{S}$	Set of ISPs and set of swarms, respectively
$\kappa_{i,s}$	Cache bandwidth allocation of ISP i to swarm s
$\lambda_{i,s}$	Arrival rate of leechers to swarm s in ISP i
θ	Abort rate of leechers
γ	Departure rate of seeds
η	Effectiveness of file sharing
μ, c	Peer upload and download capacity, respectively
$X_{i,s}(t)$	Number of leechers in ISP i in swarm s at time t
$Y_{i,s}(t)$	Number of seeds in ISP i in swarm s at time t
$u_{i,s}^{PL}(t)$	Upload rate in ISP i available to all leechers in swarm s

Table 1: Frequently used notation

intensity to state $(x_s - e_i, y_s + e_i)$, called the download completion rate, is a function of the maximum download rate of the leechers, and the available upload rate to leechers in ISP i.

3.2.1 The case of no cache

Without a cache ($K_i = 0$) the leechers in ISP i would get a share $x_{i,s}/\sum_i x_{i,s}$ of the total upload rate $u_s^P = \sum_i u_{i,s}^P$ [25, 26, 28, 29]. The download completion rate in this case can be expressed as

$$q^{\pi}_{(x_s,y_s),(x_s-e_i,y_s+e_i)} = \min(cx_{i,s}, u_s^P x_{i,s}/\textstyle\sum_i x_{i,s}). \tag{1}$$

We refer to the process defined this way as the *uncontrolled* stochastic process, and we denote it by $\mathcal{Z}$.

3.2.2 The case of cache

Consider that the instantaneous cache bandwidth allocated to swarm s is $\kappa_{i,s}$. The cache bandwidth increases the available upload rate, so that the download completion rate becomes

$$q^{\pi}_{(x_s,y_s),(x_s-e_i,y_s+e_i)} = \min(cx_{i,s}, u_s^P x_{i,s}/\textstyle\sum_i x_{i,s} + \kappa_{i,s}). \tag{2}$$

Since the cache bandwidth allocation can influence the transition intensities of the stochastic process, we refer to $\mathcal{Z}^{\pi}$ as the *controlled* stochastic process. Table 1 summarizes the notation used in the paper.

4 The Optimal Cache Bandwidth Allocation Problem and Stationary Policy

In this section we formulate the optimal cache bandwidth allocation problem and we show the existence of an optimal stationary policy.

The primary goal of ISP i when allocating cache bandwidth to swarm s is to decrease the inter-ISP traffic. Cache bandwidth allocation inherently affects the upload rate available to the leechers, and hence, it can affect the evolution of the process $\mathcal{Z}_s^\pi$.

Let us denote by $I_{i,s}(Z_s(t), \kappa_{i,s}(t))$ the rate of the incoming inter-ISP traffic in ISP i due to swarm s as a function of the cache bandwidth $\kappa_{i,s}(t)$ allocated to swarm s by ISP i and the swarm's state $Z_s(t)$. $I_{i,s}(Z_s(t), \kappa_{i,s}(t))$ also depends on $\kappa_{j,s}(t)$ of ISPs $j \neq i$, but as we focus on the bandwidth allocation problem of ISP i, for simplicity we assume that $\kappa_{j,s}(t) = \kappa_{j,s}$ constant.

We can express the expected amount of incoming inter-ISP traffic under policy $\pi \in \Pi$ from time $t = 0$ until time T as

$$C_i^\pi(z,T) = E_z^\pi \left[\int_0^T \sum_{s \in \mathcal{S}} I_{i,s}(Z_s(t), \kappa_{i,s}(t)) dt \right],$$

where E_z^π denotes the expectation under policy π with initial state $Z(0) = z$.

Given the set Π of feasible cache bandwidth allocation policies, we define the cache bandwidth allocation problem for ISP i as finding the cache bandwidth allocation policy $\pi^* \in \Pi$ that minimizes the average incoming inter-ISP traffic rate $C_i^\pi(z)$ due to P2P content distribution, that is

$$\inf_{\pi \in \Pi} C_i^\pi(z) = \inf_\pi \limsup_{T \to \infty} \frac{1}{T} C_i^\pi(z,T). \tag{3}$$

Consequently, the optimal cache bandwidth allocation problem can be modeled as a continuous-time Markov decision process (MDP) with the optimality criterion defined in (3).

4.1 Optimal Cache Bandwidth Allocation

The first two fundamental questions that we are to answer are (i) whether there is an optimal cache bandwidth allocation policy π^* that solves (3), and (ii) whether there is an optimal policy whose choices only depend on the *current* system state $Z(t)$. Such a policy is called *stationary*. In general, an optimal stationary policy might not exist for a MDP when the action space or the state space is infinite. The following theorem shows that for the cache bandwidth allocation problem there exists an optimal stationary policy.

Theorem 1. *There exists an optimal stationary policy π^* that minimizes the average traffic $C_i^\pi(z)$ of ISP i.*

Proof. Recall that the controlled processes $\mathcal{Z}_s^\pi$ are coupled through the bandwidth allocation policy π. In the following we define four criteria *C1-C4* for $\mathcal{Z}^\pi$ and we use them to prove the theorem.

C1: The set $\mathcal{K}_i$ of cache bandwidth allocations is compact.

C2: For every state $z=(x,y)$ the incoming inter-ISP traffic rate $\sum_s I_{i,s}(z_s,\kappa_{i,s})$ and the transition intensities $(q^\pi_{(x_s,y_s),(x_s-e_i,y_s+e_i)})_{s\in\mathcal{S}}$ are continuous functions of $\kappa_{i,s}$.

C3: Define $H(z)=C_i^\pi(z)-C_i^\pi(a)$, where a is an arbitrarily chosen state. Then $\sum_{z'} H(z')q^\pi_{z,z'}$ is continuous in $\kappa_{i,s}$ for every state z.

C4: The average inter-ISP traffic $C_i^\pi(z)$ is finite for every policy π and initial state z.

We now formulate the following Lemma based on (Theorem 5.9 in [35]).

Lemma 1. *For a continuous-time MDP with countably infinite state space and non-negative cost, under* C1-C4 *there exists a stationary policy π^* that is average cost optimal.*

Since the cost function $C_i^\pi(z)$ defined in (3) is the average cost, in order to prove the theorem it is sufficient to show that $\mathcal{Z}^\pi$ fullfills the criteria *C1-C4*.

Proof of C1-C3: *C1* follows from $0\le\kappa_{i,s}(t)\le K_i<\infty$. $\sum_s I_{i,s}(z_s,\kappa_{i,s})$ is continuous by assumption, the continuity of the transition intensities $(q^\pi_{(x_s,y_s),(x_s-e_i,y_s+e_i)})_{s\in\mathcal{S}}$ w.r.t $\kappa_{i,s}$ follows from (2). *C3* follows from the finiteness of $C_i^\pi(z)$ and from *C2*.

Proof of C4: In order to show the finiteness of the average inter-ISP traffic $C_i^\pi(z)$ for every policy π and initial state z, we show that $\mathcal{Z}_s^\pi$ satisfies the Foster-Lyapunov condition for every $s\in\mathcal{S}$, then we give a bound on the inter-ISP traffic rate in every state of the system. Let us define the Lyapunov function $w(z_s)=\sum_i(x_{i,s}+y_{i,s})+1$. Also, let us define the sequence $(t_n)_{n\ge0}$ of time instants, which consists of the transition epochs of the process and of the instants when $\kappa_{i,s}(t)$ changes according to the policy π. Finally, we define the generalized average drift

$$AW(z_s)=E[w(Z_s(t_{n+1}))-w(Z_s(t_n))|Z_s(t_n)=z_s]. \tag{4}$$

Consider now the Foster-Lyapunov average drift condition [36]

$$|AW(z_s)|<\infty \quad \forall z_s, \text{ and } \; AW(z_s)<-\varepsilon \quad z_s\notin C, \tag{5}$$

where $\varepsilon>0$ and $C\subset\mathbb{N}_0^{|\mathcal{I}|\times2}$ is finite. For $\lambda_s<\infty$ the uncontrolled process $\mathcal{Z}_s$ satisfies (5): $|AW(z_s)|\le1$ due to the random-walk structure of the process, and $AW(z_s)=(\lambda_s-\theta x_s-\gamma y_s)/(-q_{z_s,z_s})<-\varepsilon$ for x_s or y_s sufficiently big. Consider now the mean drift $AW^\pi(z_s)$ of the controlled process. Again, $|AW^\pi(z_s)|\le1$. Furthermore we have

$$AW^\pi(z_s)\le AW(z_s)\frac{-q_{z_s,z_s}}{-q_{z_s,z_s}-K_i}<-\varepsilon\frac{-q_{z_s,z_s}}{-q_{z_s,z_s}-K_i}<0.$$

Consequently, the controlled process $\mathcal{Z}_s^\pi$ also satisfies the Foster-Lyapunov average drift condition. Since the process is aperiodic and irreducible, the drift condition guarantees

ergodicity [36]. Furthermore, for $\tilde{M} = c > 0$ it holds that $I_{i,s}(z_s, \kappa_{i,s}) \leq \tilde{M} w(z_s)$. This together with the ergodicity of all $\mathcal{Z}_s^{\pi}$ implies that $C_i^{\pi}(z)$ is finite and concludes the proof. □

A consequence of Theorem 1 is that the optimal bandwidth allocation policy π^* is such that the allocation $\kappa_i(t)$ is only a function of the system state $Z(t)$, hence it is constant between the state transitions of $\mathcal{Z}^{\pi*}$.

The optimal policy π^* can be found using the policy iteration algorithm [35], but it requires the solution of the steady state probabilities of the controlled Markov processes $\mathcal{Z}^{\pi}$. This can be prohibitive even for a moderate number of ISPs and swarms. In the next section we propose and discuss different approximations.

5 Cache Bandwidth Allocation Policies

In this section we first discuss a baseline for bandwidth sharing. We then describe three approximations to the optimal cache bandwidth allocation policy.

Throughout the section we assume that the inter-ISP traffic functions $I_{i,s}(z_s, \kappa_{i,s})$ are known, and are continuous convex non-increasing functions of $\kappa_{i,s}$. The assumptions of continuity, convexity and non-increasingness are rather natural.

5.1 Demand-driven Bandwidth Sharing (DDS)

As a baseline for comparison, consider that ISP i does *not* actively allocate its cache bandwidth K_i, therefore leechers at different swarms compete with one another for cache bandwidth. The cache in ISP i maintains a drop-tail queue to store the requests received from the leechers in ISP i, and serves the requests according to a first-in-first-out (FIFO) policy at the available upload bandwidth K_i. Let us denote by $\alpha_{i,s}$ the rate at which leechers of swarm s in ISP i request data from the cache in ISP i, and denote by $\sigma_{i,s}$ the mean service time of these requests. Then the offered load of swarm s to the cache is $\rho_{i,s} = \alpha_{i,s}\sigma_{i,s}$. Clearly, if $\rho_{i,s} \geq 1$ then the FIFO queue is in a blocking state with probability $p_i^b > 0$.

If the requests from leechers in every swarm arrive according to a Poisson process, then the aggregate arrival process is Poisson. Since the arrival process is Poisson, an arbitrary request is blocked (i.e., dropped) with probability $p_{i,s}^b = p_i^b$ despite the possibly heterogeneous mean service times due to the PASTA property [37]. The effective (i.e., not blocked) load for swarm s can be expressed as $(1 - p_i^b)\rho_{i,s}$, and consequently the share of cache bandwidth used to serve requests for swarm s can be estimated as

$$\frac{\kappa_{i,s}}{\sum_{s\in\mathcal{S}} \kappa_{i,s}} = \frac{(1-p_i^b)\rho_s}{\sum_{s\in\mathcal{S}} (1-p_i^b)\rho_s} = \frac{\rho_s}{\sum_{s\in\mathcal{S}} \rho_s}. \tag{6}$$

In general, if the arrival process of requests is not Poisson then (6) does not hold. Nevertheless, as under the assumption of a Poisson request arrival process the cache bandwidth is shared among the swarms proportional to the offered load (demand) of the swarms, we

refer to this policy as the *demand-driven sharing* (*DDS*) policy.

5.2 One-step Look Ahead Allocation Policy (OLA)

The one-step look ahead (OLA) policy π^{OLA} is a simple approximation of the optimal stationary cache bandwidth allocation policy π^*.

Consider the controlled Markov process $\mathcal{Z}^{\pi^{OLA}}$, and let us denote the n^{th} transition epoch of the process by t_n. Then according to the OLA policy the cache bandwidth allocation $\kappa_i(t)$ of ISP i for $t_n < t \leq t_{n+1}$ is such that it minimizes the incoming inter-ISP traffic rate given the state $Z(t_n) = z$ of the process $\mathcal{Z}^{\pi^{OLA}}$

$$\kappa_i(t) = \arg\min_{\kappa_i \in \mathcal{K}_i} \sum_{s \in \mathcal{S}} I_{i,s}(z_s, \kappa_{i,s}). \tag{7}$$

By following the OLA policy the ISP minimizes the incoming inter-ISP traffic in every state of the process $\mathcal{Z}^{\pi^{OLA}}$. The OLA policy *adapts* to the system state, but unlike the optimal policy π^*, it does not consider the impact of cache bandwidth allocation on the evolution of the number of peers.

Recall that, by assumption, $I_{i,s}(z_s, \kappa_{i,s})$ are continuous convex non-increasing functions of $\kappa_{i,s}$ for every state z_s. In order to obtain the optimal solution to (7) consider the Lagrangian

$$L(z, \kappa_i, \zeta) = \sum_{s \in \mathcal{S}} I_{i,s}(z_s, \kappa_{i,s}) - \zeta \Big(\sum_{s \in \mathcal{S}} \kappa_{i,s} - K_i \Big), \tag{8}$$

where $\zeta \leq 0$ is the Lagrange multiplier. Then

$$\frac{\partial L(z, \kappa_i, \zeta)}{\partial \kappa_{i,s}} = \frac{\partial I_{i,s}(z_s, \kappa_{i,s})}{\partial \kappa_{i,s}} - \zeta \quad \text{and} \quad \frac{\partial L(z, \kappa_i, \zeta)}{\partial \zeta} = K_i - \sum_{s \in \mathcal{S}} \kappa_{i,s}. \tag{9}$$

Hence, a minimum of L over $\mathcal{K}_i$ is characterized by

$$\begin{aligned} \kappa_{i,s} > 0 \quad &\Rightarrow \quad \frac{\partial_+ I_{i,s}(z_s, \kappa_{i,s})}{\partial \kappa_{i,s}} \geq \zeta \geq \frac{\partial_- I_{i,s}(z_s, \kappa_{i,s})}{\partial \kappa_{i,s}} \\ \kappa_{i,s} = 0 \quad &\Rightarrow \quad \frac{\partial_- I_{i,s}(z_s, \kappa_{i,s})}{\partial \kappa_{i,s}} \geq \zeta, \end{aligned}$$

where ∂_+ and ∂_- denote the right and the left derivative of a semi-differentiable function. Since $\mathcal{K}_i$ is compact and convex, such a minimum exists and can be found using a projected subgradient method [38].

An important insight from the OLA policy is the following. If $I_{i,s}(z_s, \kappa_{i,s})$ are continuously differentiable then at optimality every swarm with non-zero cache bandwidth allocation provides equal marginal traffic saving. If $I_{i,s}(z_s, \kappa_{i,s})$ are not continuously differentiable, then for swarms with non-zero cache bandwidth allocation the intersection of the subdifferentials is non-empty.

5.3 Steady-state Optimal Allocation Policy (SSO)

The opposite of the *OLA* policy is to focus on the long-term evolution of the controlled Markov process $\mathcal{Z}^{\pi}$, that is, on the incoming inter-ISP traffic in steady-state and to consider time-independent cache bandwidth allocation policies $\overline{\pi} = \kappa_i$.

Let us denote the expected number of leechers and seeds in steady-state as a function of the cache bandwidth allocation policy $\overline{\pi}$ by $\overline{x}_{i,s}^{\overline{\pi}}$ and by $\overline{y}_{i,s}^{\overline{\pi}}$, respectively. They were shown to be a function of the cache upload rate $\kappa_{i,s}$ allocated to swarm s [13, 14]. As long as the total available upload rate is less than or equal to the total download rate of the leechers

$$\overline{x}_{i,s}^{\overline{\pi}} = \frac{\lambda_{i,s}}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_{i,s}}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} - \Delta_i(\mathbf{x},\mathbf{y},\kappa) \tag{10}$$

$$\overline{y}_{i,s}^{\overline{\pi}} = \frac{\lambda_{i,s}}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_{i,s}\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\theta}{\gamma}\Delta_i(\mathbf{x},\mathbf{y},\kappa), \tag{11}$$

where $\frac{1}{\nu} = \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma}) \geq 0$ [13, 26] and

$$\Delta_i(\mathbf{x},\mathbf{y},\kappa) = \frac{\sum_{j\in\mathcal{I}}\left(\lambda_{i,s}\kappa_{j,s} - \kappa_{i,s}\lambda_{j,s}\right)}{\eta\gamma\left(1+\frac{\theta}{\nu}\right)\left(\sum_{j\in\mathcal{I}}\left(\lambda_{j,s} - \kappa_{j,s}\right)\right)}. \tag{12}$$

Otherwise, when the total upload rate exceeds the total download rate, increasing the cache bandwidth allocated to the swarm does not affect the number of leechers and seeds in steady-state, which now depends on the peers' download capacity c [13, 26]

$$\overline{x}_{i,s}^{\overline{\pi}} = \frac{\lambda_{i,s}}{c(1+\frac{\theta}{c})} \qquad \overline{y}_{i,s}^{\overline{\pi}} = \frac{\lambda_{i,s}}{\gamma(1+\frac{\theta}{c})}. \tag{13}$$

It is easy to verify that $\frac{\partial \overline{x}_{i,s}}{\partial \kappa_{i,s}} \leq 0$ and that $\frac{\partial^2 \overline{x}_{i,s}}{\partial \kappa_{i,s}^2} \geq 0$ for $\kappa_{i,s} \geq 0$, that is, the number of leechers in swarm s in ISP i in steady-state is a convex non-increasing function of the cache bandwidth allocated to swarm s in ISP i.

Given the functions $\overline{x}_{i,s}^{\pi}$ and $\overline{y}_{i,s}^{\pi}$ the steady-state optimal (*SSO*) bandwidth allocation policy can be formulated as

$$\overline{\pi}^* = \arg\min_{\kappa_i \in \mathcal{K}_i} \sum_{s\in\mathcal{S}} \bar{I}_{i,s}(\kappa_{i,s}), \tag{14}$$

where $\bar{I}_{i,s}(\kappa_{i,s})$ is the incoming inter-ISP traffic rate for the number of leechers and seeds in steady-state.

Since by assumption $I_{i,s}(z_s,\kappa_{i,s})$ is convex non-increasing in $\kappa_{i,s}$ for every state z_s, the steady-state optimal policy $\overline{\pi}^*$ can be found in a similar way as the OLA policy. The difference is that $\bar{I}_{i,s}(\kappa_{i,s})$ is a function of $\kappa_{i,s}$, $\overline{x}_{i,s}^{\pi}$ and $\overline{y}_{i,s}^{\pi}$, and the latter are themselves functions of $\kappa_{i,s}$. Note that the steady-state optimal policy $\overline{\pi}^*$ is not equivalent to the optimal policy π^* of the MDP, as the cache bandwidth allocated to a swarm s in ISP i would be nonzero even when $x_{i,s}(t) = 0$, which happens with nonzero probability.

5.4 Smallest-ratio Priority Allocation

The *SSO* policy exclusively focuses on the long term evolution of the process $\mathcal{Z}^{\pi}$ by minimizing the incoming inter-ISP traffic rate at steady state. It is time independent, i.e. it does not adapt to the current state of the system. The *OLA* policy instead, adapts to the system state by minimizing the instantaneous incoming inter-ISP traffic rate. Nevertheless, the *OLA* policy disregards how cache bandwidth allocation affects the long term evolution of the system.

In the following we use the incoming inter-ISP traffic model in [14] to derive an adaptive cache bandwidth allocation policy that approximates the SSO policy.

