Future Generation Computer Systems 55 (2016) 116–128

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Multi-provider cloud computing network infrastructure optimization

Thepparit Banditwattanawong [a,1], Masawee Masdisornchote [a,*,1], Putchong Uthayopas [b]

[a] School of Information Technology, Sripatum University, Bangkok, Thailand
[b] Computer Engineering Department, Kasetsart University, Bangkok, Thailand

HIGHLIGHTS

- Byte-hit rate and hit rate could be optimal simultaneously in a nonuniform cost model.
- i-Cloud outperformed popular LRU, GDSF and LFUDA schemes in a nonuniform cost environment.
- i-Cloud's performances were stable and close to those of infinite cache size.
- Window size had small performance effect when relative cache sizes were big.
- Accounting data-out charge rates improved all performance aspects at small cache sizes.

ARTICLE INFO

Article history:
Received 24 February 2014
Received in revised form
3 May 2015
Accepted 2 September 2015
Available online 10 September 2015

Keywords:
Cache replacement
Multi-provider cloud
Federated cloud
Hybrid cloud
Artificial neural network
Cost-saving ratio

ABSTRACT

Cloud-adopting enterprises have been increasingly employing multiple cloud providers concurrently, for example, to consume unique services and to mitigate data lock-in risk. As a consequence, the enterprises must be able to address contrasting quality-of-service degrees offered by the different providers. This paper presents an intelligent cloud cache eviction approach, namely i-Cloud, as the core component of a client-side cloud cache. i-Cloud is capable of reducing public cloud data-out expenses, improving cloud network scalability and lowering cloud service access latencies specifically in multi-provider cloud environments. Trace-driven simulations have shown that i-Cloud outperformed well-known approaches in all performance metrics. In addition, i-Cloud is not only able to achieve optimal performances in all metrics simultaneously but also delivered relatively stable performances across all performance metrics. The results have also indicated that taking the nonuniformity of data-out charge rates into cache eviction decisions improved caching performances in all metrics.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Big data has been increasingly hosted in clouds as in, for example, Facebook, Youtube, Street View Google Maps, iTune Store. This is because cloud enables not only fast, economical and greener big data analytics [1–3] but also highly-scalable and distributed sharing of the data. This is the evolution of data volume, data placement and data consumption behavior that altogether has led both practitioners and researchers to address several new cloud-computing problems including the downstream bandwidth saturation of network connections between external cloud and consumer premises, increase in external private-cloud data-out charge imposed by public cloud providers (e.g., [4–6]) and long-delayed cloud service responsiveness. These difficulties have been recognized in [7–10] as the forms of data transfer bottlenecks, data transfer costs or cloud computing economics and SaaS SLA responsiveness. A straightforward way to handle these problems is network bandwidth upgrade, which unfortunately impedes cloud economy. A wiser means is disk shipping by overnight delivery services [7,8] that is only applicable for delay-tolerant cloud services. Another solution to meet these scalability, economy and responsiveness requirements of cloud computing services at the same time is the consumer-initiated partial replication of cloud-hosted data to consumer (nearby) locus as in Amazon CloudFront [11], which offers a caching service through content delivery network. We refer to this solution as client-side cloud caching.

Client-side cloud caches are located in or nearby consumer premises where HTTP requests to external private clouds are proxied by the cloud caches, which in turn reply with the valid copies of the requested data objects either from their local storages

* Corresponding author.
E-mail address: masawee.ma@spu.ac.th (M. Masdisornchote).
[1] Sripatum university, 2410/2 Phaholyothin Road, Jatujak, Bangkok 10900, Thailand.

http://dx.doi.org/10.1016/j.future.2015.09.002
0167-739X/© 2015 Elsevier B.V. All rights reserved.

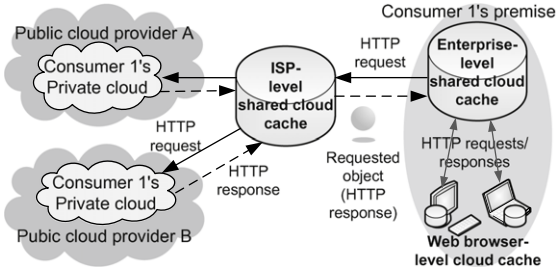

Fig. 1. Cloud cache deployment scenario in a multi-provider cloud environment.

(i.e., cache hits) or by downloading updated copies from the clouds (i.e., cache misses). The cloud caches are basically evolved from traditional forward web caching proxies since cloud data is also delivered by using the same set of HTTP/TCP/IP protocol stacks as in WWW. Unavoidably, the same limitation as in web caching is also shared in the cloud caches that is caching entire remote data in local cache storage is not economically sensible, thus a cache replacement policy is also mandatory for the cloud caches. In fact, client-side cloud caches can also be deployed in a hierarchical structure as portrayed in Fig. 1: the web-browser level cloud caches are nonshared by individuals whereas the enterprise- and ISP-level cloud caches are shared. This paper focuses on a client-side shared cloud cache at the enterprise level. Another main characteristic of the client-side shared cloud cache is multi-tenancy support whereby workloads initiated by different enterprises are shared for hardware but isolated for user data. The other properties of the client-side shared cloud cache lie in its various Cache-as-a-Service (CaaS) models [12] (e.g., RAM Multitenancy Isolated database, and SSD Multitenancy Isolated database), which are not offered by the traditional web caching proxies.

