

Latency-Aware Cache Mechanism for Resolver Service of Domain Name Systems

Ibirisol Fontes Ferreira
Graduate School of Informatics
Kyoto University
Kyoto, Japan
ferreira.ibirisol.d00@kyoto-u.jp

Eiji Oki
Graduate School of Informatics
Kyoto University
Kyoto, Japan
oki@i.kyoto-u.ac.jp

Abstract

The domain name system (DNS) is essential to access Internet services. However, although DNS has become a vital part of Internet users in the network and facilitates the process, it is still a critical part of the service chain. This is principally due to its resolver component, which can negatively impact the final user experience of consuming services. Furthermore, to improve service quality and increase user experience, this study intends to mitigate the latency effect of the resolution element of domain name systems in the network. Some studies have developed strategies to reduce response time to resolution. Still, they needed some integrated module in the user's devices or a more complex and expensive service architecture that was not scalable for edge deployments. This study proposes integrating prediction techniques and a user-oriented resolution name approach to improve the overall quality of access to the Internet for users, even when there are limited computing resources in the edge infrastructure.

Index Terms

Caching, domain name systems, quality of experience, service architecture, heuristics, machine learning

I. INTRODUCTION

When an Internet user tries to access a service (web page, e-mail, instant messaging, etc.), their system triggers a connection process to reach the intended service and get the desired content. The name resolution occurs before the service access, and it translates service names into network addresses using the resolver, the intermediate element of the domain name system (DNS) [1], [2] that facilitates the translation procedure.

The DNS is a distributed database that keeps the Internet domain translation data stored in distributed servers, which deliver a portion of the data under their authority when requested by the resolver. Although DNS resolution is a simple process, it can significantly increase the response time to the desired service [3], [4], thus delivering an impaired quality of experience (QoE). Despite the importance of the Internet for DNS and the trending adoption of edge computing technologies such as mobile edge computing (MEC), DNS has become an even more vital component for accessing services hosted on the network's edge, playing a critical role in determining service access latency [5], [6].

Fig. 1 shows the process triggered by the user when they make a service (application) request through a DNS client (stub resolver) to the local infrastructure resolver server. The response time of the DNS resolver (represented by the height of the DNS resolution time bars), due to queries involved in the process, can drastically impact the total time that users will effectively receive the requested service (total response time).

To mitigate the resolution time, the resolver server stores the responses from previous queries in a local cache to shorten the process. However, these data in the cache are valid only for a time-to-live (TTL) period predetermined by the domain's authoritative server. After this period, a local cache consistency mechanism stale the query data when the TTL expires.

When the DNS resolver cannot reuse the cache or the domain has no answer stored locally (cache miss), it needs to start the recursive query process from the beginning only for that missed entry. Therefore, there is a drastic impact on user access to the desired service [7]–[11]. Among the reasons for cache misses occurring in the local resolver are some technical issues, such as cache pollution by one-time-only entries [12]–[14] that exhaust computational resources, domain with a wrong TTL policy that prematurely [15] or unnecessarily [11] affects the entries evictions from the cache. Security problems related to attacks on the nonexistent domain (NXDomain) intended to overload servers [16]. In addition to the above issues, the location of the resolver in the Internet infrastructure can also have an impact on the final result as perceived by the user, as well as on the entries stored in the cache [17]–[19]. Therefore, it reminds us of the importance of the local DNS resolver, which is crucial for efficient resource management to reduce the impact on user perceived time in using the service [20]. The regional aspect of the resolvers is a challenge even for large-scale public DNS resolver services [21], [22].