5.4.1 Incoming inter-ISP Traffic Model

We first reproduce the incoming inter-ISP traffic model for completeness. As the model is for a single swarm, we omit the subscript s for clarity. The model is based on two assumptions. First, leechers compete with each other for the available upload rate as long as they would be able to download at a higher rate. Second, given a single byte downloaded in ISP i, the distribution of its sources is proportional to the amount of upload rate exposed to the leechers that are located in ISP i.

The leechers in ISP i demand data at a total rate of cx_i. As the cache appears as an arbitrary peer to the leechers in ISP i, the demand is directed to the upload rate κ_i of ISP i's cache and to the publicly available upload rate $u_i^{PL} + \sum_{j \neq i} u_j^P$ of all ISPs. The leechers demand from the cache's upload rate with a probability proportional to its value, i.e, with probability $\kappa_i / (u_i^{PL} + \sum_{j \neq i} u_j^P + \kappa_i)$. The rest they demand from the publicly available upload rate, so the rate D_i^d that leechers in ISP i demand from the publicly available upload rate can be expressed as

$$D_i^d = cx_i \left(1 - \frac{\kappa_i}{u_i^{PL} + \sum_{j \neq i} u_j^P + \kappa_i} \right). \tag{15}$$

If the system is limited by the download rate of the leechers, then the leechers receive the demanded rate. If the system is limited by the available upload rate, then the rate at which the leechers receive is proportional to the total publicly available upload rate divided by the total demanded rate

$$D_i^r = D_i^d min \left(1, \frac{\sum_j u_j^P}{\sum_j D_j^d} \right). \tag{16}$$

The rate that the leechers receive can originate from any ISP. Using the assumption that for a single byte downloaded in ISP i, the distribution of its sources is proportional to the amount of upload rate exposed to leechers in ISP i we get the following estimate of the incoming inter-ISP traffic of ISP i

$$I_i(z_s, \kappa_i) = D_i^r \left(\frac{\sum_{j \neq i} u_j^P}{u_i^{PL} + \sum_{j \neq i} u_j^P} \right). \tag{17}$$

$I_i(z_s, \kappa_i)$ defined by (15) to (17) is a continuous convex non-increasing function of the cache bandwidth κ_i allocated by ISP i.

5.4.2 Smallest-ratio Priority Allocation

Our approximation of the *SSO* policy is based on the results in [13, 14], which show that the dynamics of swarm s in ISP i only depend on the aggregate arrival intensity of leechers $\sum_{j\neq i}\lambda_{j,s}$ and on the aggregate cache capacity $\sum_{j\neq i}\kappa_{j,s}$ in the rest of the ISPs. This observation allows us to focus on a single swarm spread over two ISPs, $\mathcal{I}=\{1,2\}$. ISP 1 is the tagged ISP and ISP 2 is the aggregation of all other ISPs in the network. We denote the ratio of the arrival rates in the two ISPs by $r=\lambda_2/\lambda_1$.

Our focus will be on how the partial derivative $\frac{\partial \bar{I}_1(\kappa_1)}{\partial \kappa_1}$ of the steady-state inter-ISP traffic depends on r. For small κ_1 the incoming inter-ISP traffic $I_1(z,\kappa_1)$ of ISP 1 defined by (15) to (17) can be approximated by

$$I_1(z,\kappa_1) \approx \frac{x_1}{x_1+x_2}u_2^P. \tag{18}$$

We consider the case when the system is limited by the available upload rate, so we substitute (10) and (11) into (18) to obtain an approximation of the steady-state incoming inter-ISP traffic $\bar{I}_1(\kappa_1)$ of ISP 1 as a function of the cache bandwidth. Consider now the derivative at $\kappa_1=0$ and $\kappa_2=0$

$$\frac{\partial \bar{I}_1(\kappa_1)}{\partial \kappa_1}\Big|_{\substack{\kappa_1=0\\ \kappa_2=0}} = -\frac{r^2(\gamma+\nu)(\gamma-\mu)-r\mu(\theta-\gamma)}{(1+\frac{\theta}{\nu})\mu\eta\gamma^2(1+r)^2}. \tag{19}$$

Recall that $\gamma-\mu>0$ is a necessary condition for the upload rate to be the limit, and it implies $\nu>0$ [13, 26]. Hence for $\theta-\gamma\leq 0$, (19) is negative and decreases monotonically in r.

For $\theta-\gamma>0$ we have to consider the mixed second order partial derivative at $\kappa_1=0$ and $\kappa_2=0$

$$\frac{\partial^2 \bar{I}_1(\kappa_1)}{\partial \kappa_1 \partial r}\Big|_{\substack{\kappa_1=0\\ \kappa_2=0}} = -\frac{2r(\gamma+\nu)(\gamma-\mu)+(r-1)\mu(\theta-\gamma)}{(1+\frac{\theta}{\nu})\mu\eta\gamma^2(1+r)^3}. \tag{20}$$

Since $\theta-\gamma>0$, (20) is negative for $r\geq 1$. Consequently allocating cache bandwidth to swarms with a higher ratio r of arrival rates leads to a faster decrease of the steady-state inter-ISP traffic. At the same time, due to the term $(1+r)^3$ in the denominator $\lim_{r\to\infty}\frac{\partial^2 \bar{I}_1(\kappa_1)}{\partial \kappa_1 \partial r}|_{\kappa_1=0}=0$, i.e, swarms with a high arrival ratio r provide approximately the same gain.

This approximation suggests that a priority-based policy that assigns the highest priority to the swarms with highest ratio $r=\lambda_2/\lambda_1$ would resemble the *SSO* allocation policy for small cache bandwidths. We use this insight to define the *smallest-ratio priority* (*SRP*) cache bandwidth allocation policy. Under *SRP* the priority of a swarm is calculated based on the instantaneous ratio of the local leechers to the number of peers in the overlay outside

of ISP i, $\hat{r}_{i,s} = \frac{x_{i,s}(t)}{\sum_{j \neq i} z_{j,s}(t)}$. The priority of swarms with $\hat{r}_{i,s} = 0$ and $\hat{r}_{i,s} = \infty$ is lowest, and the priorities of the remaining swarms are assigned in decreasing order of the ratios $\hat{r}_{i,s}$. and the priority is inverse proportional to $\hat{r}_{i,s}$ otherwise.

Practical Considerations

ISP i requires global information on the system state in order to compute any of the active cache bandwidth allocation policies presented above. The calculation of the *SSO* allocation relies on the incoming inter-ISP traffic rate for the number of peers in steady-state. In order to compute (10) and (11), ISP i needs to estimate the arrival rates of leechers to the different swarms, i.e. $\lambda_{i,s}$ $\forall s \in \mathcal{S}$, $\forall i \in \mathcal{I}$. The *OLA* policy assumes that the incoming inter-ISP traffic function $I_{i,s}(z_s, \kappa_{i,s})$ is known, furthermore it requires knowledge of the total number of peers in each swarm z_s $\forall s \in \mathcal{S}$. Similarly, under *SRP*, the priority of a swarm is calculated based on the number of local leechers $x_{i,s}$ and on the number of peers z_s in the overlay outside ISP i. In lack of interaction between ISPs and the P2P overlay, ISP i can collect information about the number of local peers in each swarm $z_{i,s}(t)$, and can estimate the aggregate number of peers outside its network $\sum_{j \neq i} z_{j,s}(t)$ by interrogating the tracker. If ISP-overlay interaction is possible [15], e.g., through an ALTO-like service [39], then ISP i could use the service to obtain information about the number of local and remote peers in each swarm. Cache bandwidth optimization could thus be a potential use-case for services like ALTO.

6 Performance Evaluation and Insights

In this section we use simulations and experiments to compare the three approximate cache bandwidth allocation policies to DDS, and to provide insight into the characteristics of an optimal cache bandwidth allocation policy.

6.1 Performance Evaluation Setup

In the following we describe the simulation and experimental settings and the implementation of the different cache bandwidth allocation policies. In both simulations and experiments we consider $I = 2$ ISPs, and use a BitTorrent seed with upload bandwidth K_1 as cache of ISP 1. The cache joins all swarms, but uploads only to leechers in ISP 1. The different cache bandwidth allocation policies are implemented in the peer.

6.1.1 Simulation setup

We used the P2P simulation and prototyping tool-kit ProtoPeer and the corresponding library for BitTorrent [40, 41] for the simulations. The simulations are flow-level: data transmissions are flows and the bandwidth for each flow is calculated according to the

max-min-fair-share principle [42], an approximation of the bandwidth sharing behavior of TCP.

We simulate 12 to 26 BitTorrent swarms, each sharing a file of 150MB. The number of swarms is large enough to show the impact of the policies. At the same time, it keeps the run-time of the simulations at a reasonable level of a few hours per simulation run. The peers have an access bandwidth of 1Mbit/s upstream and 16Mbit/s downstream. The peers join swarm s in ISP i according to a Poisson process and, after completing the download, they remain in the swarm for an exponentially distributed seeding time with average $1/\gamma =$ 10 minutes.

In order to implement the *OLA* and the *SSO* policies, we use the traffic model in Section 5.4. We implement the *SRP* in the simulator by assigning a priority level to every data flow and by modifying the bandwidth sharing algorithm. The bandwidth of flows with the same priority is calculated according to the original max-min-fair-share algorithm, while flows with a lower priority can only use the link bandwidth not used by flows with higher priority.

6.1.2 Experimental setup

We perform experiments involving approximately 500 Planet-lab nodes using BitTorrent 4.4.0. We scale down the file size, the upload rates and the download rates by a factor of 43 compared to the simulations in order to avoid interfering with other Planet-lab traffic: the file size is 3.5MB, and the upload and download bandwidths of the peers are 23kbit/s and 373kbit/s, respectively.

For every swarm we assign every Planet-lab node to one of the two ISPs, and measure the traffic exchanged between peers belonging to different ISPs. We use one peer per swarm as the cache of ISP 1; these 12 peers run on a dedicated Linux computer. We implement the cache bandwidth allocation policies using hierarchical token bucket (HTB) queues in Linux traffic control. We use one filter per swarm to redirect the upload traffic of the 12 peers to a HTB class that enforces the total cache upload bandwidth limit K_1. For the *SSO* and the *SRP* policies we attach to this class one subclass per swarm. By default each subclass has 500B/s of guaranteed bandwidth in order to keep the TCP connections alive. The actual priority and guaranteed bitrate are then set according to the cache bandwidth allocation policy. The excess bandwidth is distributed among the swarms as defined by the HTB queue. For *SRP* we update the priorities every 10 seconds based on the average number of leechers and seeds over the preceding 30 seconds.

6.2 Stationary Arrival Process

We start by considering the case when peers join swarm s in ISP i according to a stationary Poisson process at a rate of $\lambda_{i,s}$. This corresponds to a system in steady state. Every simulation run corresponds to 6.5 hours of simulated time, and we use the results following a warm-up period of 1.5 hours. For every configuration we show the average of 5 simulation

Scenario	Number of swarms (S)	Identical swarms (s)	$\frac{\lambda_s}{\lambda}$	$\frac{\lambda_{2,s}}{\lambda_{1,s}}$
unif.,1:10	12	1,..,12	1/12	10
zipf,1:10	12		$\propto \frac{1}{s}$	10
unif.,1:1+1:10	12	1,..,10	1/12	10
		11,12	1/12	1
het.,2:2+1:10	15	1,..,4	1/8	10
		5,..,15	1/22	1

Table 2: Relative peer arrival rates in the simulated scenarios.

runs together with the 95%-confidence intervals. Every experiment runs for 4 hours, and we use the results after an initial warm-up period of 1 hour.

6.2.1 Cache Bandwidth Allocation Matters

We simulate four scenarios to investigate under what conditions active cache bandwidth allocation can be beneficial. For simplicity, we denote the total arrival rate by $\lambda = \sum_i \sum_s \lambda_{i,s}$. We use the same total arrival rate $\lambda = 30$/min for all four scenarios, but the four scenarios differ in terms of the arrival rates $\lambda_{i,s}$ of the peers between swarms and between ISPs. Table 2 shows the relative arrival rates for the four scenarios. The ratio $\frac{\lambda_s}{\lambda}$ is related to the size of swarm s compared to all swarms, while $\frac{\lambda_{2,s}}{\lambda_{1,s}}$ is related to the share of local peers in swarm s. In [43], the authors measured the top-AS fraction of different swarms, defined as the maximum number of peers in one AS of the swarm normalized by the size of the swarm ($\max_i \frac{z_i}{\sum_j z_j}$). The top-AS fraction was found to vary from a minimum of slightly less than 0.1, for swarms sharing international content, to a maximum of 0.5, for swarms sharing regional content. We can use these numbers to obtain estimates of the relative arrival intensities of leechers as follows. In absence of a cache ($\kappa_i = 0 \; \forall i \in \mathcal{I}$), the numbers of both seeders and leechers are proportional to the arrival rate of leechers to swarms, whether the system is upload rate limited (10-11) or download rate limited (13). Consequently, by substituting $\kappa_i = 0 \; \forall i \in \mathcal{I}$ in (10-13), we can use the top-AS fraction to calculate the ratio $\frac{\lambda_{2,s}}{\lambda_{1,s}}$. Using this approximation a top-AS fraction of 0.1 corresponds to $\frac{\lambda_{2,s}}{\lambda_{1,s}}$ slightly greater than 9, and a top-AS fraction of 0.5 corresponds to $\frac{\lambda_{2,s}}{\lambda_{1,s}} = 1$. Given these approximate relative arrival intensities, our evaluation scenarios are constructed so that they allow us to isolate the factors that influence the efficiency of cache bandwidth allocation policies.

As an example, in scenario *unif.,1:1+1:10* all $S = 12$ swarms have the same arrival rate $\lambda_s = \lambda/12$. The arrival rates for swarms 1 to 10 are asymmetric ($\lambda_{2,s} = 10\lambda_{1,s}$), while for swarms 11 and 12 they are symmetric ($\lambda_{2,s} = \lambda_{1,s}$). In scenario *het.,2:2+1:10* the swarms have different arrival rates. 4 out of 15 swarms have an arrival rate of $\lambda_s = \lambda/8$, and are

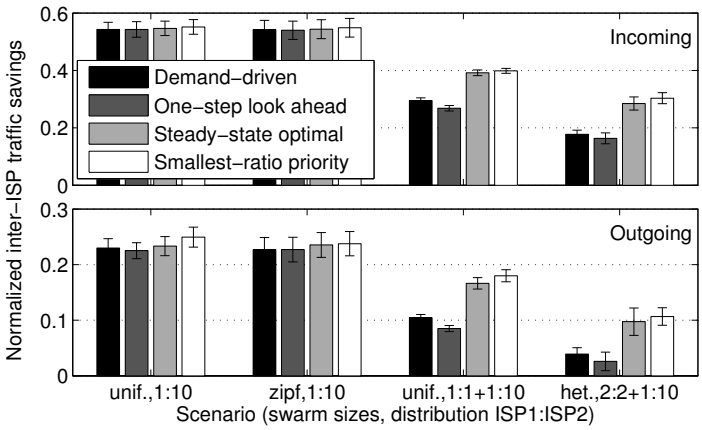

Figure 1: Incoming and outgoing inter-ISP traffic savings for the four scenarios and four policies for $K_1 = 30$Mbit/s. Simulation results.

asymmetric, $\lambda_{2,s} = 10\lambda_{1,s}$. The remaining 11 swarms have an arrival rate of $\lambda_s = \lambda/22$ and are symmetric $\lambda_{2,s} = \lambda_{1,s}$. Compared to *unif.,1:1+1:10*, in this scenario the symmetric swarms, though more popular in ISP 1, are less popular in total than the asymmetric ones. The use of Zipf's law for the arrival intensities in scenario *zipf,1:10* is motivated by recent measurements that show that the distribution of the number of concurrent peers over swarms exhibits Zipf like characteristics over a wide range of swarm sizes [44, 45]. Symmetric and asymmetric swarms are motivated by measurements that show the difference in terms of the spatial distribution of peers between contents of regional and of global interest (e.g., the popularity of movies depending on the language [45]).

Fig. 1 shows the normalized incoming and outgoing inter-ISP traffic saving of ISP 1 for the four scenarios for the *DDS*, *OLA*, *SSO* and *SRP* allocation policies. We calculate the normalized inter-ISP traffic saving as the decrease of the average inter-ISP traffic due to installing a cache divided by the average inter-ISP traffic without a cache ($K_1 = 0$), that is, $(C_i|_{K_1=0} - C_i^{\pi})/C_i|_{K_1=0}$. The upload bandwidth of the cache in ISP 1 is $K_1 = 30$Mbit/s.

For the *unif.,1:10* and the *zipf,1:10* scenarios, in which the ratio $\lambda_{2,s}/\lambda_{1,s} = 10$ is the same for all swarms, the difference between the results for the different cache bandwidth allocation policies is within the confidence interval. However, for the scenarios *unif.,1:1+1:10* and *het.,2:2+1:10* the bandwidth allocation policies make a significant difference in terms of traffic savings, both in terms of incoming and outgoing inter-ISP traffic. These results indicate that cache bandwidth allocation affects the inter-ISP traffic savings when the distribution of the peers over the ISPs is different among swarms, as for the *unif.,1:1+1:10* and the *het.,2:2+1:10* scenarios.

A comparison of the different policies in Fig. 1 for the *unif.,1:1+1:10* scenario reveals

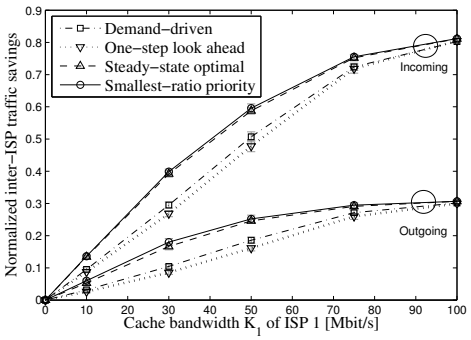

Figure 2: Incoming and outgoing inter-ISP traffic saving for the *unif.,1:1+1:10* scenario vs. cache bandwidth in ISP 1. Simulation results.

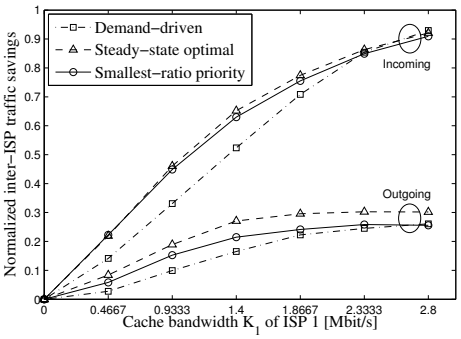

Figure 3: Incoming and outgoing inter-ISP traffic saving for the *unif.,1:1+1:10* scenario vs. cache bandwidth in ISP 1. Experiment results.

that the effect of the *OLA* policy on the inter-ISP traffic saving is opposite to the effect of the *SSO* and the *SRP* policies. The *OLA* policy performs worse than *DDS*, but the *SSO* and *SRP* policies compared to *DDS* drastically increase the incoming and outgoing inter-ISP traffic savings. For the *SSO* policy, the incoming inter-ISP traffic savings increase by about 33 percent and the outgoing inter-ISP traffic savings by over 60 percent. For the *het.,2:2+1:10* scenario the savings increase by 60 and 150 percent, respectively. The *SRP* policy achieves even better gains. For the *het.,2:2+1:10* scenario for example, the savings increase by 71 percent for the incoming inter-ISP traffic and 172 percent for the outgoing traffic. Considering that P2P cache eviction policies achieve within 10 to 20 percent of the hit rate of the optimal off-line eviction policy [10, 11], the 30 to 70 percent decrease of the incoming inter-ISP traffic achieved through cache bandwidth allocation is more than what could be achieved through improved cache eviction policies.