As illustrated in Fig. 1, an enterprise-level cloud cache can be deployed in a multi-provider cloud environment where a consumer enterprise employs multiple concurrent cloud providers to host its external private clouds that are either independent from each other [13,14] or interconnected as a hybrid cloud [15]; both are the forms of federated cloud [16,17]. There are several benefits of multi-provider cloud deployment such as load balancing [18,19], enabling of planned downtime for system maintenance [17], confidentiality protection [20], risk mitigation (mandated by hospitals, stock markets, air transportation controls, etc.) [20,21] and the utilization of unique capabilities offered by different cloud providers [13]. Nevertheless, the deployment of multiple cloud providers usually imposes different levels of QoS perceived by consumers in the forms of unbalancing downstream network throughputs and nonuniform cloud data-out charge rates. This complicates cloud caching optimization in that cloud cache replacement must be aware of QoS heterogeneity to achieve both efficiency and economy. To accomplish such optimization, our extensive investigation of related work in Section 6 gives several reasons that we cannot simply exploit existing web cache eviction techniques. The other reason for that client-side cloud cache replacement policies must be designed differently from the traditional WWW ones lies in the intrinsic characteristic of cloud computing data itself: Cloud data objects (e.g., big data) have larger sizes than traditional WWW objects [22]. To our current experiences, requesting for big objects made available via cloud computing services (e.g., cloud storages) still keep users waiting for long whereas downloading small objects from clouds nowadays is fast as if the objects were fetched from user locality. This means that priority must be given to the loading optimization of big objects whereas existing web cache replacement policies had been totally optimized for small objects [23,24] (because, in the past, fetching small data object was so delayed due to slow Internet connection that it was unacceptable for users to experience such delays on every request).

As an early attempt in the field of enterprise-level client-side shared cloud caching towards multi-provider clouds, this paper presents an intelligent cloud cache replacement policy, i-Cloud (named so for its intended application domain), along with its technical and economical performances and important findings.

The organization of this paper is as follows. Section 2 explains monetary cost models as a basis for multi-provider cloud employment. Section 3 explains our experimental data sets and how the raw traces were preprocessed to obtain training and evaluation data sets that realistically represent the traffics of shared cloud networks. i-Cloud is described in detail in Section 4. Section 5 discusses comparative performance results and presents algorithmic success factor analysis to formulate an open question for future research in this new field of study. We extensively contrast and compare this work with relevant ones in Section 6. Section 7 draws the conclusion of crucial findings.

2. Monetary cost models

When an organization employs cloud computing, there are two possible monetary cost models [25] depending on the number of contracted cloud providers. If a single provider is contracted, a uniform cost model is employed in which object requests made by the organization entirely go to the same cloud provider, thus a single data-out charge rate is applied to all of the requested objects. Otherwise, the other nonuniform cost model is applied in which object requests separately go to multiple cloud providers, offering different data-out charge rates. In other words, objects retrieved from the same cloud service domain were always charged by the same provider.

Since this paper aims for multi-provider clouds, both i-Cloud training (Section 4.2.2) and i-Cloud evaluation (Section 5) are based on the nonuniform cost model. In particular, we assumed that an organization's cloud-hosting domains were hosted separately by two contracted public cloud providers, who offer different data-out charge rates, thus a pair of monetary costs associated with data-out transfers. The first provider was Google and thus its charge rate was set to 0.1535 USD/GB while the other provider was AWS and its charge rate was set to 0.0829 USD/GB.

Both of the flat charge rates were converted from their original regressive rates (i.e., Google Cloud Storage's network egress charge in Asia-Pacific region as of September 2013 [5] and Amazon S3's data transfer out to Internet charge (US Standard) as of September 2013 [4]) by means of weighted averages: First, we determined the total amount of data transferred out of each provider to which a regressive rate was applied. Once, we knew the total data-out volume and the range of corresponding regressive rates, we can figure out the total expense and consequently the flat charge rate in USD/GB for such provider. The total data-out volume was realistically assumed based on a representative scenario where an organization utilizing data residing in cloud by transferring it out of the cloud through 10 Gbps Metro Ethernet with 50% average downstream bandwidth utilization for 8 work hours a day requires the total amount of cloud data-out transfer 4570.31 TB per year (as of 260 workdays per year) or 380.86 TB per month; we also assumed for the ease of understanding that the workload 380.86 TB per month is distributed equally among the two public cloud providers to emulate a multi-provider cloud circumstance, thus a data-out volume is 190.43 TB per provider-month. This split volume costs 29 925.88 USD/month for one provider and 14 704.74 USD/month for the other provider making the organization liable to pay for cloud data-out transfer totally 44 630.62 USD/month.

Download English Version:

https://daneshyari.com/en/article/425563

Download Persian Version:

https://daneshyari.com/article/425563

Daneshyari.com