6.2.2 Inter-ISP traffic savings

Fig. 2 shows the incoming and outgoing inter-ISP traffic savings normalized by the inter-ISP traffic without cache ($K_1 = 0$) for the *unif.,1:1+1:10* scenario, as a function of the cache bandwidth K_1. The figure confirms that the observations made in Fig. 1 hold for a wide range of cache bandwidths K_1. Only above $K_1 \approx 75$Mbit/s, when the available upload bandwidth in ISP 1 exceeds the aggregate download bandwidth of the leechers within ISP 1, the marginal traffic saving diminishes and so does the difference between the policies. We note that the *SRP* policy performs slightly better than the *SSO* policy for all cache bandwidths. This is because the the *SSO* allocation can be far from optimal when the instantaneous number of peers in the system is far away from the steady-state average number of peers. We show the corresponding experimental results in Fig. 3. We

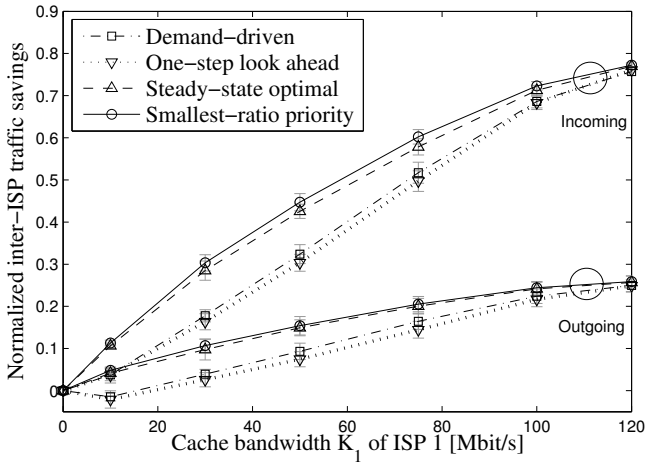

Figure 4: Incoming and outgoing inter-ISP traffic savings for the *het.,2:2+1:10* scenario vs. cache bandwidth in ISP 1. Simulation results.

omit the results for the *OLA* policy since it performed poorly in all simulated scenario. As shown in Fig. 3, the experimental results match the simulation results (cf. Fig. 2) and confirm the significant gain of cache bandwidth allocation observed in the simulations. The only difference is that the *SRP* policy performs slightly worse than in the simulations, which is due to the impact of the network layer implementation of bandwidth allocation and priorities on TCP congestion control. An application layer implementation of the policy could prevent this.

Fig. 4 shows the incoming and outgoing inter-ISP traffic saving normalized by the inter-ISP traffic without cache ($K_1 = 0$) for the *het.,2:2+1:10* scenario as a function of the cache bandwidth K_1. The figure allows us to draw similar conclusions as Fig. 2, except for the dip in the outgoing inter-ISP traffic saving for *DDS* at $K_1 = 10$Mbit/s. While surprising at first sight, the potential increase of the outgoing inter-ISP traffic due to caching for small, symmetric swarms (i.e., swarms 5 to 15) was pointed out in [13]. Since the *SRP* and the *SSO* policies allow little cache bandwidth to be used by the symmetric swarms for low K_1, they provide outgoing inter-ISP traffic savings even at $K_1 = 10$Mbit/s.

6.2.3 Cache Upload Rate to Swarms

In order to understand how the different policies allocate bandwidth to the different swarms, we show two indifference maps of ISP 1 for the *unif.,1:1+1:10* scenario in Fig. 5 and Fig. 6. The horizontal and the vertical axes show the cache bandwidth allocated to each of the 10 asymmetric ($\lambda_{2,s} = 10\lambda_{1,s}$) and to each of the 2 symmetric ($\lambda_{2,s} = \lambda_{1,s}$) swarms, respec-

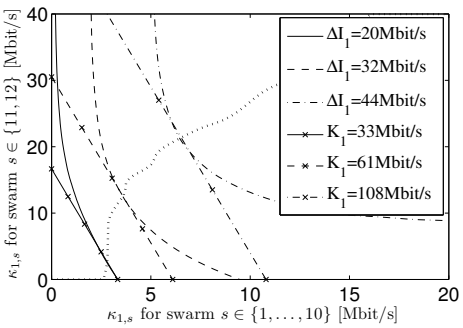

Figure 5: Indifference map of ISP 1 for the *unif.,1:1+1:10* scenario based on simulation results. The dotted line shows the *SSO* cache bandwidth allocation for different values of cache bandwidth K_1.

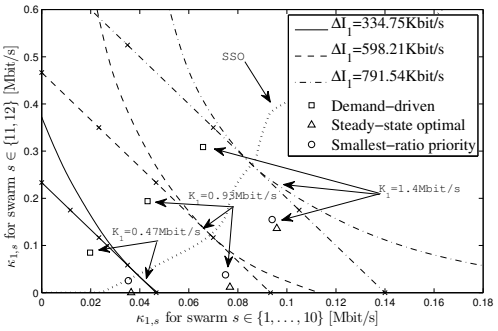

Figure 6: Indifference map of ISP 1 for the *unif.,1:1+1:10* scenario based on the experiments, and the actual average cache upload rates for the *DDS*, *SSO* and *SRP* policies. Experiment results.

tively. The curves show combinations of bandwidth allocations that lead to a particular inter-ISP traffic saving ΔI_1 (i.e., was there no cache bandwidth constraint K_1, ISP 1 would be indifferent between allocations on the same indifference curve). The straight diagonal lines show different cache bandwidth constraints K_1. The *SSO* cache bandwidth allocation for K_1 is given by the coordinates of the point at which the cache bandwidth constraint line for K_1 is tangent to the indifference curve. The dotted line connects all such points: it shows the *SSO* cache bandwidth allocation for different K_1.

Fig. 5 shows the indifference map based on simulation results. We note that for $K_1 \leq$ 30Mbit/s all cache bandwidth should be allocated to the 10 asymmetric swarms, above that, as K_1 increases so does the bandwidth that should be allocated to the 2 symmetric swarms. We also note that the shape of the indifference curves confirms that the inter-ISP traffic saving for a single swarm is a concave non-decreasing function of $\kappa_{i,s}$.

Fig. 6 shows the indifference map and the actual average cache upload rate received by the asymmetric (horizontal) and the symmetric (vertical) swarms under the three allocation policies, based on experiment results. There is one marker per policy and total cache bandwidth K_1. The figure shows how the cache upload rate received by the swarms differs under the three policies depending on the cache bandwidth limit K_1. Under both *SRP* and *SSO* the cache uploads to the symmetric swarms at a significantly lower rate than under *DDS* except for very high K_1. which is the key to the higher inter-ISP traffic savings of both policies.

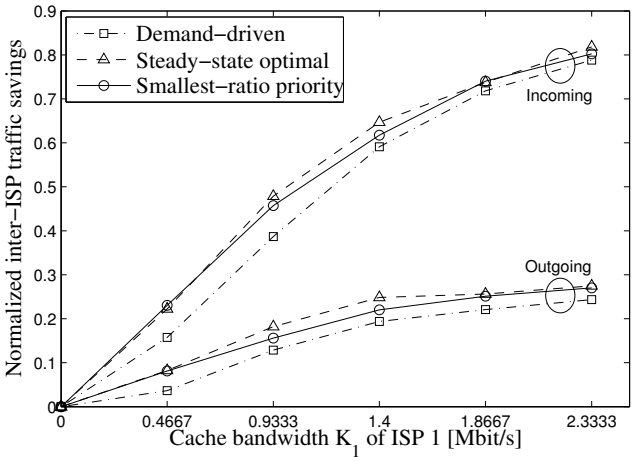

Figure 7: Incoming and outgoing inter-ISP traffic saving for the non-stationary scenario vs. cache bandwidth in ISP 1. Experiment results.

6.3 Non-Stationary Arrival Process

So far we only looked at a system in steady state. In order to investigate the robustness of the bandwidth allocation policies to the system dynamics, we now turn to the case of a non-stationary arrival process. We consider that the leechers join swarm s in ISP i according to a non-stationary Poisson process with rate

$$\lambda_{i,s}(t) = \lambda_{i,s}^0 e^{-\frac{t}{\tau}}, \tag{21}$$

where $\lambda_{i,s}^0$ is the initial arrival rate and τ is the attenuation parameter of peer arrival rate. (21) has been shown to be a good model of the peer arrival rate during the swarm's lifespan in [46, 47]. To derive the inter-arrival times of peers to swarm s, we simulated the non-stationary arrival process using the thinning method by Lewis and Schedler [48]. We considered $\sum_{i\in\mathcal{I}} \lambda_{i,s}^0 = \frac{1}{8}$ and $\tau = 4000$, which result in a swarm lifespan of about 5 hours, and a total peer population during the lifespan of 500 peers. We performed experiments starting a new swarm every 15 minutes, for 6.5 hours. The swarms starting at 0h, 2h, 4h, 6h are symmetric ($\lambda_{2,s} = \lambda_{1,s}$), while the rest of the swarms are asymmetric ($\lambda_{2,s} = 10\lambda_{1,s}$).

Fig. 7 shows the incoming and outgoing inter-ISP traffic saving normalized by the inter-ISP traffic without cache ($K_1 = 0$) for the non-stationary scenario described above, as a function of the cache bandwidth K_1. The inter-ISP traffic savings show a similar trend as under the stationary arrival process (c.f., Fig. 3). Comparing Figures 7 and 3, we observe that the benefit of cache bandwidth allocation is slightly reduced, although still significant: the *SSO* and the *SRP* policies achieve savings in the order of 20 to 30 percent compared to

the *DD* allocation. It is important to note that the computation of the *SSO* policy and the derivation of the *SRP* policy in Section 5.4 assume that the system is in steady-state, yet the policies provide significant savings in a non-stationary system.

7 Conclusion

Motivated by the large amount of inter-ISP P2P traffic, we investigated a new dimension of P2P cache resource management, the allocation of cache upload bandwidth between overlays. We formulated the problem of cache bandwidth allocation as a Markov decision process, and showed the existence of an optimal stationary allocation policy. Based on insights obtained from the model, we proposed three bandwidth allocation policies to approximate the optimal allocation policy. We performed simulations and experiments to evaluate the performance of the proposed policies. We demonstrated the importance of capturing the cache's impact on the swarm dynamics for cache bandwidth allocation. We identified the heterogeneity of the swarm's distribution between ISPs as the primary factor that influences the potential traffic savings through cache bandwidth allocation. Our results show that the proposed smallest ratio priority policy can decrease the amount of inter-ISP traffic between 30 to 60 percent, which is significantly higher than what could potentially be achieved exclusively through improved peer-to-peer cache eviction policies.

8 Acknowledgement

The authors thank Alexandre Proutiere for his comments on the drafts of the conference version of this work.

References

[1] H. Schulze and K. Mochalski, "Internet Study 2008/2009," 2009. [Online]. Available: http://www.ipoque.com/resources/internet-studies

[2] P. Eckersley, F. von Lohmann, and S. Schoen, "Packet forgery by ISPs: A report on the Comcast affair," White paper, Nov. 2007.

[3] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPs and P2P systems co-operate for improved performance?" *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 3, pp. 29–40, Jul. 2007.

[4] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: Provider portal for applications," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 351–362, Oct. 2008.

[5] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang, "Improving traffic locality in BitTorrent via biased neighbor selection," in *Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS)*, Jul. 2006, pp. 66–75.

[6] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-ISP traffic in peer-to-peer systems," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 363–374, Oct. 2008.

[7] S. L. Blond, A. Legout, and W. Dabbous, "Pushing bittorrent locality to the limit," *Computer Networks*, vol. 55, no. 3, pp. 541–557, 2011.

[8] OverCache P2P, "http://www.oversi.com."

[9] PeerApp UltraBand, "http://www.peerapp.com."

[10] A. Wierzbicki, N. Leibowitz, M. Ripeanu, and R. Woźniak, "Cache replacement policies for P2P file sharing protocols," *Euro. Trans. on Telecomms.*, vol. 15, pp. 559–569, Nov. 2004.

[11] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1447–1460, Dec. 2008.

[12] N. Megiddo and D. Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache," in *Proc. of USENIX File & Storage Technologies Conference (FAST)*, 2003, pp. 115 – 130.

[13] F. Lehrieder, G. Dán, T. Hoßfeld, S. Oechsner, and V. Singeorzan, "The impact of caching on BitTorrent-like peer-to-peer systems," in *Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P)*, Aug. 2010.

[14] F. Lehrieder, G. Dan, T. Hossfeld, S. Oechsner, and V. Singeorzan, "Caching for BitTorrent-Like P2P Systems: A Simple Fluid Model and Its Implications," *IEEE/ACM Trans. Netw.*, vol. 20, no. 4, pp. 1176–1189, 2012.

[15] G. Dán, T. Hoßfeld, S. Oechsner, P. Cholda, R. Stankiewicz, I. Papafili, and G. Stamoulis, "Interaction patterns between P2P content distribution systems and ISPs," *IEEE Communications Magazine*, vol. 49, no. 5, pp. 222–230, May 2011.

[16] S. Ren, E. Tan, T. Luo, L. Guo, S. Chen, and X. Zhang, "TopBT: A topology-aware and infrastructure-independent BitTorrent," in *Proc. IEEE INFOCOM*, Apr. 2011.

[17] N. Leibowitz, A. Bergman, R. Ben-Shaul, and A. Shavit, "Are file swapping networks cacheable? Characterizing P2P traffic," in *Proc. Int'l Workshop on Web Content Caching and Distribution (WCW)*, Aug. 2002.

[18] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should internet service providers fear peer-assisted content distribution?" in *Proc. of ACM SIGCOMM IMC*, 2005, pp. 63–76.

[19] G. Dán, "Cooperative caching and relaying strategies for peer-to-peer content delivery," in *Proc. Int'l Workshop on Peer-to-Peer Systems (IPTPS)*, 2008.

[20] J. Dai, B. Li, F. Liu, B. Li, and H. Jin, "On the efficiency of collaborative caching in ISP-aware P2P networks," in *Proc. IEEE INFOCOM*, Apr. 2011.

[21] I. Papafili, G. D. Stamoulis, F. Lehrieder, B. Kleine, and S. Oechsner, "Cache capacity allocation to overlay swarms," in *International Workshop on Self-Organizing Systems*, Feb. 2011.

[22] R. S. Peterson and E. G. Sirer, "Antfarm : Efficient Content Distribution with Managed Swarms," in *Proc. of NSDI*, 2009, pp. 107–122.

[23] R. S. Peterson, B. Wong, and E. G. Sirer, "A Content Propagation Metric for Efficient Content Distribution," in *Proc. of ACM SIGCOMM*, vol. 41, no. 4, New York, New York, USA, 2011, pp. 326–337.

[24] A. Sharma, A. Venkataramani, and A. A. Rocha, "Pros & Cons of Model-based Bandwidth Control for Client-assisted Content Delivery," *CoRR*, vol. abs/1209.5, 2012.

[25] X. Yang and G. de Veciana, "Service capacity of peer to peer networks," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2242–2252.

[26] D. Qiu and R. Srikant, "Modeling and performance analysis of BitTorrent-like peer-to-peer networks," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 367–378.

[27] F. Clévenot-Perronnin, P. Nain, and K. W. Ross, "Multiclass P2P networks: Static resource allocation for service differentiation and bandwidth diversity," *Performance Evaluation*, vol. 62, pp. 32–49, Oct. 2005.

[28] Y. Tian, D. Wu, and K. W. Ng, "Modeling, analysis and improvement for BitTorrent-like file sharing networks," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–11.

[29] I. Rimac, A. Elwalid, and S. Borst, "On server dimensioning for hybrid P2P content distribution networks," in *Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P)*, Sep. 2008, pp. 321–330.

[30] R. Izhak-Ratzin, H. Park, and M. van der Schaar, "Online Learning in BitTorrent Systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 12, pp. 2280–2288, 2012.

[31] E. J. Friedman, J. Y. Halpern, and I. Kash, "Efficiency and Nash Equilibria in a Scrip System for P2P Networks," in *Proc. of ACM EC*, 2006, pp. 140–149.

[32] H. Park and M. V. D. Schaar, "A Framework for Foresighted Resource Reciprocation in P2P Networks," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 101–116, 2009.

[33] R. Izhak-Ratzin, H. Park, and M. van der Schaar, "Reinforcement Learning in BitTorrent Systems," in *Proc. of IEEE INFOCOM*, 2011, pp. 406–410.

[34] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurement, analysis, and modeling of BitTorrent-like systems," in *Proc. ACM Internet Measurement Conf. (IMC)*, Oct. 2005, pp. 35–48.

[35] X. Guo and O. Hernandez-Lerma, *Continuous-time Markov Decision Processes*, ser. Springer Series in Stochastic Modelling and Applied Probability. Springer, 2009.

[36] R. L. Tweedie, "Criteria for ergodicity, exponential ergodicity and strong ergodicity of markov processes," *J. Appl. Prob.*, vol. 18, pp. 122–130, 1981.

[37] R. W. Wolff, "Poisson arrivals see time averages," *Operations Research*, vol. 30, no. 2, pp. 223–231, 1982.

[38] N. Z. Shor, *Minimization Methods for Non-differentiable Functions*, ser. Springer Series in Computational Mathematics. Springer, 1985.

[39] M. Stiemerling, S. Kiesel, S. Previdi, and M. Scharf, "ALTO Deployment Considerations," Internet Engineering Task Force (IETF), Internet-Draft, 2013.

[40] "Protopeer," http://protopeer.epfl.ch/index.html.

[41] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "ProtoPeer: A P2P toolkit bridging the gap between simulation and live deployment," in *Proc. International Conference on Simulation Tools and Techniques*, Mar. 2009.

[42] D. Bertsekas and R. Gallagher, *Data Networks*. Prentice Hall, 1987.

[43] T. Hoß feld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel, "Characterization of BitTorrent Swarms and their Distribution in the Internet," *Computer Networks*, vol. 55, no. 5, pp. 1197–1215, 2011.

[44] G. Dán and N. Carlsson, "Power-law revisited: A large scale measurement study of P2P content popularity," in *Proc. Int'l Workshop on Peer-to-Peer Systems (IPTPS)*, April 2010.

[45] T. Hoßfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel, "Characterization of BitTorrent swarms and their distribution in the Internet," *Computer Networks*, vol. 55, no. 5, Apr. 2011.

[46] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "A Performance Study of BitTorrent-like Peer-toPeer Systems," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 1, pp. 155–169, 2007.

[47] N. Carlsson, G. Dán, A. Mahanti, and M. Arlitt, "A Longitudinal Characterization of Local and Global BitTorrent Workload Dynamics," in *Proc. of Passive and Active Measurement Conference (PAM)*, 2012, pp. 252–262.

[48] P. Lewis and G. Shedler, "Simulation of Nonhomogeneous Poisson Processes by Thinning," *Nav. Res. Logist. Q.*, vol. 26, pp. 403 – 413, 1979.

Paper B

Convergence in Player-Specific Graphical Resource Allocation Games

Valentino Pacifici and György Dán

in IEEE Journal on Selected Areas in Communications, December 2012.

Convergence in Player-Specific Graphical Resource Allocation Games

Valentino Pacifici and György Dán *

Abstract

As a model of distributed resource allocation in networked systems, we consider resource allocation games played over a influence graph. The influence graph models limited interaction between the players due to, e.g., the network topology: the payoff that an allocated resource yields to a player depends only on the resources allocated by her neighbors on the graph. We prove that pure strategy Nash equilibria (NE) always exist in graphical resource allocation games and we provide a linear time algorithm to compute equilibria. We show that these games do not admit a potential function: if there are closed paths in the influence graph then there can be best reply cycles. Nevertheless, we show that from any initial allocation of a resource allocation game it is possible to reach a NE by playing best replies and we provide a bound on the maximal number of update steps required. Furthermore we give sufficient conditions in terms of the influence graph topology and the utility structure under which best reply cycles do not exist. Finally we propose an efficient distributed algorithm to reach an equilibrium over an arbitrary graph and we illustrate its performance on different random graph topologies.

1 Introduction

Resource allocation is a fundamental problem in communication systems. In general, resource allocation problems are characterized by a set of nodes competing for the same set of resources, and arise in a wide variety of contexts, from congestion control, through content management [1, 2] to dynamic channel allocation for opportunistic spectrum access [3, 4]. The nodes competing for the resources are often autonomous entities, such as end hosts or mobile nodes, and therefore the resource allocation problem has to be solved in a decentralized manner. Furthermore, the solution should be compatible with the nodes' own interests.

*The authors are with the Laboratory for Communication Networks, School of Electrical Engineering, and the ACCESS Linnaeus Centre, KTH Royal Institute of Technology. Osquldas väg 10, 10044 Stockholm. Tel.: +46 87904251. E-mail: {pacifici,gyuri}@kth.se. Manuscript received December 15, 2011; revised May 25, 2012; accepted July 15, 2012.

Congestion games are widely used as a game theoretic model of distributed resource allocation [5]. In congestion games a set of players allocate resources in order to maximize their own utility, but the utility provided by an individual resource to a player is a function of how many other players have allocated the resource. In the case of networked systems, the interactions between the players are often limited by the network topology, which can be captured by introducing an influence graph in the model of congestion games [6, 7, 8]. Under certain conditions of symmetry, e.g., homogeneous or linear utility functions [6, 7, 8, 9, 10, 11, 12], congestion games allow a potential function [13], and thus the two most fundamental questions, the existence of pure Nash equilibria and the convergence of myopic learning rules to the equilibria are answered. For congestion games that do not admit a potential function, the answer to these two fundamental questions is, in general, not known [14, 15, 16].

In this work we consider a class of resource allocation games that gives rise to a graphical player-specific congestion game that does not admit a potential function. In the model we consider, the player-specific utility of a resource to a player is amortized if any of its neighbors allocates the same resource. This model captures problems arising in a number of fields: object placement in the context of CPU caching in computer architectures [17], the placement of contents in clean-slate information centric network architectures [1], the problem of cooperative caching between Internet service providers [2], the distributed allocation of radio spectrum to radio transmitters [3, 4], and the distributed scheduling of jobs [18].

Our contributions are the following. We show that Nash equilibria exist in graphical resource allocation games for arbitrary influence graphs and payoff structures, and give a bound on the complexity of finding equilibria. We show that nodes might cycle arbitrarily long before reaching equilibrium if the influence graph is non-complete, thus the game does not admit a potential function. We then provide sufficient conditions in terms of the graph topology and the payoff structure that guarantee that cycles do not exist if players play best replies. Furthermore, we give a sufficient condition under which a simple and efficient distributed algorithm can be used to reach an equilibrium, and illustrate the efficiency of the algorithm with numerical results.

The rest of the paper is structured as follows. We define graphical resource allocation games in Section 2, and provide results for arbitrary graph topologies in Section 3. We then analyze the convergence to equilibria for complete graphs in Section 4 and for specific payoff structures in Section 5. Section 6 introduces the distributed algorithm for reaching equilibrium. In Section 7 we discuss related work, and Section 8 concludes the paper.

2 The Resource Allocation Game

In the following we formulate the problem of graphical resource allocation as a non-cooperative game played on a graph. We consider a set of nodes N and a set of resources $\mathcal{R}$. Every node is located at a vertex of an undirected graph $\mathcal{G} = (N, E)$, called the influence graph. We denote the set of neighbors of node i by $\mathcal{N}(i)$, i.e., $\mathcal{N}(i) = \{j|(i,j) \in E\}$. Each node i allocates $K_i \in \mathbb{N}_+$ different resources: we describe the set of resources allocated by node i with the $|\mathcal{R}|$ dimensional vector $a_i = (a_i^1, \ldots, a_i^{|\mathcal{R}|})$, whose component $a_i^r \in \{0,1\}$ is 1 if resource r is allocated by node i. Thus the set of feasible resource allocation vectors for node i is $\mathcal{A}_i = \{a_i | \sum_r a_i^r \leq K_i\} \subseteq \{0,1\}^{|\mathcal{R}|}$. To ease notation we say that a resource $r \in \mathcal{R}$ is *i-busy* if it is allocated by at least one of node i's neighbors and we define $\pi_i^r(a_{-i}^r) \triangleq \prod_{j \in \mathcal{N}(i)} (1 - a_j^r) = 0$, otherwise we say that resource $r \in \mathcal{R}$ is *i-free*.
We call c_{ir} the value of resource r for node i. The payoff $U_i^r(1, a_{-i}^r)$ that a node i gets from allocating a resource r is influenced by the resource allocation of its neighboring nodes $\mathcal{N}(i)$. A node i allocating resource r gets as payoff the value c_{ir} if resource r is *i-free*. If resource r is *i-busy*, node i gets payoff $\delta_i c_{ir}$, where δ_i is the cost of sharing for node i, $0 \leq \delta_i < 1$,

$$U_i^r(1, a_{-i}^r) = \begin{cases} c_{ir} & \text{if } \pi_i^r(a_{-i}^r) = 1 \\ \delta_i c_{ir} & \text{if } \pi_i^r(a_{-i}^r) = 0. \end{cases} \quad (1)$$

A node does not get any payoff from the resources it does not allocate, i.e, $U_i^r(0, a_{-i}^r) = 0$.
We model the resource allocation problem as a strategic game

$$\Gamma =< N, (\mathcal{A}_i)_{i \in N}, (U_i)_{i \in N} >,$$

where the utility function of player i is the sum of the payoffs $U_i(a_i, a_{-i}) = \sum_r U_i^r(a_i^r, a_{-i}^r)$. Note that the influence graph influences the payoffs that player i gets from the allocated resources via the neighbor set $\mathcal{N}(i)$, i.e., the utility of player i is entirely specified by the actions of players $j \in \mathcal{N}(i)$. Under the assumption that every player i allocates always K_i resources, the graphical resource allocation game we consider is a graphical player-specific matroid congestion game.
In the following we outline two applications of graphical resource allocation games.

2.1 Selfish Object Replication on Graphs

Consider a set N of nodes and a set $\mathcal{R}$ of objects of unit size. The demand for object $r \in \mathcal{R}$ at node $i \in N$ is given by the rate $w_i^r \in \mathbb{R}_+$. Node i has integer storage capacity K_i which it uses to replicate objects locally. The marginal cost of serving requests for object r in node i is α_i if the object is replicated in node i, it is β_i if the object is replicated in a node $j \in \mathcal{N}(i)$ neighboring i, and it is γ_i

otherwise. It is reasonable to consider that it is not more costly to access a resource replicated locally than one replicated at a neighbor, and it is less costly to access a resource replicated at a neighbor than retrieving it directly from the common set of resources. Formally $\alpha_i \leq \beta_i < \gamma_i$. The cost of node i due to object r is proportional to the demand w_i^r, and is a function of a_i and the replication states a_{-i} of the neighboring nodes, and its total cost is

$$\begin{aligned} C_i(a_i, a_{-i}) = &\quad \textstyle\sum_{r\in\mathcal{R}} C_i^r(a_i^r, a_{-i}^r) \\ = &\quad \textstyle\sum_{r\in\mathcal{R}} w_i^r \Big[\alpha_i a_i^r + (1-a_i^r)\left[\gamma_i \pi_i^r(a_{-i}^r) + \beta_i(1-\pi_i^r(a_{-i}^r))\right]\Big] \end{aligned} \tag{2}$$

The goal of node i is to choose a replication strategy a_i^* that minimizes its total cost given the strategy profile a_{-i} of the other nodes. Observe that the cost of object r for node i can be expressed as

$$C_i^r(a_i^r, a_{-i}^r) = C_i^r(0, a_{-i}^r) - \big(C_i^r(0, a_{-i}^r) - C_i^r(a_i^r, a_{-i}^r)\big) = C_i^r(0, a_{-i}^r) - CS_i^r(a_i^r, a_{-i}^r),$$

where $CS_i^r(a_i^r, a_{-i}^r)$ is the cost saving that node i achieves through object r given the other nodes' replication strategies. Since the cost $C_i^r(0, a_{-i}^r)$ is independent of the action a_i^r of node i, finding the minimum cost is equivalent to maximizing the aggregated cost saving

$$\begin{aligned} \arg\min_{a_i} C_i(a_i, a_{-i}) &= \arg\min_{a_i} \sum_r C_i^r(a_i^r, a_{-i}^r) \\ &= \arg\min_{a_i} \Big(\sum_r C_i^r(0, a_{-i}^r) - \sum_r CS_i^r(a_i^r, a_{-i}^r) \Big) \\ &= \arg\max_{a_i} \sum_r CS_i^r(a_i^r, a_{-i}^r). \end{aligned}$$

We can express the cost saving $CS_i^r(a_i^r, a_{-i}^r)$ of node i by substituting (2)

$$\begin{aligned} CS_i^r(a_i^r, a_{-i}^r) &= w_i^r \left[\beta_i(1-\pi_i^r(a_{-i}^r)) + \gamma_i \pi_i^r(a_{-i}^r)\right] \\ &\quad - w_i^r \left(\alpha_i a_i^r + (1-a_i^r)\left[\beta_i(1-\pi_i^r(a_{-i}^r)) + \gamma_i \pi_i^r(a_{-i}^r)\right]\right) \\ &= a_i^r w_i^r \left[\beta_i(1-\pi_i^r(a_{-i}^r)) + \gamma_i \pi_i^r(a_{-i}^r) - \alpha_i\right]. \end{aligned}$$

Thus $CS_i^r(0, a_{-i}^r) = 0$. Then, by defining $\delta_i \triangleq \frac{\beta_i - \alpha_i}{\gamma_i - \alpha_i}$ and $w_i^r\left[\gamma_i - \alpha_i\right] = c_{ir}$, we obtain that $CS_i^r(1, a_{-i}^r) = U_i^r(1, a_{-i}^r)$ defined in (1). This model of object replication was used, for example, in [17] to model distributed cache allocation in a computer architecture. It was used to model cooperative caching of contents among Internet Service Providers (ISPs) in [2], where the ISPs are neighbors if they have a peering agreement, and the costs γ_i, β_i, and α_i correspond to the cost of downloading contents over transit, peering and local links, respectively. The model also captures the cost structure of cooperative caching for multiview video streaming systems considered in [19], in which case the costs correspond to the access of image partitions from a remote repository, remote servers and local servers, respectively.

2.2 Graph Multi-coloring, Distributed Radio Spectrum Allocation and Medium Access Control

Proper graph multi-coloring is a generalization of the proper graph coloring problem. Given a graph $\mathcal{G} = (N, E)$ and a set $\mathcal{R}$ of colors the task is to assign K_i distinct colors to vertex $i \in N$ such that no adjacent vertices have the same color. In general, the goal is to use as few as possible colors. In the case of an improper coloring the same color can be assigned to adjacent vertices, but this involves a penalty δ_i for vertex i.

Graph multi-coloring has a number of applications. In the case of scheduling the nodes are jobs, the colors are time units, and K_i is the time needed to finish job i. The minimum number of colors is then the makespan of all jobs [18]. In the case of medium access control there is a set N of nodes that contend for some radio channels or for time slots [3, 4]. The influence graph $\mathcal{G}$ models the potential conflicts due to co-channel interference that can occur between nodes. The model of an undirected influence graph is appropriate for a system in which the pairs of nodes that communicate to each other are relatively close to each other, and fast fading is not a dominant fading component. In the case of channel assignment, each node $i \in N$ has K_i radio interfaces that it can use to transmit data. Alternatively, in the case of time slot allocation, each node $i \in N$ can use K_i time slots. The throughput that node i can achieve transmitting on resource $r \in \mathcal{R}$ in the absence of interference is c_{ir}. In the case of interference the expected throughput drops to $\delta_i c_{ir}$, where δ_i is the resource deterioration expected by node i under interference. In some applications a node can represent an entire network. In the case of coexisting Zigbee PRO networks, for example, the coordinator of each network $i \in N$ is in charge of selecting the channel for the entire network, so as to minimize the expected interference with other networks.

3 Existence of Equilibria and Convergence

We start with addressing two fundamental questions. First, we address the question whether in the graphical resource allocation problem there exist equilibrium resource allocations, from which no player would like to deviate unilaterally. Second, we address the question whether the players could reach an equilibrium state in a distributed manner.

3.1 Existence of equilibria

In order to formalize equilibrium allocations in the resource allocation game and to formulate our results, we start with the definition of three key terms used throughout the section.

Definition 1. A *best reply* of player i is a best strategy a_i^* of player i given the other players' strategies, that is, $U_i(a_i^*, a_{-i}) \geq U_i(a_i, a_{-i}),\ \ \forall\ a_i \in \mathcal{A}_i$.

A *best reply improvement path* is a sequence of strategy profiles, such that in every step t there is one player that *strictly increases* its utility by updating her strategy through performing a *best reply*. Note that a best reply improvement path implies that only one player at a time updates her strategy; this property is known in the literature as asynchronous updates. An improvement path terminates when no player can perform an improvement step. The resulting strategy profile is a pure strategy NE, defined as a strategy profile a^* in which every player's strategy is a best reply to the other players' strategies

$$U_i(a_i^*, a_{-i}^*) \geq U_i(a_i, a_{-i}^*)\ \ \forall\ a_i \in \mathcal{A}_i,\ \forall\ i \in N. \tag{3}$$

Example 1. *Consider* $|N| = 2$ *players,* $|\mathcal{R}| = 3$ *resources and* $K_i = 1$. *Let* $c_{11} > c_{12} > c_{13} > c_{11}\delta_1 > ..$ *and* $c_{22} > c_{22}\delta_2 > c_{21} > c_{23} > ...$ *If the initial strategy profile is* $(3,1)$ *then the following is a best reply improvement path where the unique deviator is marked at every step:* $(3,1) \underset{1}{\rightarrow} (2,1) \underset{2}{\rightarrow} (2,2) \underset{1}{\rightarrow} (1,2)$. *Observe that* $(1,2)$ *is a NE.*

Let us now turn to the question whether all graphical resource allocation games have at least one pure strategy Nash equilibrium. Consider the strategy profile $a(0)$ that consists of the best replies that the players would play on an edgeless social graph. Let us consider now a best reply path starting from the strategy profile $a(0)$. For $t \leq n$ each player $i \in N$ has a chance to play her first best reply at $t = i$. For $t > n$ they play in an arbitrary order. We can make two important observations about the players' best replies.

Lemma 1. *Player* $i \in N$ *allocates only* i-free *resources when she* first *updates her strategy at* $t = i$.

Proof. Player i first updates her strategy at $t = i$. Define the evicted set as $E_i(t) = \{r | a_i^r(0) = 1 \wedge a_i^r(t) = 0\}$ and the inserted set as $I_i(t) = \{r | a_i^r(0) = 0 \wedge a_i^r(t) = 1\}$. Consider now two resources $r \in E_i(t = i)$ and $r' \in I_i(t = i)$. By definition $a_i^r(0) = 1$ and $a_i^{r'}(0) = 0$, thus by the definition of best reply and because of the edgeless social graph at $t = 0$

$$c_{ir} \geq c_{ir'} \tag{4}$$

Since r' is allocated and r is evicted, at the improvement step $t = i$ it must hold that

$$U_i^{r'}(1, a_{-i}^{r'}(i)) > U_i^r(1, a_{-i}^r(i)). \tag{5}$$

Consider now the definition of the payoff (1) and (4). If r is i-*free* then $U_i^r(1, a_{-i}^r(i)) = c_{ir}$ and consequently (5) cannot hold. Similarly, if r' is i-*busy* then $U_i^{r'}(1, a_{-i}^{r'}(i)) = \delta_i c_{ir'}$ and consequently (5) cannot hold. Thus the only possibility to satisfy (5) is that r' is i-*free* and r is i-*busy*. □

Lemma 2. *Consider a sequence of best reply steps and the best reply $a_i(t)$ played by player i at step $t > 0$. A necessary condition for $a_i(t)$ not being a best reply for i at step $t' > t$ is that at least one of the following holds*

(i) An i-free *resource r allocated by i ($a_i^r(t) = 1$, $\pi_i^r(t) = 1$) becomes* i-busy *by step t',*

(ii) An i-busy *resource r not allocated by i ($a_i^r(t) = 0$, $\pi_i^r(t) = 0$) becomes* i-free *by step t'.*

The proof of Lemma 2 is in the Appendix. Lemma 1 implies that a resource allocated by player i cannot change from *i-free* to *i-busy* during the first round of best reply steps ($1 \leq t \leq n$). Consequently, condition (i) in Lemma 2 cannot hold, and the only reason why player i would update her strategy a second time (at some $t > n$) is that condition (ii) holds, and she would start allocating an *i-free* resource. Thus, by induction, condition (i) will never hold, and in every step t player $i \in N$ starts allocating only *i-free* resources. Since no player ever starts allocating an *i-busy* resource, according to the expression of the payoff in (1) the utilities of the players cannot decrease for $t > 0$. Nevertheless, every time a player updates her strategy her utility must strictly increase. Since the players' utilities cannot increase indefinitely, the best reply path must end in a NE. Hence we can state the following.

Theorem 1. *Every graphical resource allocation game possesses a pure strategy Nash equilibrium.*

For a complete influence graph every player makes at most one best-reply improvement step [20], but this does not hold even for a simple non-complete influence graph: on a ring of 4 players with $c_{ir} = c_{jr}$ ($\forall r \in \mathcal{R}$) at least one player updates her strategy twice. We can however bound the number of required improvement steps. We know that from $a(0)$ every player can only start allocating *i-free* resources. In the worst case each player i inserts exactly one *i-free* resource r at every best reply step. According to condition (ii) in Lemma 2, r becomes *i-free* as an effect of a best reply of a player $j \in \mathcal{N}(i)$. The number of resources that can change from *i-busy* to *i-free* for an arbitrary player i are at most $\sum_{j \in \mathcal{N}(i)} K_j$, thus we obtain the following result

Theorem 2. *It is possible to compute a Nash equilibrium of a graphical resource allocation game in at most $\sum_{i \in N} \sum_{j \in \mathcal{N}(i)} K_j$ steps.*

Due to space limitations we do not address issues such as the price of anarchy and price of stability, and the occurrence of phenomena such as Bélády's anomaly [21]. Instead, in the rest of the paper our focus is on how to reach Nash equilibria in an efficient way, with the aim of providing guidelines for designing distributed algorithms that would be able to reach an equilibrium allocation with little overhead.

3.2 Convergence to Nash Equilibria

The existence of equilibrium states is important, but in a distributed system it is equally important to have efficient distributed algorithms that the nodes can use to reach an equilibrium state. The algorithm in Theorem 1 can be used to reach an equilibrium state if the values c_{ir} of the resources in the nodes never change, so once a NE is reached, the nodes will not deviate from it. Nevertheless, the algorithm would be inefficient if the values c_{ir} or the influence graph can change over time, as the equilibrium states for different values and different influence graphs are, in general, different. Hence, an important question is whether the players will reach a NE given an arbitrary initial strategy profile, e.g., a NE for past c_{ir} or a NE for a past influence graph, and given the myopic decisions the players make to update their strategies.

A straightforward distributed algorithm would be to let every player play her best reply at the same time (synchronously) until an equilibrium is reached. The algorithm only requires synchronization between neighboring players, thus it is relatively easy to implement. Unfortunately, it is easy to find resource allocation games where players cannot find an equilibrium this way.

Example 2. *Consider two players and* $|\mathcal{R}| \geq 2$. *Let* $c_{i1} > c_{i2} > c_{i1}\delta_i > c_{i2}\delta_i$ *and* $K_i = 1$. *If the initial allocation strategies are* $a_i(0) = (1, 0)$ *then we have* $(1, 0) \underset{1,2}{\longrightarrow} (0, 1) \underset{1,2}{\longrightarrow} (1, 0)$, *etc.*

As an alternative, consider an algorithm that only allows one player to play a best reply at a time. This is the case of the asynchronous updates used in Section 3.1, which generate best reply improvement paths. The algorithm requires global synchronization to ensure that no players perform an update simultaneously. In what follows we show that even best reply improvement paths can be arbitrarily long.

Consider a resource allocation game played over the influence graph shown in Figure 1, where $\mathcal{R} = \{a, b, c, d\}$, and $K_i = 1$. Each player $1 \leq i \leq 4$ has a resource $r^* \in \mathcal{R}$ such that $c_{ir^*}\delta_i > c_{ir}\delta_i \ \forall r \neq r^*$, and at least one resource $r' \in \mathcal{R}$ such that $c_{ir'} > c_{ir^*}\delta_i$. For players $5 \leq i \leq 8$ there is a resource $r^* \in \mathcal{R}$ such that $c_{ir^*}\delta_i > c_{ir} \ \forall r \neq r^*$. In the following we show an asynchronous best reply dynamic that cycles, we omit the strategies of players $5 \leq i \leq 8$ since they always allocate the resource that has the highest value at their respective neighboring player $i - 4$. The *i-busy* resources are in bold.

$$\begin{aligned}(\boldsymbol{a}, b, \boldsymbol{d}, \boldsymbol{a}) \underset{3}{\rightarrow} \quad (\boldsymbol{a}, b, c, \boldsymbol{a}) \underset{1}{\rightarrow} \quad (\boldsymbol{b}, \boldsymbol{b}, c, \boldsymbol{a}) \underset{4}{\rightarrow} \quad (\boldsymbol{b}, \boldsymbol{b}, c, d) \underset{2}{\rightarrow} \quad (\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{c}, d) \\ \underset{1}{\rightarrow} \quad (a, \boldsymbol{c}, \boldsymbol{c}, d) \underset{3}{\rightarrow} \quad (a, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{d}) \underset{2}{\rightarrow} \quad (a, b, \boldsymbol{d}, \boldsymbol{d}) \underset{4}{\rightarrow} \quad (\boldsymbol{a}, b, \boldsymbol{d}, \boldsymbol{a})\end{aligned}$$

The existence of a cycle in the best reply paths implies that resource allocation games played over an arbitrary graph do not admit a potential function [13]. It also raises the question whether there could be an initial strategy from which *every*

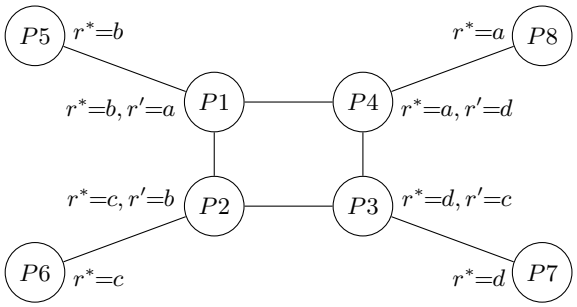

Figure 1: Influence graph and players' preferences that allow a cycle in best replies.

best reply improvement path is infinite. Starting a best reply improvement path from such an initial strategy profile, the players would never reach a NE using the asynchronous algorithm.

We show that fortunately this is not the case; from every strategy profile there exists at least one finite best reply path that leads to a NE. This property is called weak acyclicity [14, 22].

Definition 2. A game is *weakly acyclic under best replies* if from every strategy profile a, there is a best reply improvement path starting from a and ending in a pure Nash equilibrium.

To show that resource allocation games are weakly acyclic, let us consider a best reply $a_i(t)$ of a player i at step t. We can define four not mutually exclusive properties of $a_i(t)$, depending on whether the involved resources are *i-busy* as shown in Table 1.

1	$\exists r \in E_i(t)$, r *i-busy*	$\exists r' \in I_i(t)$, r' *i-busy*
2	$\exists r \in E_i(t)$, r *i-busy*	$\exists r' \in I_i(t)$, r' *i-free*
3	$\exists r \in E_i(t)$, r *i-free*	$\exists r' \in I_i(t)$, r' *i-busy*
4	$\exists r \in E_i(t)$, r *i-free*	$\exists r' \in I_i(t)$, r' *i-free*

Table 1: Four properties of a best reply move

Note that a best reply necessarily has at least one of the listed properties. The following two lemmas state that the best replies in a cycle cannot be arbitrary.

Lemma 3. *In every best reply cycle there exists at least one strategy profile $a(t)$ in which at least one player $i \in N$ performs a best reply that has property 1). Furthermore, in a best reply cycle it is not possible to perform a best reply that has property 3).*

Lemma 4. *Assume that a player i performs a best reply with property 1) at step t in a best reply cycle such that $r' \in I_i(t)$ and r' is i-*busy*. Then every resource $r'' \in \mathcal{R}$ for which $c_{ir''} > c_{ir'}$ is allocated by player i at step t ($a_i^{r''}(t)=1$).*

The proofs of the lemmas are given in the Appendix. Let us consider a strategy profile $a(t)$ in a best reply cycle in which at least one player can perform a best reply that has property 1). Such a strategy profile exists according to Lemma 3. Starting from $a(t)$, let us perform a sequence of best replies with property 1). From Lemma 4, it follows that player i cannot evict the resources that she inserted as *i-busy* through these best replies, thus every player i can perform at most K_i best replies with property 1). So after at most $\sum_{i \in N} K_i$ best replies with property 1) we reach a strategy profile $a(t')$ in which there is no player that can perform a best reply that has property 1). If in $a(t')$ no player can make a best reply then $a(t')$ is a NE. Otherwise, if a player i can perform a best reply in $a(t')$ then it will have neither property 1) nor 3). As a consequence of a best reply that has property 2) or 4) only, condition (i) of Lemma 2 cannot hold for any player, so the only new reason why a player j would perform a best reply is that condition (ii) in Lemma 2 is satisfied, and she would start allocating *j-free* resources. Thus, by induction, condition (i) of Lemma 2 will never hold, and in every step $t'' > t'$ players only perform best replies that have property 2) or 4) but not 1) or 3). A player's utility strictly increases when it performs a best reply, and a best reply with property 2) or 4) does not decrease any other player's utility. Since the players' utilities cannot increase indefinitely, this path must end in a NE after a finite number of steps. Hence we can state the following.

Theorem 3. *Every graphical resource allocation game is weakly acyclic under best replies.*

Weak acyclicity has two important consequences for system design. First, if the nodes that compete for the resources update their allocations one at a time in a myopic way and the order of updates is fixed then the nodes might cycle forever without reaching an equilibrium. Therefore, it is advisable that the next node to perform the update step is chosen at random by the system. By doing so the system will reach an equilibrium after a finite number of update steps *on average*, even though the number of update steps can be arbitrarily high. Second, weak acyclicity in best replies implies that various complex learning rules can be used to reach an equilibrium with high probability. One example is adaptive play, described in [22], when nodes select the next allocation simultaneously based on a random sample of a finite history of the allocations. Such a learning rule resembles a system in which the nodes update their allocations based on a sliding window estimator of the other players' allocations. Another example is regret based learning [23], in which case every node updates its allocation simultaneously so as to minimize its loss of utility in retrospect given the history of all allocations.
In the discussion above we showed that every best reply path that starts from a

strategy profile $a(t')$ in which no player can perform a best reply that has property 1) ends in a NE. It follows that in every strategy profile of a best reply cycle there is at least one player i that can perform a best reply with property 1). Let us consider now the number of steps needed to reach a NE starting from an arbitrary strategy profile $a(t)$. We have already shown that after performing at most $\sum_{i\in N} K_i$ best replies with property 1) we reach a strategy profile $a(t')$ in which there is no player that can perform a best reply that has property 1). Given that there is no player that can perform a best reply with property 1), we can use the same reasoning that we formulated to show Theorem 2 to bound the maximum number of best reply steps from $a(t')$ to a NE to $\sum_{i\in N}\sum_{j\in\mathcal{N}(i)} K_j$. Summing it to the maximum number of steps required from $a(t)$ to $a(t')$ we obtain the following upper bound.

Corollary 1. *From an arbitrary strategy profile there exists a best reply path that reaches a NE in at most* $\sum_{i\in N} K_i + \sum_{i\in N}\sum_{j\in\mathcal{N}(i)} K_j$ *steps.*

As an example, consider a graphical resource allocation game with $K_i = K$ $\forall i \in N$ and a regular influence graph of degree d. In this case the maximum length of a best reply path would be $|N| \cdot K + |N| \cdot d \cdot K = |N|K(1+d)$.

4 The Case of Complete Influence Graph

In this section we consider the case that the influence graph is *complete* and use it to illustrate the influence of the graph topology on the convergence to equilibria. We make use of the notion of the finite best reply property [14].

Definition 3. A game possesses the *finite best reply property* (FBRP) if every best reply improvement path is finite.

For a complete influence graph the resource allocation game is a special case of the player-specific matroid congestion games investigated in [16], and it is known that if $|\mathcal{A}_i| = 2$ $\forall i \in N$ or if $|N| = 2$, then the game has the FBRP. It is not known whether the game possesses the FBRP in general.

Let us denote by $F_i(t)$ the set of *i-free* resources and by $B_i(t)$ the set of *i-busy* resources allocated by i at step t. Note that $|F_i(t)| + |B_i(t)| = K_i$. We can prove the following lemmas.

Lemma 5. *On a complete influence graph, a best reply step* $a_i(t)$ *performed by player* i *in a best reply cycle cannot affect the number of* i-busy *resources allocated by* i*, that is,* $|B_i(t)| = |B_i(t-1)|$.

The proof of Lemma 5 is given in the Appendix.

Lemma 6. *On a complete influence graph, for a best reply step* $a_i(t)$ *performed by player* i *in a best reply cycle it holds that* $c_{ir'} < c_{ir}$, $\forall r' \in E_i(t), r \in I_i(t)$.

Proof. Player i solves a knapsack problem to construct her best reply. Recall that according to Lemma 5 we have $|B_i(t-1)| = |B_i(t)|$ and consequently $|F_i(t-1)| = |F_i(t)|$. Hence we can construct the best reply of player i by dividing the knapsack problem into two similar subproblems: we can solve the knapsack problem for all the *i-busy* resources and populate the set $B_i(t)$, and do the same with the set $F_i(t)$ using the *i-free* ones. Suppose that k resources are evicted from one set, consequently k are inserted, and in order for the result to be the solution of the knapsack problem, the cost saving yielded by each inserted resource must be higher than the cost saving yielded by each evicted resource. That is, for every $r, r' \in \mathcal{R}$ such that r was inserted and r' was evicted from $B_i(t)$, we have $c_{ir}\delta_i > c_{ir'}\delta_i \Rightarrow c_{ir} > c_{ir'}$. Similarly, for every $r, r' \in \mathcal{R}$ such that r was inserted and r' was evicted from $F_i(t)$, we have $c_{ir} > c_{ir'}$. This proves the lemma. □

Assume now that there is a best reply cycle. According to Lemma 6 each best reply step can only move towards resources with higher value. The number of resources $|\mathcal{R}|$ is finite, hence every player can only perform a finite number of best replies and the best reply path terminates after a finite number of steps. Thus the following result holds.

Theorem 4. *Every resource allocation game played over a complete influence graph has the FBRP.*

Theorem 4 has important practical implications. Recall that based on the results for a general graph topology shown in Section 3.2 it was advisable to ensure that nodes would update their allocations one at a time and in a random order so as to ensure that an equilibrium allocation can be reached; the number of update steps was unbounded in general but finite on average. By Theorem 4, if the influence graph is complete and the nodes that compete for the resources update their allocations one at a time by using some synchronization protocol, then they will reach an equilibrium allocation after a *bounded* number of update steps starting from an arbitrary initial allocation independent of the order of the updates. Since the order of updates can be arbitrary (e.g,. fixed), the system would require less coordination between the nodes.

5 The Importance of the Utility Structure

In this section we allow an arbitrary influence graph but we introduce a constraint on the payoff structure: we consider the case when $\delta_i = 0$. This case was used in the context of replication to model the problem of cooperative caching between peering ISPs in [2], and in the context of radio spectrum allocation $\delta_i = 0$ corresponds to the case when interference leads to zero expected throughput (i.e., proper multi-coloring). Throughout the section we consider a notion of improvement paths that we refer to as lazy.

Definition 4. A step $a_i(t+1)$ of player i is an *improvement step* if $U_i(a_i(t+1), a_{-i}(t)) > U_i(a_i(t), a_{-i}(t))$. A *lazy improvement step* of player i is an improvement step such that the payoff of every inserted resource exceeds that of every evicted resource. That is, for every $r \in E_i(t+1)$ and $r' \in I_i(t+1)$ we have $U_i^r(1, a_{-i}^r(t)) < U_i^{r'}(1, a_{-i}^{r'}(t))$.

Clearly, every best reply improvement step is a lazy improvement step. The following result shows that all lazy improvement paths are finite for arbitrary graph topologies if $\delta_i = 0$.

Proposition 5. *In a graphical resource allocation game with $\delta_i = 0$ $\forall i \in N$ every lazy improvement path is finite, i.e., the game possesses the* finite lazy improvement property.

Proof. We prove the proposition by showing that under the condition $\delta_i = 0$ the resource allocation game has a generalized ordinal potential function [13] for lazy improvement steps. A function $\Psi : \times_i(\mathcal{A}_i) \to \mathbb{R}$ is a generalized ordinal potential function for the game if the change of Ψ is strictly positive if an arbitrary player i increases its utility by changing her strategy from a_i to a_i'. Formally, $U_i(a_i, a_{-i}) - U_i(a_i', a_{-i}) > 0 \Rightarrow \Psi(a_i, a_{-i}) - \Psi(a_i', a_{-i}) > 0$. In the following we show that the function

$$\Psi(\mathbf{a}) = \sum_i U_i(a_i, a_{-i}), \tag{6}$$

where the utility function was defined in (1), is a generalized ordinal potential for the game. Substituting $\delta_i = 0$ into (1), one notes that player i benefits only from allocating *i-free* resources. Furthermore, the utility of player i does not depend on resources that she does not allocate herself. Given a strategy profile $\mathbf{a} = (a_i, a_{-i})$ player i can improve its utility by combining three kinds of lazy improvement steps.

First, if player i has free storage capacity (that is, $\sum_r a_i^r < K_i$) then she has to allocate a resource r for which $a_i^r = 0$ but $U_i^r(1, a_{-i}^r) > 0$. By (1) we know that resource r is *i-free* and hence the utility of her neighbors will not be affected if she allocates resource r. Consequently, if we denote the new strategy of player i by $a_i' = (a_i^1, \ldots, a_i^{r-1}, 1, a_i^{r+1}, \ldots, a_i^{|\mathcal{R}|})$ then

$$\Psi(a_i', a_{-i}) - \Psi(a_i, a_{-i}) = U_i(a_i', a_{-i}) - U_i(a_i, a_{-i}) = U_i^r(1, a_{-i}^r) > 0. \tag{7}$$

Second, if player i stops allocating a resource r for which $U_i^r(1, a_{-i}^r) = 0$ and starts allocating a resource r' for which $U_i^{r'}(1, a_{-i}^{r'}) > 0$. By (1) we know that resource r is *i-busy*, but resource r' is *i-free*. Let us denote the strategy of player i after the change by a_i'. We first observe that the utility of the neighboring players cannot decrease when player i stops allocating resource r (it can potentially increase). At the same time the utility of the neighboring players does not change when player i starts allocating resource r'. Hence we have that

$$\Psi(a_i', a_{-i}) - \Psi(a_i, a_{-i}) \geq U_i^{r'}(1, a_{-i}^{r'}) > 0. \tag{8}$$

Third, if player i stops allocating a resource r for which $U_i^r(1, a_{-i}^r) > 0$ and starts allocating a resource r' for which $U_i^{r'}(1, a_{-i}^{r'}) > U_i^r(1, a_{-i}^r)$. By (1) we know that both resource r and r' are *i-free*. Hence the utility of the neighboring players is not affected by the change. The utility of player i increases, however. Let us denote the strategy of player i after the change by a_i', then

$$\Psi(a_i', a_{-i}) - \Psi(a_i, a_{-i}) = U_i^{r'}(1, a_{-i}^{r'}) - U_i^r(1, a_{-i}^r) > 0. \quad (9)$$

By summing (7)-(9) we can see that the function Ψ is a generalized ordinal potential function for an arbitrary combination of lazy improvement steps. Finite games that allow a generalized ordinal potential function were shown to have the finite improvement property in [13]. Following the arguments of (Lemma 2.3, [13]) it follows that the resource allocation game has the finite lazy improvement property. We note that if non-lazy improvement steps are allowed, then a player can allocate an *i-busy* resource as part of an improvement step and thus the proof would not hold. □

The finite lazy improvement property shown in Proposition 5 allows more freedom for system design than the results in Theorems 3 and 4 that are valid for best reply updates. Based on Proposition 5 an update step can be as simple as evicting the worst allocated resource and inserting the best non-allocated resource, and the system is guaranteed to reach an equilibrium for an arbitrary graph topology in a bounded number of steps.

6 Fast convergence based on independent sets

Until now we considered asynchronous updates. Unfortunately, the implementation of the asynchronous update rule in a distributed system would require global synchronization, which can be impractical in large distributed systems. Hence, an important question is whether the previous results of convergence to a NE would be preserved even if some players would update their strategies simultaneously. In the following we show that relaxing the requirement of asynchronicity is indeed possible: both Theorem 3 and Proposition 5 hold if the players that simultaneously make an update at each step form an independent set of the influence graph $\mathcal{G}$, that is, two players i,$j \in N$ can make an update simultaneously only if $j \notin \mathcal{N}(i)$. We refer to this dynamic as the *plesiochronous* dynamic.

Proposition 6. *Every resource allocation game is weakly acyclic under plesiochronous best replies.*

The proof of the proposition is analogous to the proof of Theorem 3.

Proposition 7. *In a graphical resource allocation game with $\delta_i = 0$ $\forall i \in N$ every plesiochronous lazy improvement path is finite.*

Proof. Consider a sequence of the subsets of the players $N^*(t) \subseteq N$ $(t = 0, \ldots)$ such that for $i, j \in N^*(t)$ we have $j \notin \mathcal{N}(i)$. Observe that, by definition, each set $N^*(t)$ is an independent set of the influence graph $\mathcal{G}$. The players $i \in N^*(t)$ make an improvement step at step t simultaneously from $a_i(t-1)$ to $a_i(t)$. Each player can combine the three kinds of lazy improvement steps discussed in the proof of Proposition 5 to increase its utility.

Since we require that none of the players that update their strategies are neighbors of each other, then their updates do not affect each others' utilities. Formally, for $i \in N^*(t)$ we have $U_i(a_i(t), a_{-i}(t)) = U_i(a_i(t), a_{-i}(t-1))$. Consequently, we can use the same arguments as in the proof of Proposition 5 to show for every $i \in N^*(t)$ that Ψ defined in (6) satisfies $U_i(a_i(t), a_{-i}(t)) - U_i(a_i'(t-1), a_{-i}(t-1)) > 0 \Rightarrow \Psi(a(t)) - \Psi(a(t-1)) > 0$.
The rest of the proof is similar to that of Proposition 5. □

In order to maximize the convergence speed of the plesiochronous dynamic we need to find a minimum vertex coloring of $\mathcal{G}$, i.e., we have to find the chromatic number $\chi(\mathcal{G})$ of graph $\mathcal{G}$. Finding the chromatic number is NP-hard in general, but efficient distributed graph coloring algorithms exist [24], and can be used to find a coloring in a distributed system. The chromatic number can be bounded based on the largest eigenvalue $\lambda_{max}(\mathcal{G})$ of the graph's adjacency matrix [25],

$$\chi(\mathcal{G}) \leq 1 + \lambda_{max}(\mathcal{G}) \leq \max_{i \in N} |\mathcal{N}(i)|$$

where the second inequality follows from the Perron-Frobenius theorem ([26], Lemma 2.8). Given a coloring, the number of steps needed to reach equilibrium can be significantly smaller than for the corresponding asynchronous dynamic for sparse graphs.

We illustrate the convergence speedup of the plesiochronous dynamic compared to the asynchronous dynamic in Figures 2 and 3. For the plesiochronous dynamic we used the Welsh-Powell algorithm to find a coloring [27] of the influence graphs and we denote the number of colors by $\xi(\mathcal{G})$. Each player can allocate $K_i = 5$ resources and we considered two scenarios: $\delta_i = 0$ and $\delta_i = 0.5$. Figure 2 shows the average number of lazy improvement steps needed to reach equilibrium as a function of the average node degree in Erdős-Rényi random graphs with 87 vertices. Each data point is the average of the results obtained on 160 random graphs with the same average node degree. The figure shows the 95% confidence intervals for the case $\delta_i = 0$. We omitted the confidence intervals for $\delta_i = 0.5$ to improve readability. The results confirm that the plesiochronous dynamic converges significantly faster than the asynchronous dynamic, especially over sparse influence graphs. The figure also confirms that the convergence properties are different on a complete influence graph than on a sparse graph, as the number of steps necessary for the asynchronous dynamic to reach a NE drops for average node degree equal to 86.
Figure 3 shows the speedup (the ratio of the number of steps needed to reach equilibrium under the asynchronous dynamic and the plesiochronous dynamic) as a

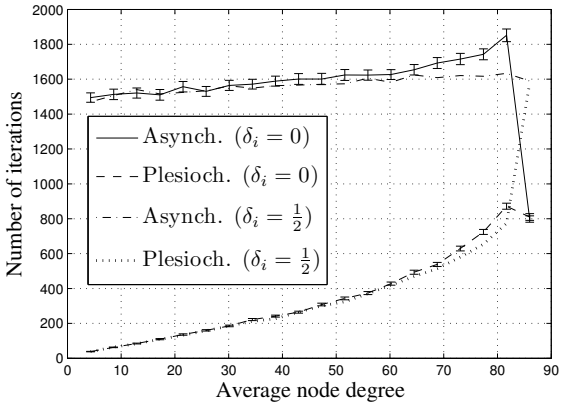

Figure 2: Average number of improvement steps needed to reach a NE for asynchronous and plesiochronous dynamics vs. the average node degree of the ER random graphs.

function of the average node degree for Erdős-Rényi (ER) and for Barabási-Albert (BA) random graphs with 87 vertices for $\delta_i = \frac{1}{2}$. Each data point is the average of the results obtained on 160 random graphs with the same average node degree. The results show that the speedup for BA random graphs is slightly smaller than for ER random graphs but shows a similar trend. A comparison of the speedup with the average size of the independent sets after the coloring of the random graphs ($\frac{|N|}{\xi(\mathcal{G})}$) indicates that the speedup of the plesiochronous dynamic exceeds the average number of players that can perform an improvement step simultaneously.

The results formulated in Propositions 6 and 7 allow one to simplify the design of large systems significantly. The convergence results of Sections 3, 4 and 5 required that only one node at a time would update its allocation. In order to ensure this, there needs to be a coordination protocol for the entire system that ensures that there be only one node at a time that updates its allocation. In contrast, Propositions 6 and 7 show that local coordination is sufficient to ensure that the system would reach an equilibrium allocation, and the numerical results show that convergence is in fact very fast. In practice the information needed for the coordination can be piggybacked with the information about the updated allocations, and thus an equilibrium allocation can be reached with very little communication overhead.

7 Related Work

There is a large body of works on congestion games [5] on complete influence graphs. Most works consider congestion games that allow a potential function [13], and analyze the number of steps needed to reach equilibria [9], the price of anarchy and the price of stability [10, 11], or the complexity of calculating equilibria [12].

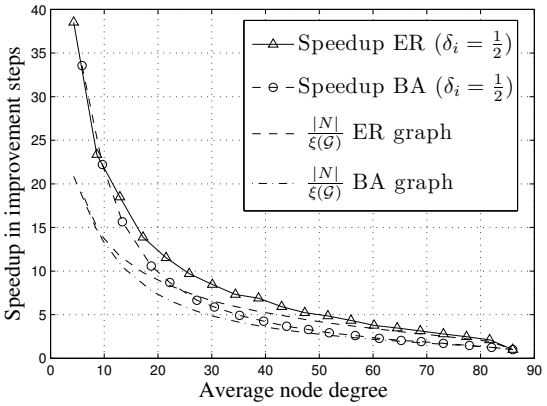

Figure 3: Speedup in terms of the number of improvement steps needed to reach a NE vs. the average node degree of the ER and the BA random graphs.

Player-specific congestion games on a complete influence graph that do not admit a potential function were considered in [14, 20, 15, 16]. In [14] it was shown that for non-weighted player-specific congestion games with *singleton action sets* the best reply paths are finite. In [20] the authors showed the existence of equilibria for a game of replication. In [15], the authors considered congestion games with player-specific constants, which correspond to all players having the same cost of sharing($\delta_i = \delta$) in our model, and showed that improvement paths are finite in the unweighted version of the game. In [16] the authors showed that player-specific congestion games with matroid action sets are weakly acyclic in better replies, and provided bounds on the length of the shortest improvement paths. They also showed that games with 2 players, or with 2 actions per player are acyclic in best replies, similar to [14].

A few recent works considered graphical congestion games that allow potential functions [6, 7, 8]. [6] gave results on the price of anarchy and stability for games with linear payoff functions, and showed that cycles can exist if the influence graph is directed and contains cycles. [7] addressed similar questions for weighted graphical congestion games with linear payoff functions. In [8] the authors analyzed the number of steps needed to reach equilibrium in graphical congestion games with homogeneous resources and singleton action sets.

The resource allocation game we consider combines the concept of graphical congestion games with player-specific payoffs on non-singleton action sets. Our results on equilibrium existence, and the results on convergence rely on non-standard techniques and could be of interest for the analysis of congestion games that do not admit a potential function. Furthermore, the proposed plesiochronous update dynamic based on independent sets is a promising candidate for implementation in large distributed systems.

8 Conclusion

In this work we considered a player-specific graphical resource allocation game, a resource allocation game played over an influence graph. The game models a system in which every node can choose a subset of resources, and the value of a selected resource to a node is decreased if any of its neighbors chooses the same resource. We showed that pure strategy Nash equilibria exist in the game for arbitrary influence graph topologies even though the game does not admit a potential function, and gave a bound on the complexity of finding equilibria. We then considered the problem of reaching an equilibrium when nodes update their allocations one at a time to their best allocation, that is, every node performs a best reply. We showed that for non-complete influence graphs there might be cycles in best replies but the system would reach an equilibrium eventually if updates are done in a random order. For complete influence graphs there are no cycles, and hence an equilibrium is reached after a finite number of steps. We also showed that if neighboring nodes try to allocate disjoint sets of resources then there are no cycles, even if the nodes' update steps are improvements instead of best replies. Finally, we showed that the convergence properties hold even if improvement steps are performed simultaneously by nodes that form an independent set of the influence graph, and proposed an efficient algorithm to reach a NE over sparse influence graphs that only requires local coordination between neighboring nodes.

References

[1] 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.

[2] G. Dán, "Cache-to-cache: Could ISPs cooperate to decrease peer-to-peer content distribution costs?" *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 9, pp. 1469–1482, 2011.

[3] L. Narayanan, "Channel assignment and graph multicoloring," in *Handbook of wireless networks and mobile computing*, 2002, pp. 71–94.

[4] K. I. Aardal, V. Hoesel, P. M. Stan, A. M. C. A. Koster, C. Mannino, and A. Sassano, "Models and solution techniques for frequency assignment problems," *Annals of Operations Research*, vol. 153, no. 1, pp. 79–129, 2007.

[5] R. W. Rosenthal, "A class of games possessing pure-strategy Nash equilibria," *International Journal of Game Theory*, vol. 2, no. 1, pp. 65–67, 1973.

[6] V. Bilò, A. Fanelli, M. Flammini, and L. Moscardelli, "Graphical Congestion Games." in *Proc. of WINE*, vol. 5385, 2008, pp. 70–81.

[7] D. Fotakis, V. Gkatzelis, A. C. Kaporis, and P. G. Spirakis, "The Impact of Social Ignorance on Weighted Congestion Games," in *Proc. of WINE*, 2009, pp. 316–327.

[8] R. Southwell and J. Huang, "Convergence Dynamics of Resource-Homogeneous Congestion Games," in *Proc. of GameNets*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 75, no. 412509, 2011, pp. 281–293.

[9] S. Chien and A. Sinclair, "Convergence to approximate Nash equilibria in congestion games," in *Proc. of the 18th annual ACM-SIAM Symposium on Discrete Algorithms*, 2007, pp. 169–178.

[10] G. Christodoulou and E. Koutsoupias, "The price of anarchy of finite congestion games," in *Proc. of ACM Symposium on Theory of Computing*, 2005, pp. 67–73.

[11] B. Awerbuch, Y. Azar, and A. Epstein, "The price of routing unsplittable flow," in *Proc. of ACM Symposium on Theory of Computing*, 2005, pp. 57–66.

[12] A. Fabrikant, C. Papadimitriou, and K. Talwar, "The complexity of pure Nash equilibria," in *Proc. of ACM Symposium on Theory of Computing*, 2004, pp. 604–612.

[13] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, pp. 124–143, 1996.

[14] I. Milchtaich, "Congestion games with player-specific payoff functions," *Games and Economic Behavior*, vol. 13, no. 1, pp. 111–124, 1996.

[15] M. Mavronicolas, I. Milchtaich, B. Monien, and K. Tiemann, "Congestion games with player-specific constants," *Mathematical Foundations of Computer Science 2007*, pp. 633–644, 2007.

[16] H. Ackermann, H. Röglin, and B. Vöcking, "Pure Nash equilibria in player-specific and weighted congestion games," *Theor. Computer Science*, vol. 410, no. 17, pp. 1552–1563, 2009.

[17] A. Leff, J. L. Wolf, and P. S. Yu, "Replication Algorithms in a Remote Caching Architecture." *IEEE Trans. Parallel Distrib. Syst.*, vol. 4, no. 11, pp. 1185–1204, 1993.

[18] M. M. Halldórsson and G. Kortsarz, "Tools for Multicoloring with Applications to Planar Graphs and Partial k-Trees," *Journal of Algorithms*, vol. 42, no. 2, pp. 334–366, 2002.

[19] H. Huang, B. Zhang, G. Chan, G. Cheung, and P. Frossard, "Coding and Replication Co-Design for Interactive Multiview Video Streaming," in *Proc. of mini-conference in IEEE INFOCOM*, 2012, pp. 1–5.

[20] N. Laoutaris, O. Telelis, V. Zissimopoulos, and I. Stavrakakis, "Distributed selfish replication," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 12, pp. 1401–1413, 2006.

[21] L. A. Belady, R. A. Nelson, and G. S. Shedler, "An Anomaly in Space-Time Characteristics of Certain Programs Running in a Paging Machine," in *Communications of the ACM*, vol. 12, no. 6, 1969, pp. 349–353.

[22] H. P. Young, "The evolution of conventions," *Econometrica: Journal of the Econometric Society*, vol. 61, no. 1, pp. 57–84, 1993.

[23] J. R. Marden, G. Arslan, and J. S. Shamma, "Regret Based Dynamics: Convergence in Weakly Acyclic Games," in *Proc. of AAMAS*, 2007, pp. 194–201.

[24] F. Kuhn, "Weak Graph Colorings: Distributed Algorithms and Applications," in *ACM SPAA*, 2009, pp. 138–144.

[25] H. S. Wilf, "The eigenvalues of a graph and its chromatic number," *J. London Math. Soc.*, vol. 42, pp. 330–332, 1967.

[26] R. S. Varga, *Matrix Iterative Analysis*, 2002.

[27] 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.

Proof of Lemma 2. According to the structure of the utility function $U_i(a_i, a_{-i}) = \sum_{\{r|a_i^r=1\}} U_i^r(1, a_{-i})$, a best reply $a_i(t)$ can stop to be such in two situations:

(i) The payoffs of one or more allocated resources $\{r \in \mathcal{R}|a_i^r(t) = 1\}$ decrease;

(ii) The payoffs of one or more not allocated resources $\{r \in \mathcal{R}|a_i^r(t) = 0\}$ increase;

According to the definition of cost saving in (1), case (i) can happen only if some *i-free* resources allocated by i become *i-busy*. This requires that some player $j \in \mathcal{N}(i)$ starts allocating some *j-busy* resources. Similarly, case (ii) can happen only if a neighbor $j \in \mathcal{N}(i)$ allocating an resource r evicts it, making resource r *i-free*. □

Proof of Lemma 3. We prove the lemma by showing that a best reply satisfying condition 1) must occur in a best reply cycle in order for the cycle to exist. The utility of at least one player must decrease at least once in a best reply cycle. According to the definition of cost saving in (1), in order for the utility $U_j(a(t))$ of a player j to decrease, some neighbor $i \in \mathcal{N}(j)$ needs to start allocating some *i-busy* resource allocated by j. It follows that, from Table 1, in a best reply cycle there has to be at least one best reply satisfying condition 1) or 3). Let $r \in E_i(t)$ and $r' \in I_i(t)$ be two resources in the evicted and inserted sets, respectively, during a best reply of player i at step t in a best reply cycle. It follows from the definition

of inserted and evicted set that $a_i^r(t-1) = 1$ and $a_i^{r'}(t-1) = 0$. Assume now that this best reply satisfies 3). This implies $c_{ir} < c_{ir'}\delta_i$. Note that this inequality also implies that player i always prefers r' over r (r' always provides a higher cost saving) and thus if $a_i^r(t-1) = 1$, then $a_i^{r'}(t-1) = 1$. Since there cannot be a best reply satisfying 3), there must be at least one best reply satisfying 1). This proves the lemma. □

Proof of Lemma 4. Since r' is *i-busy* it yields to player i the payoff $c_{ir'}\delta_i$. Consider an object r'' for which $c_{ir''} > c_{ir'}$. Since player i is performing a best reply, if r'' is *i-free* then its payoff is $c_{ir''} > c_{ir'}\delta_i$ and consequently $a_i^{r''}(t) = 1$. Similarly, if r'' is *i-busy* then its payoff is $c_{ir''}\delta_i > c_{ir'}\delta_i$, and consequently $a_i^{r''}(t) = 1$. □

Proof of Lemma 5. Part A: First we show that $|B_i(t-1)| \geq |B_i(t)|$. Player i can only increase the number of *i-busy* allocated resources if she evicts at least one *i-free* resource r' from $a_i(t-1)$ and inserts an *i-busy* resource r at step t. Thus by (1) we have

$$c_{ir'} < c_{ir}\delta_i \tag{10}$$

Since we are in a best reply cycle, at some step $t' > t$ the strategy $a_i(t-1)$ must become a best reply for player i, i.e. $a_i(t') = a_i(t-1)$. This requires either $c_{ir'} > c_{ir}$ or $c_{ir'} > c_{ir}\delta_i$, and both contradict (10).
Part B: Second we show that for every step in a best reply cycle $|B_i(t-1)| = |B_i(t)|$ must hold. We denote by $C(t)$ the set of the chosen resources, the resources allocated by at least one player in $a(t)$, $C(t) = \{r | a_j^r(t) = 1 \text{ for some } j \in N\} \subseteq \mathcal{R}$. Similarly, we denote by $C(t)_{-i}$ the set of resources allocated by the players not including i, $C(t)_{-i} = \{r | a_j^r(t) = 1 \text{ for some } j \in N \setminus \{i\}\}$. It is easy to see that the sets $|F_i(t)|$ and $|C(t)|$ are related and on a complete influence graph it holds that $|C(t)| = |C(t)_{-i}| + |F_i(t)|$. On one hand, a best reply for which $|F_i(t-1)| = |F_i(t)|$ does not affect $|C(t)|$ since $|C(t-1)_{-i}| = |C(t)_{-i}|$. On the other hand, a best reply for which $|F_i(t-1)| > |F_i(t)|$ decreases the size of set C

$$\begin{aligned} |C(t)| &= |C(t)_{-i}| + |F_i(t)| = |C(t-1)_{-i}| + |F_i(t)| \\ &< |C(t-1)_{-i}| + |F_i(t-1)| = |C(t-1)| \end{aligned}$$

Since best replies for which $|F_i(t-1)| > |F_i(t)|$ do not exist in a cycle (*Part A*), best replies for which $|F_i(t-1)| < |F_i(t)|$ cannot exist either, as otherwise the size of set C would increase indefinitely. Hence $|F_i(t-1)| = |F_i(t)|$, which proves the lemma. □

Valentino Pacifici studied computer engineering at Politecnico di Milano in Milan, Italy. In October 2010, he completed a joint M.Sc. degree in computer engineering, between Politecnico di Milano and KTH, Royal Institute of Technology in Stockholm, Sweden. Now he is a researcher at the Laboratory for Communication Networks in KTH, Royal Institute of Technology and he is pursuing his Ph.D. His research interests include the modeling, design and analysis of content management systems using game theoretical tools.

György Dán received the M.Sc. degree in computer engineering from the Budapest University of Technology and Economics, Hungary in 1999 and the M.Sc. degree in business administration from the Corvinus University of Budapest, Hungary in 2003. He worked as a consultant in the field of access networks, streaming media and videoconferencing 1999-2001. He received his Ph.D. in Telecommunications in 2006 from KTH, Royal Institute of Technology, Stockholm, Sweden, where he currently works as an assistant professor. He was visiting researcher at the Swedish Institute of Computer Science in 2008. His research interests include the design and analysis of distributed and peer-to-peer systems and cyber-physical system security.

Paper C

Content-peering Dynamics of Autonomous Caches in a Content-centric Network

Valentino Pacifici and György Dán

in Proc. of IEEE International Conference on Computer Communications (INFOCOM), 2013.

Content-peering Dynamics of Autonomous Caches 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

March 22, 2014

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 or alternatively, whether the interaction needs to rely on explicit announcements of content reachability in order for the system to be stable. We show that content-level peering leads to stable cache configurations, both with and without coordination. If the ASes do coordinate, then coordination that avoids simultaneous updates by peering ISPs provides faster and more cost efficient convergence to a stable configuration. Furthermore, if the content popularity estimates are inaccurate, content-level peering is likely to lead to cost efficient cache allocations. We validate our analytical results using simulations on the measured peering topology of more than 600 ASes.

1 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. Several recent works aimed at optimizing the performance of a cache network through dimensioning cache sizes as a function of their location in the cache network [4], by routing interest messages to efficiently find contents [5] and by tuning the cache eviction policies used by the individual caches [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. The effect of BGP coordination on the stability and performance of global IP routing has been extensively investigated, e.g., the negative impact of damping route flaps [8, 9], the number of updates needed for BGP convergence [10], and general conditions for cycle-free IP routes [11].

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. Alternatively, does the interaction need to rely on explicit announcements of content reachability, resembling the BGP announcements in today's Internet, and if so, is the system 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. Furthermore, we investigate how the convergence speed and the cost efficiency of the reached cache configuration are affected by coordination. Finally, we give insight into the structure of the most likely cache allocations in the case of inaccurate estimation of the arrival rate of user requests. We illustrate the analytical results using simulations on the measured peering topology of more than 600 ASes.

The rest of the paper is organized as follows. In Section 2 we describe the system model. In Section 3 we consider caching under perfect information, and in Section 4 we consider the case of imperfect information. In Section 5 we present numerical results, and in Section 6 we review related work. Section 7 concludes the paper.

2 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.

2.1 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[12, 13], 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. 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 the set $\mathcal{H}_i$. Similar to other modeling works, we adopt the Independent Reference Model (IRM) [14, 12, 13] 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 $\gamma_i > \alpha_i$.

2.2 Content-peering

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, before ISP i would forward the interest message to its transit provider, it can leverage its neighbors' caches according to one of two scenarios.

Uncoordinated Content-peering Without coordination, if ISP i finds that an item o is not available locally, it forwards the interest message to all of its neighbors $j \in \mathcal{N}(i)$. If a neighbor has the item in cache, it returns the item to ISP i. If none of the neighbors has the item, ISP i forwards the interest message to its transit provider.

Coordinated Content-peering In the case of coordination peering ISPs *synchronously exchange information* about the contents of their summary caches *periodically*, at the end of every time slot. If, upon an interest message for item o, ISP i finds that item o is not available locally, 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. The total cost can then be expressed as

$$C_i(\mathcal{C}_i, \mathcal{C}_{-i}) = \alpha_i \sum_{\mathcal{L}_i \cup \mathcal{R}_i} w_i^o + \gamma_i \sum_{\mathcal{O} \setminus \{\mathcal{L}_i \cup \mathcal{R}_i\}} w_i^o, \tag{2}$$

which is a function of the cache contents of the peering ISPs $\mathcal{N}(i)$.

2.3 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 [15] 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

$$P(\overline{w}_i^o < \overline{w}_i^p) \propto \epsilon e^{-\frac{1}{\beta}(w_i^o - w_i^p)}. \tag{3}$$

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 [13]. 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 $w_i^p\tau$. The difference $k = \overline{w}_i^o\tau - \overline{w}_i^p\tau$ of two estimates thus follows the Skellam distribution [16] with density function

$$f(k, w_i^o\tau, w_i^p\tau) = e^{-\tau(w_i^o + w_i^p)} \left(\frac{w_i^o}{w_i^p}\right)^{k/2} I_{|k|}(2\tau\sqrt{w_i^o w_i^p}),$$

where $I_{|k|}(.)$ is the modified Bessel function of the first kind. The probability of misestimation is $\sum_{k=-\infty}^{-1} f(k, w_i^o\tau, w_i^p\tau)$, which decreases exponentially in $w_i^o - w_i^p$ for $\tau > 0$.

3 Content-peering under Perfect Information

We start the analysis by considering the case of perfect information, that is, when the cache eviction policies are not prone to misestimation, and we first consider the case of coordinated peering.

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 content-peering could potentially lead to increased costs for the peering ISPs, as shown by the following simple example in which as a consequence of coordination every ISP evicts and fetches the same items repeatedly over transit connections, thereby increasing their traffic costs compared to no content-peering.

Example 1. *Consider two ISPs and $\mathcal{O} = \{1, 2\}$. Let $K_1 = K_2 = 1$. Without content peering both ISPs cache their most popular item and forward interest messages to their transit provider for the least popular item. Their cost is thus $C_i = \alpha_i w_i^{h_i} + \gamma_i w_i^{l_i}$, where $w_i^{h_i} > w_i^{l_i}$. With content peering, if the initial allocation strategies are $\mathcal{C}_1 = \mathcal{C}_2 = \{1\}$, then the cache contents of the ISPs will evolve indefinitely as $(\{1\}, \{1\}) \to (\{2\}, \{2\}) \to (\{1\}, \{1\})$, etc. The average cost for the ISPs is thus $C_i' = \alpha_i \left(\frac{w_i^{h_i} + w_i^{l_i}}{2}\right) + \gamma_i \left(\frac{w_i^{h_i} + w_i^{l_i}}{2}\right) > C_i$.*

This simple example illustrates that content peering could potentially lead to undesired oscillations of the cache contents of the ISPs, with the consequence of increased traffic costs. 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. Such an allocation is 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 defined in (2).

Definition 1. A cache allocation $\mathcal{C}^* \in \times_{i\in N}\mathfrak{C}_i$ is an equilibrium allocation (pure strategy Nash equilibrium) if no single ISP can decrease its cost by deviating from it, that is

$$\forall i \in N, \forall \mathcal{C}_i \in \mathfrak{C}_i : C_i(\mathcal{C}_i^*, \mathcal{C}_{-i}^*) \leq C_i(\mathcal{C}_i, \mathcal{C}_{-i}^*) \tag{4}$$

3.1 Cache-or-Wait (CoW) Algorithm

Example 1 suggests that if one does not allow peering ISPs to update their cache configurations simultaneously, then they would converge to an allocation from which neither of them would have an interest to deviate. In the case of Example 1, such allocations are $(\{1\}, \{2\})$ or $(\{2\}, \{1\})$. Before we describe the Cache-or-Wait (CoW) algorithm, let us recall the notion of an independent set.

Definition 2. 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:

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.

- 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 $\mathcal{C}_i(t)$

Figure 1: Pseudo-code of the Cache-or-Wait (CoW) Algorithm

Definition 3. 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 $p \in E_i(t)$

$$C_i^o(\mathcal{C}(t)) + C_i^p(\mathcal{C}(t)) < C_i^o(\mathcal{C}(t-1)) + C_i^p(\mathcal{C}(t-1)) \tag{5}$$

The requirement of efficiency is rather reasonable. 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. In order to prove that the *efficiency* condition is sufficient for CoW to converge, we will rely on the *generalized group ordinal potential function* defined as follows.

Definition 4. A function $\Psi : \times_i(\mathfrak{C}_i) \to \mathbb{R}$ is a *generalized group ordinal potential function* for the game $< N, (\mathfrak{C}_i)_{i\in N}, (C_i)_{i\in N} >$ if the change of Ψ is strictly positive whenever an arbitrary subset $\mathcal{V} \subseteq \mathcal{I} \in \mathfrak{I}$ of ISPs decrease their costs by changing their strategies,

$$C_i(\mathcal{C}_i', \mathcal{C}_{-i}) - C_i(\mathcal{C}_i, \mathcal{C}_{-i}) > 0, \quad \forall i \in \mathcal{V} \;\; \Rightarrow \;\; \Psi(\mathcal{C}_\mathcal{V}, \mathcal{C}_{-\mathcal{V}}) - \Psi(\mathcal{C}_\mathcal{V}', \mathcal{C}_{-\mathcal{V}}) > 0. \tag{6}$$

Observe that if we define every independent set to be a *singleton*, then the group ordinal potential function Ψ is the *generalized ordinal potential function* defined in [17].
We start constructing a generalized group ordinal potential function by defining the cost saving of ISP i for cache allocation $\mathcal{C}_i$ as $\Psi_i(\mathcal{C}) = C_i(\varnothing, \mathcal{C}_{-i}) - C_i(\mathcal{C}_i, \mathcal{C}_{-i})$. After substituting (2) we obtain

$$\begin{aligned} \Psi_i(\mathcal{C}) &= \alpha_i \sum_{\mathcal{H}_i \cup \mathcal{R}_i} w_i^o + \gamma_i \sum_{\mathcal{O} \setminus \{\mathcal{H}_i \cup \mathcal{R}_i\}} w_i^o - \left[\alpha_i \sum_{\mathcal{C}_i \cup \mathcal{H}_i \cup \mathcal{R}_i} w_i^o + \gamma_i \sum_{\mathcal{O} \setminus \{\mathcal{C}_i \cup \mathcal{H}_i \cup \mathcal{R}_i\}} w_i^o \right] \\ &= \sum_{o \in \mathcal{C}_i \setminus \{\mathcal{H}_i \cup \mathcal{R}_i\}} [\gamma_i - \alpha_i] \, w_i^o. \end{aligned} \tag{7}$$

Note that the value of $\Psi_i(\mathcal{C})$ is not influenced by any item $o \notin \mathcal{C}_i$. We are now ready to prove the following.

Theorem 1. *If every ISP performs efficient updates then the function* $\Psi : \times_i(\mathfrak{C}_i) \to \mathbb{R}$ *defined as* $\Psi(\mathcal{C}) = \sum_{i \in N} \Psi_i(\mathcal{C})$ *increases strictly upon every update and* CoW *terminates in an equilibrium allocation after a finite number of updates.*

Proof. We will start by showing that an efficient update made by any ISP i in the independent set $\mathcal{I}$ strictly increases Ψ_i and cannot decrease Ψ_j of any ISP $j \neq i$, hence Ψ is a generalized group ordinal potential function for efficient updates. Without loss of generality, consider the efficient update $\mathcal{C}_i(t)$ made by ISP $i \in \mathcal{I}_t$ at time slot t. In the following we show that $\Psi_j(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1)) \geq \Psi_j(\mathcal{C}(t-1))$ for all $j \in N$. Observe that from the definition of $\Psi_i(\mathcal{C})$ it follows directly that for ISP i

$$\Psi_i(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1)) > \Psi_i(\mathcal{C}(t-1)).$$

A) Consider $k \notin \mathcal{N}(i)$. Observe that the cost of ISP k is not a function of $\mathcal{C}_i$:

- if $k \notin \mathcal{I}$, ISP k does not make any efficient update at time slot t, thus $\Psi_k(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1)) = \Psi_k(\mathcal{C}(t-1))$;
- if $k \in \mathcal{I}$, $k \neq i$, Ψ_k is not influenced by $\mathcal{C}_i$.

B) Consider $j \in \mathcal{N}(i)$. Consider $o \in I_i(t)$ and $p \in E_i(t)$. From the cost function defined in (1) it follows that $C_i^p(t+1) \geq C_i^p(t)$. Substituting it in the definition of efficient improvement step in (5), it follows that $C_i^o(t) > C_i^o(t+1) \Rightarrow o \notin \mathcal{R}_i(t) \Rightarrow o \notin \mathcal{C}_j(t)$, thus $\Psi_j(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1))$ is not affected by item o.
Consider now item p:

- If $p \notin \mathcal{C}_j(t)$, then $\Psi_j(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1))$ is not affected by item p.
- If $p \in \mathcal{C}_j(t)$, then $\Psi_j(\mathcal{C}_i(t), \mathcal{C}_{-i}(t-1)) \geq \Psi_j(\mathcal{C}(t-1))$ (the inequality is strict if $p \notin \{\mathcal{H}_j \cup \mathcal{R}_j(t+1)\}$).

It follows that the function Ψ increases strictly upon every efficient update. Since $\times_i(\mathfrak{C}_i)$ is a finite set, Ψ cannot increase indefinitely and CoW must terminate in an equilibrium allocation after a finite number of updates. □

The following corollaries are consequences of Theorem 1

Corollary 1. *In the case of coordinated peering under perfect information there is at least one equilibrium allocation.*

Corollary 2. *If every ISP performs efficient updates then the number of time slots needed to reach an equilibrium is finite with probability 1.*

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.

3.2 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. Since the number of independent sets equals at least $\chi(\mathcal{G})$, the chromatic number of the ISP peering graph, an ISP can perform an update on average in every $\chi(\mathcal{G})^{th}$ time slot, in the worst case once every $|N|$ time slots. 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 $\mathcal{C}_i(t)$

Figure 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.*

Proof. Every update of the cache allocation of an ISP is triggered by an interest message sent by a local user. Consider now the arrival of interest messages for item o generated by the local users of ISP i, which has intensity w_i^o. Given that the distribution F_i^o of the inter-arrival times is exponential with parameter w_i^o, there is a non-zero probability $e^{-w_i^o \Delta}$ that item o is not requested during a time slot of length Δ.

Let us consider now a sequence of time slots. For every ISP $i \in \mathcal{I}_t$ there is a positive probability $\epsilon_i(\mathcal{C}_i(t-1))$ that the interest messages generated during time slot t are for items that are either cached locally, are cached by a peering ISP or are not popular enough for being cached. If the cache configuration $\mathcal{C}_i(t-1)$ minimizes ISP i's expected future cost with respect to $\mathcal{C}_{-i}(t-1)$, then $\epsilon_i(\mathcal{C}_i(t-1)) = 1$. Otherwise, with probability $0 < \epsilon_i(\mathcal{C}_i(t-1)) < 1$ ISP i does not update its cache configuration at time slot t, even if in principle its cache configuration could be improved, because no interest message arrives for an item o that could improve its cache configuration. To summarize, at every time slot $t \geq 1$, $\mathcal{C}_i(t)$ is updated according to the following

- $\mathcal{C}_i(t) = \mathcal{C}_i(t-1)$ w.p. $\epsilon_i(\mathcal{C}_i(t-1))$
- $\mathcal{C}_i(t)$ is an efficient update w.p. $1 - \epsilon_i(\mathcal{C}_i(t-1))$.

In the following we use an argument similar to the one used in [18] in order to prove that from every cache configuration $\mathcal{C}(t)$ there is a path to a Nash equilibrium, despite simultaneous cache updates by peering ISPs.

Consider $\mathcal{C}(t-1)$ at the beginning of time slot t. If $\mathcal{C}(t-1)$ is not a Nash equilibrium, then there is a non-zero probability that during time slot t the ISPs that make an update belong to an independent set $\mathcal{I}_t$. This probability is lower bounded by

$$\prod_{i \in \mathcal{I}_t} [1 - \epsilon_i(\mathcal{C}_i(t-1))] \cdot \prod_{i \in N \setminus \mathcal{I}_t} \epsilon_i(\mathcal{C}_i(t-1)) > 0,$$

which is the probability that the ISPs updating their cache configuration during time slot t are exactly $\mathcal{I}_t$. By Theorem 1 we know that if in every time slot it is only ISPs that form an independent set that perform efficient updates, then they reach a Nash equilibrium after a finite number of steps.
Thus there exists a positive probability p and a positive integer T such that, starting from an arbitrary global cache configuration $\mathcal{C}(t)$ at an arbitrary time slot t, the probability that CnW will reach a Nash equilibrium by time slot $t+T$ is at least p. Consequently, the probability that CnW has not reached a Nash equilibrium within a time slot t' is at most $(1-p)^{\frac{t'}{T-1}}$. Observe that $(1-p)^{\frac{t'}{T-1}}$ goes to zero as $t' \to \infty$, and this proves the theorem. □

3.3 Uncoordinated Content-peering

Let us consider now the case of uncoordinated content peering under perfect information. Recall that in this case ISP i does not receive information from ISPs $j \in \mathcal{N}(i)$ about the contents of their caches. As long as an item is cached at ISP i, that is, $o \in \mathcal{C}_i$, the ISP would not forward interest messages to ISPs $j \in \mathcal{N}(i)$, and thus it would not know if o is cached at its neighbors. When ISP i receives an interest message for an item $o \notin \mathcal{C}_i$, it would have to discover whether ISPs $j \in \mathcal{N}(i)$ have item o in their cache by forwarding an interest message to every ISP $j \in \mathcal{N}(i)$. Every discovery of the cache contents of the neighboring ISPs is thus triggered by a cache miss at ISP j. If $o \notin \mathcal{R}_i$, it has to be retrieved over a transit link, and hence ISP i can cache it by evicting an item $p \in \mathcal{C}_i$ if $w_i^p < w_i^o$. Observe that any cache allocation in which no future request can trigger a change is a stable cache allocation under uncoordinated content-peering

Definition 5. $\mathcal{C}_i$ is a *stable cache allocation* for uncoordinated content peering if, $\forall i \in N$ and $\forall o, p \in \mathcal{O}$

$$p \in \mathcal{C}_i, o \notin \{\mathcal{L}_i \cup \mathcal{R}_i\} \Rightarrow w_i^p > w_i^o.$$

For example, the cache allocation where every ISP $i \in N$ is caching the K_i items with highest arrival intensities w_i^o, is a stable allocation for the case of uncoordinated content peering. There can be many other stable allocations, as shown by the following proposition.

Proposition 3. *Every equilibrium allocation under coordinated content-peering is a stable allocation under uncoordinated content-peering.*

Proof. Consider the equilibrium allocation $\mathcal{C}$ under coordinated content-peering. Assume that an interest message for item $o \in \mathcal{O}$ arrives at ISP i. If $o \in \{\mathcal{L}_i \cup \mathcal{R}_i\}$, then ISP i retrieves item o locally or from a peering ISP and its cache configuration does not change. If $o \notin \{\mathcal{L}_i \cup \mathcal{R}_i\}$, then ISP i retrieves item o through a transit link. Let us assume that o is then inserted by ISP i in place of item $p \in \mathcal{C}$, meaning that $\mathcal{C}$ is not a stable allocation under uncoordinated content-peering, and $w_i^o > w_i^p$. Since $o \notin \mathcal{R}_i(t)$, we know from (2) that

$$C_i^o(\mathcal{C}) = \gamma_i w_i^o > C_i^o(\mathcal{C}') = \alpha_i w_i^o \tag{8}$$

Similarly, since $p \in \mathcal{C}_i$ we have that $C_i^p(\mathcal{C}) = \alpha_i w_i^p$. Now consider the following two cases:

- $p \in \mathcal{R}_i(t)$, then $C_i^p(\mathcal{C}') = C_i^p(\mathcal{C}) = \alpha_i w_i^p$.
- $p \notin \mathcal{R}_i(t)$, then $C_i^p(\mathcal{C}') = \gamma_i w_i^p$.

Putting these together we obtain

$$C_i^o(\mathcal{C}') + C_i^p(\mathcal{C}') < C_i^o(\mathcal{C}) + C_i^p(\mathcal{C}), \tag{9}$$

which means that, by Definition 3, $\mathcal{C}_i'$ is an efficient update. Since ISP i was able to make an efficient update in cache configuration $\mathcal{C}$, it follows that $\mathcal{C}$ could not have been an equilibrium allocation under coordinated content-peering, which contradicts our initial assumption. This proves the proposition. □

We can easily show that the converse is not true: the cache allocation $(\{1\}, \{1\})$ in Example 1 is not an equilibrium allocation under coordinated content-peering, but since $w_i^1 > w_i^2$, no ISP will insert item 2 in its cache under uncoordinated peering and thus it is a stable allocation.

Given a stream of interest messages and perfect information, in the following we address whether caching without coordination converges to a stable allocation. For the analysis, we define the *instantaneous download* (ID) assumption, under which after a cache miss the file is instantaneously downloaded into the cache, *before* another interest message could arrive. The ID assumption is common practice in the caching literature [12, 13], and as we show, it allows fast convergence to a stable allocation.

Proposition 4. *Under the ID assumption uncoordinated peering reaches a stable cache allocation after a finite number of cache updates.*

Proof. Consider an arbitrary cache update of ISP $i \in N$. Consider an interest message that arrives for object $o \notin \{\mathcal{L}_i \cup \mathcal{R}_i\}$, and assume that ISP i updates its

cache configuration from $\mathcal{C}_i$ to $\mathcal{C}'_i$ inserting item o in place of item p. Recall from the proof of Proposition 3, that $\mathcal{C}'_i$ is an efficient update. The ID assumption guarantees that ISP i is the only ISP that updates its cache configuration in this time instant. Therefore we know from Theorem 1 that the global function $\Psi : \times_i(\mathfrak{C}_i) \to \mathbb{R}$ strictly increases at every cache update. Since $\times_i(\mathfrak{C}_i)$ is a finite set, Ψ cannot increase indefinitely and a stable allocation is reached after a finite number of updates. This proves the proposition. □

With the ID assumption uncoordinated peering converges in a finite number of updates, similar to *CoW*. Without this assumption the convergence is much slower.

Proposition 5. *Without the ID assumption uncoordinated peering reaches a stable cache allocation with probability one.*

Proof. Given that the arrival processes of the interest messages at different ISPs are independent, there is a positive probability $\epsilon > 0$ that at some point in time ISP i is the only ISP that needs to update its cache, and that it can do so before another interest message arrives. Following the same arguments as in the proof of Theorem 2, it can be shown that the probability of reaching a stable cache allocation after t time units approaches 1 when $t \to \infty$. □

We have thus far shown that under perfect information both coordinated and uncoordinated content-peering will lead to equilibrium, alternatively stable cache allocations. We now turn to the case of imperfect information.

4 The case of Imperfect Information

Until now we assumed that following a cache miss, when ISP i has to decide whether to cache item o it has a perfect estimate of the arrival intensity w_i^o of every item, and thus it is always able to evict one of the least popular items. In the following we consider that the estimation of the item popularities is imperfect. We consider CoW throughout the section for two reasons. First, under the ID assumption uncoordinated content-peering behaves very similar to CoW. Second, the set of equilibria under CoW coincides with those under CnW.

Under imperfect information 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, and in the following we show that it is possible to characterize them.

We model the evolution of the system state (the set of cached items) under imperfect information by a regular perturbed Markov process. Consider first the case of coordinated peering under perfect information and the CoW algorithm. For the following analysis we can restrict ourselves to the case that each independent set $\mathcal{I}$ is a singleton. The evolution of the system state can be modeled by a Markov chain P^0; the transition probability between states $\mathcal{C}$ and $\mathcal{C}'$ is non-zero if and only

if $\mathcal{C}' = (\mathcal{C}'_i, \mathcal{C}_{-i})$ for some i such that the update $\mathcal{C}_i$ to $\mathcal{C}'_i$ by ISP i is an efficient update. Observe that the Markov chain P^0 is not irreducible, as every equilibrium state is an absorbing state. We refer to this chain as the unperturbed process.

Consider now coordinated peering under imperfect information. Under imperfect information the probability that item o will be evicted and item p inserted even though $w_i^o > w_i^p$ is non-zero as given in (3). The resulting Markov process P^β is a regular perturbed Markov process of P^0 [19], because for every $\beta > 0$ it is irreducible and for every $\mathcal{C}$ and $\mathcal{C}'$, P^β converges to P^0 at an exponential rate, $\lim_{\beta\to 0} P^\beta_{\mathcal{C},\mathcal{C}'} = P^0_{\mathcal{C},\mathcal{C}'}$. Furthermore, if $P^\beta_{\mathcal{C},\mathcal{C}'} > 0$ for some $\beta > 0$ then for some $r(\mathcal{C},\mathcal{C}') \geq 0$

$$\lim_{\beta\to 0} e^{\frac{1}{\beta} r(\mathcal{C},\mathcal{C}')} \cdot P^\beta_{\mathcal{C},\mathcal{C}'} = \epsilon > 0. \tag{10}$$

We refer to $r(\mathcal{C},\mathcal{C}') \geq 0$ as the resistance of the transition from allocation $\mathcal{C}$ to $\mathcal{C}'$. The resistance is 0 if there is a transition in the unperturbed Markov process. Since P^β is an irreducible aperiodic finite Markov process, it has a unique stationary distribution for $\beta > 0$. We now recall a result from Young [19].

Lemma 1 (Young [19])**.** *Let P^β be a regular perturbed Markov process, and let μ^β be the unique stationary distribution of P^β for each $\beta > 0$. Then $\lim_{\beta\to 0} \mu^{e^{-\frac{1}{\beta}}} = \mu^0$ exists, and μ^0 is a stationary distribution of P^0. The domain of μ^0 is a non-empty subset of the absorbing states of the unperturbed Markov process.*

By Lemma 1 there is thus a stationary distribution μ^0 of the unperturbed process such that, for small β, the system will likely be in a state in the domain of μ^0. In the rest of the section we try to infer what cache allocations are most likely to occur in the following scenario. Consider a set of $N = \{1, \ldots, |N|\}$ ISPs, and items $\mathcal{O} = \{1, \ldots, |\mathcal{O}|\}$. Let $\rho_i(o)$ be the rank in terms of popularity of item o in ISP i, and let $\mathcal{T}_i$ be the set of the K_i items such that $\rho_i(o) \leq K_i$. For a cache allocation $\mathcal{C}$ denote by $h(\mathcal{C})$ the number of items o such that o is cached by an ISP i but $\rho_i(o) > K_i$.
We consider that the items with highest arrival intensity are the same among the different ISPs, and we denote them by the set $\mathcal{T} = \bigcup_i \mathcal{T}_i$. We start by investigating the cache allocations that are most likely to occur in the case of disjoint interests. In this case the K_i items with highest arrival intensity of the ISPs form disjoint sets, namely $\mathcal{T}_i \cap \mathcal{T}_j = \varnothing$, for all $i \neq j \in N$. We will first show the following

Lemma 2. *Let $\mathcal{C}^*$ be the allocation in which every ISP caches its most popular items, namely $\mathcal{C}^*_i = \mathcal{T}_i$. For any absorbing state $\mathcal{C}'$ such that $h(\mathcal{C}') > 2$, there exists an absorbing state $\mathcal{C}''$ such that $h(\mathcal{C}'') = 2$ and $r(\mathcal{C}^*,\mathcal{C}'') < r(\mathcal{C}^*,\mathcal{C}')$.*

Proof. Let $\mathcal{S}$ be the path with least resistance from $\mathcal{C}^*$ to $\mathcal{C}'$. Observe that, since $\mathcal{C}^*_i = \mathcal{T}_i$, at least $h(\mathcal{C}')$ mistakes are needed to reach $\mathcal{C}'$. Denote by i the first ISP that makes a mistake in $\mathcal{S}$, and by o and q the mistakenly evicted and inserted items, respectively. Since $\mathcal{S}$ is the path with least resistance, there exists $j \in \mathcal{N}(i)$

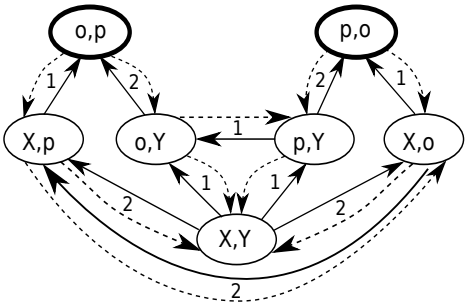

Figure 3: State transition diagram of the unperturbed Markov process (solid lines). (o,p) and (p,o) are absorbing states in the unperturbed Markov process, but only the equilibrium (o,p) is the domain of μ^0.

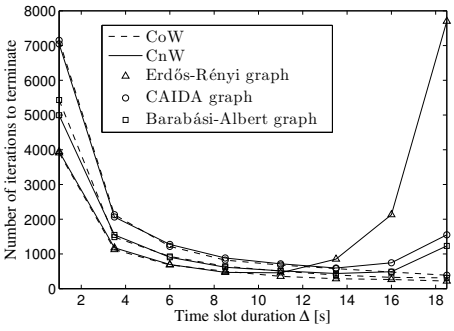

Figure 4: Average number of iterations needed to reach an equilibrium allocation as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

that makes at least one mistake. Consider the first mistake of ISP j and call p and r the evicted and inserted item, respectively. Observe that $o,p \in \mathcal{T}_i \cup \mathcal{T}_j$. We will now show that these two mistakes are enough to reach the absorbing state $\mathcal{C}''$ defined as $\mathcal{C}''_j = \mathcal{C}^*_j \setminus \{p\} \cup \{o\}$, $\mathcal{C}''_i = \mathcal{C}^*_i \setminus \{o\} \cup \{p\}$, $\mathcal{C}''_h = \mathcal{C}^*_h \ \ \forall h \in N \setminus \{i,j\}$, and hence $r(\mathcal{C}^*,\mathcal{C}'') < r(\mathcal{C}^*,\mathcal{C}')$. Let us start from $\mathcal{C}^*$ and consider the state reached after committing the two mistakes. Observe that, since $q \notin \mathcal{T}_i \cup \mathcal{T}_j$, then $\rho_i(p) < \rho_i(q)$. Furthermore we know that there is no ISP $h \neq j$, such that $p \in \mathcal{C}^*_h$. Hence ISP i can evict q and insert p without making a mistake. If $r = o$ then we reached $\mathcal{C}''$. If $r \neq o$ then, following the same argument, ISP j can insert o and evict r without making a mistake, reaching $\mathcal{C}''$. □

We will now use Lemma 2 to prove the following

Proposition 6. *If $\mathcal{T}_i \cap \mathcal{T}_j = \varnothing$ for all $i \neq j \in N$, then $\lim_{\frac{1}{\beta}\to\infty} P(\mathcal{C}(t) = \mathcal{C}^*) = 1$.*

Proof. As a consequence of Lemma 2 it is sufficient to show that for every absorbing state $\mathcal{C}''$ such that $h(\mathcal{C}'') = 2$, it holds that $r(\mathcal{C}^*,\mathcal{C}'') > r(\mathcal{C}'',\mathcal{C}^*)$. For brevity define $\mathcal{C}''$ as in the proof of Lemma 2. Assume, w.l.o.g., that in the path with least resistance from $\mathcal{C}^*$ to $\mathcal{C}''$, ISP i makes a mistake before ISP j by inserting item $q \notin \mathcal{T}_i \cup \mathcal{T}_j$ in place of item o . Then $r(\mathcal{C}^*,\mathcal{C}'') > w_i^o - w_i^q$. Observe now that, since $q \notin \mathcal{T}_i \cup \mathcal{T}_j$, from the absorbing state $\mathcal{C}''$ the mistake of ISP i of evicting item p and inserting q, with resistance $w_i^p - w_i^q$, is enough to reach $\mathcal{C}^*$. Hence $r(\mathcal{C}'',\mathcal{C}^*) \leq w_i^p - w_i^q$. This proves the Proposition. □

The following illustrates the proof on a simple example.

Example 2. *Consider a complete graph and $K_i = 1$. The $|N|$ most popular items are the same in every ISP, but item o has a distinct rank at every ISP. In every*

equilibrium the $|N|$ most popular items are cached, one at every ISP, and thus there are $|N|!$ equilibria. Fig 3 shows the state transition diagram of the unperturbed Markov process (with solid lines) for the case of two ISPs, $|N| = 2$. The figure only shows the transitions between *states. X and Y stand for an arbitrary item other than o and p, and the states (p, Y) and (X, p) ((o, Y) and (X, o)) represent all states in which item p (item o) is cached by ISP 1 and ISP 2, respectively. The dashed lines show transitions due to mistakes that are needed to move from one equilibrium to a state from which both equilibria are reachable (there is a positive probability of reaching it) in the unperturbed process. These transitions only exist in the perturbed Markov process. With perfect information there are two equilibrium allocations, which are the absorbing states (o, p) and (p, o) of the unperturbed process. The two equilibrium allocations are, however, not equally likely to be visited by the perturbed process.*

Observe that in the unperturbed process, equilibrium (o, p) is reachable from every allocation except from equilibrium (p, o). Therefore, in the perturbed process one mistake suffices to leave equilibrium (p, o) and to enter a transient state of the unperturbed process from which both equilibria are reachable in the unperturbed process. It takes, however, two mistakes in close succession to leave equilibrium (o, p) and to enter a transient state of the unperturbed process from which both equilibria are reachable in the unperturbed process. As β decreases, the probability of two successive mistakes decreases exponentially faster than that of a single mistake, and thus the perturbed process will be almost exclusively in state (o, p), thus $\mathcal{C}^* = (o, p)$.

A similar reasoning can be used to get insight into the evolution of the system state in the case that the ranking of the items is the same among all ISPs, namely $\mathcal{T}_i = \mathcal{T}_j$ for all $i, j \in N$. As an example, we show the following.

Proposition 7. *If the arrival intensity w_i^o for an item o for which $\rho_i(o) \leq K_i$ increases at ISP i, then $\lim_{\beta \to 0} P(o \in \mathcal{C}_i(t))$ increases.*

Proof. Consider the state transition diagram of the perturbed Markov process P^β. For a state $\mathcal{C}$ for which $o \in \mathcal{C}_i$, the transition probability that corresponds to ISP i mistakenly evicting o decreases. For a state $\mathcal{C}$ for which $o \notin \mathcal{C}_i$, the transition probability to the states $\mathcal{C}'$ for which $o \in \mathcal{C}_i'$ increases, and the transition probability to other states decreases. Reconciling these changes with the global balance equation for the set of states $\{\mathcal{C} | o \in \mathcal{C}_i\}$ proves the proposition. □

The impact of the number of peers of an ISP and that of the amount of storage K_i can be analyzed similarly, but we omit the analysis due to lack of space.

5 Numerical Results

In the following we show simulation results to illustrate the analytical results of Sections 3 and 4 for CoW and CnW.

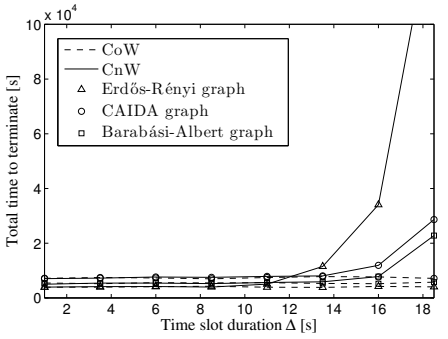

Figure 5: Average time needed to terminate as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

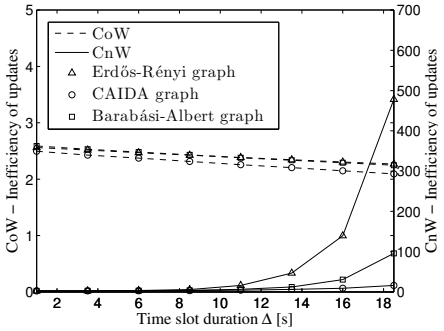

Figure 6: Average inefficiency as a function of the time slot duration Δ for three different peering graphs and algorithms CoW and CnW.

5.1 Perfect Information

Figures 4 and 5 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 [20]. The dataset contains 36878 ASes and 103485 transit and peering links between ASes as identified in [21]. 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. For the CoW algorithm, we used the Welsh-Powell algorithm to find a coloring [22] 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 4 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 like the ones shown in Example 1 cannot occur. Consistently, the total time needed for the CoW algorithm to converge, shown in Figure 5, 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 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 4 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 5 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 6 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.

5.2 Imperfect Information

In the following we show results for the case when the estimation of the items' arrival intensities is imperfect. We consider that every ISP estimates the arrival intensities of the items by counting the number of arrivals under a period of τ seconds. As in the case of imperfect information the CoW algorithm would never terminate, we collected the statistics on the permanence of the various items in the cache of each ISP over 10^5 time slots. We considered 50 ISPs and a time slot of 70 seconds, which in the case of perfect information would guarantee a fast termination of the CoW algorithm. We first validate Proposition 6 for the case of $K_i = 1$, hence we consider that the item with the highest arrival intensity is different at every ISP. Figure 7 shows the average relative permanence in the ISPs' caches of the three items with highest arrival intensity, as a function of the estimation interval τ, for three random peering graphs. The results show that the probability of caching the item with highest arrival intensity approaches 1 when τ increases, and thus validate Proposition 6. Furthermore we observe that the probability of caching items with lower arrival intensities decreases exponentially with τ.

In the next scenario we start from the setting described in Proposition 7, where the ranking of the items' intensities is the same among all ISPs. We scale the arrival intensity w_1^o of every item o at ISP 1 by the same factor, while keeping the intensities at the other ISPs constant. Figure 8 shows the average relative permanence in

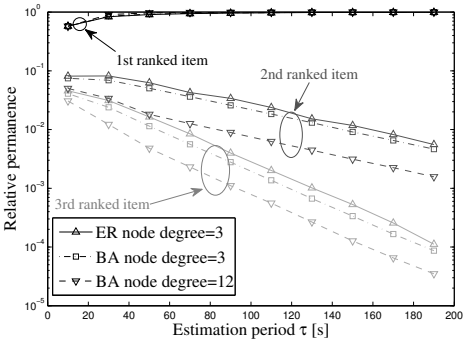

Figure 7: Relative permanence of the three items with highest arrival intensity in the ISPs' caches, as a function of the intensity estimation interval τ, for three different random peering graphs.

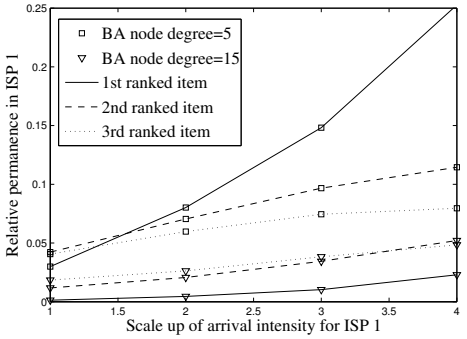

Figure 8: Relative permanence of the three items with highest arrival intensity in ISP 1's cache, as a function of the scaling factor at ISP 1, for two Barabási-Albert peering graphs with different average node degree.

ISP 1's cache of the three items with highest arrival intensities as a function of w_1. The results confirm that a higher w_1 leads to a higher relative permanence in the ISP's cache of the items with highest arrival intensity. Concerning the influence of the peering graph, the figure shows a constantly lower permanence of the best items for the BA graph with higher average node degree. This is due to that with a higher number of peering links the probability that the best items are in a peering ISP's cache gets higher.

6 Related Work

There is a large variety of cache eviction policies from Least recently used (LRU) to the recent Adaptive replacement cache [15]. Most analytical work on the performance of cache eviction policies for stand-alone caches focused on the LRU policy [23, 24, 13]. An iterative algorithm for calculating the cache hit rate was proposed in [23], closed-form asymptotic results were provided for particular popularity distributions in [24] and recently in [13]. These works considered stand-alone caches.

The cache hit rate for cache hierarchies was investigated in the context of web caches and content-centric networks [25, 26, 27, 28]. General topologies were considered for content-centric networks [4, 12, 6]. An iterative algorithm to approximate the cache miss rate in a network of caches was proposed in [12]. The authors in [4] considered various network topology-aware policies to improve the overall cache hit rate in a network of caches. In [6] the authors probabilistic caching to increase the cache hit rate in a network of caches. These works consider that the caches route requests irrespective of the associated traffic costs, and assume a single net-

work operator with a single performance objective. In our work we account for the profit maximizing behavior of individual network operators and model the resulting interaction between caches.

Replication for content delivery in a hierarchy of caches was considered recently in [29, 30]. The authors in [29] considered a centralized algorithm for content placement, while distributed algorithms were analyzed in [30]. A game theoretical analysis of equilibria and convergence for the case of replication on an arbitrary topology was provided in [31]. Opposed to replication, we consider an arbitrary topology of caches, and we consider that caches do not follow an algorithm engineered for good global performance but they follow their individual interests.

To the best of our knowledge ours is the first work that considers a network of selfish caches, including the effects of evictions, and provides a game-theoretical analysis of the resulting cache allocations.

7 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. However, in order for the resulting stable cache configurations to be efficient in terms of cost, the ASes would have to periodically exchange information about the content of their caches. If fast convergence to a stable allocation is important too then synchronization is needed to avoid simultaneous cache evictions by peering ISPs. Furthermore, we showed that if the content popularity estimates are inaccurate, content-peering is likely to lead to cost efficient cache allocations, and we gave insight into the structure of the most likely cache allocations.

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] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz, "Route Flap Damping Exacerbates Internet Routing Convergence," in *Proc. of ACM SIGCOMM*, vol. 32, no. 4, 2002, p. 221.

[9] S. Agarwal, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "The impact of BGP dynamics on intra-domain traffic," in *Proc. of ACM SIGMETRICS*, vol. 32, no. 1, 2004, p. 319.

[10] R. Sami, M. Schapira, and A. Zohar, "Searching for Stability in Interdomain Routing," in *Proc. of IEEE INFOCOM*, 2009, pp. 549–557.

[11] L. G. L. Gao and J. Rexford, "Stable Internet routing without global coordination," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 681–692, 2001.

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

[13] 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.

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

[15] N. Megiddo and D. Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache," in *Proc. of USENIX File & Storage Technologies Conference (FAST)*, 2003, pp. 115 – 130.

[16] 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.

[17] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, pp. 124–143, 1996.

[18] H. P. Young, *Strategic learning and its limits*, 2004.

[19] ——, "The evolution of conventions," *Econometrica: Journal of the Econometric Society*, vol. 61, no. 1, pp. 57–84, 1993.

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

[21] 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.

[22] 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.

[23] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes," in *Proc. of ACM SIGMETRICS*, vol. 18, no. 1, 1990, pp. 143–152.

[24] P. R. Jelenković, "Asymptotic Approximation of the Move-to-Front Search Cost Distribution and Least-Recently Used Caching Fault Probabilities," *Annals of Applied Probability*, vol. 9, no. 2, pp. 430–464, 1999.

[25] M. Busari and C. Williamson, "Simulation Evaluation of a Heterogeneous Web Proxy Caching Hierarchy," in *Proc. of MASCOTS*, 2001, p. 379.

[26] H. Che, Z. Wang, and Y. Tung, "Analysis and design of hierarchical Web caching systems," in *Proc. of IEEE INFOCOM*, vol. 3, 2001, pp. 1416–1424.

[27] C. Williamson, "On filter effects in web caching hierarchies," *ACM Trans. Int. Tech.*, vol. 2, no. 1, pp. 47–77, 2002.

[28] I. Psaras, R. G. Clegg, R. Landa, W. K. Chai, and G. Pavlou, "Modelling and evaluation of CCN-caching trees," in *Proc. of IFIP Networking*, 2011, pp. 78–91.

[29] M. Korupolu and M. Dahlin, "Coordinated placement and replacement for large-scale distributed caches," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 6, pp. 1317–1329, 2002.

[30] S. Borst, V. Gupta, and A. Walid, "Distributed Caching Algorithms for Content Distribution Networks," in *Proc. of IEEE INFOCOM*, 2010, pp. 1478–1486.

[31] V. Pacifici and G. Dán, "Selfish Content Replication on Graphs," in *Proc. of the 23rd International Teletraffic Congress*, 2011, pp. 119–126.