Faced with these issues directly impacting local DNS servers, this study proposes to address a resolver system architecture that manages the local cache data intended to reduce the impact of resolver latency on user access time to Internet services by

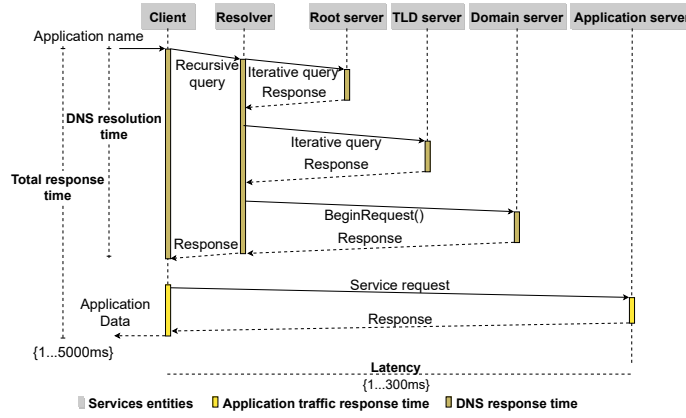


Fig. 1. DNS query flow during service access process.

designing a DNS cache management policy. This system aims to predict domains that the resolver must hold or request before its user claims the service resolution data. Therefore, this proposed study will focus on the quality of resolver performance to improve access to Internet services by users on edge networks.

II. RELATED WORKS

Among all components for the construction of computing and networking systems, the caching component calls research attention mainly because it helps either reduce the traffic to fetch data, decrease latency to access these data, or reduce the cost related to moving the data from the source to destination, mainly when many users request the same content. It can be applied in various domains to produce systems with a better user experience [23]–[25].

Cache management can be divided into several main methods, which can generally take care of cache policy in admission, eviction, or both. The most fundamental are bounds models and statistical hypothesis solutions, which, by conveying the expected behavior of the cache demand and caching working, bring optimal solutions for defined scenarios and assumptions [26]–[28] even though most focus on cache hit ratio as a metric and usually do not consider changes in distribution or application characteristics.

In contrast, heuristic-based algorithms typically have a lower computational complexity for workload-dependent cache operations, although they perform badly in general cases in storing required entries. Conventional examples are last-in-first-out (LIFO), first-in-first-out (FIFO), least frequently used (LFU), and least recently used (LRU) [29]. Prominent examples that stand out in some aspects are 2Q [30], greedy-dual-size-frequency (GDSF) [31], adaptive replacement cache (ARC) [32], quadruply segmented LRU (S4LRU) [33], TinyLFU [34], GD-Wheel [35], AdaptSize [36], RobinHood [37], SIEVE [38], simple scalable FIFO-based algorithm with three static queues (S3-FIFO) [39], cost and recency aware (CRA) [40], and resolver aware remaining time (RART) [16].

Machine learning (ML) and dynamic algorithms have recently promoted workload-adaptable solutions. Notable examples are learning from optimal (LFO) [41], learning cache replacement (LeCaR) [42], learning relaxed belady (LRB) [43], parrot [44], cacheus [45], meta-policy [46], heuristic-augmented learned preferences (HALP) [47], machine learning at the tail (MAT) [48], and follow the prediction (FtP) [49]. Despite that, traffic data analysis applications or ML estimation techniques still evolve to build the network cache [50]–[53]. Generally, they are applied to a specific cache service [43], [54], [55], governed by different rules to cache the data. It brings the advantages of dynamic cache solutions to fetch real traffic. Although ML techniques require high computational resources [50], they could provide good performance results with context-specific solutions [47], [48].

Additionally, for cache admission and eviction, a prefetching mechanism can be applied to have the data before it is requested [56]–[61]. The study [62] uses renewal and preresolving strategies based on user browser data to reduce cache misses. Although this approach dynamically changes the DNS cache content and enhances QoE, it needs users' data from the last access to achieve these results and only works with web browsing traffic. Work in [63] increases the performance and reliability of resolvers cooperatively, but they need a coordinated method and keep the latency of the edge with extended reach connectivity delay.

Thus, assuming the current state of the art, this study aims to answer the following research questions.

(i-a) Which information do we use in the ML model to influence cache decisions with custom metrics? (i-b) How do we obtain and extract that information from DNS resolvers online? (i-c) How much data do we store to balance the cache working improvement and the computational cost related?

(ii-a) Can the DNS data ML approach for managing cache improves the cache-efficient hit ratio? (ii-b) Can it also improve the performance balance between computing resource usage and cache effectiveness?

Designed to address those research questions, Fig. 2 proposes integrating dynamic DNS resolver cache management with machine learning-based decision improvement components, which can influence cache operation to manage entries.

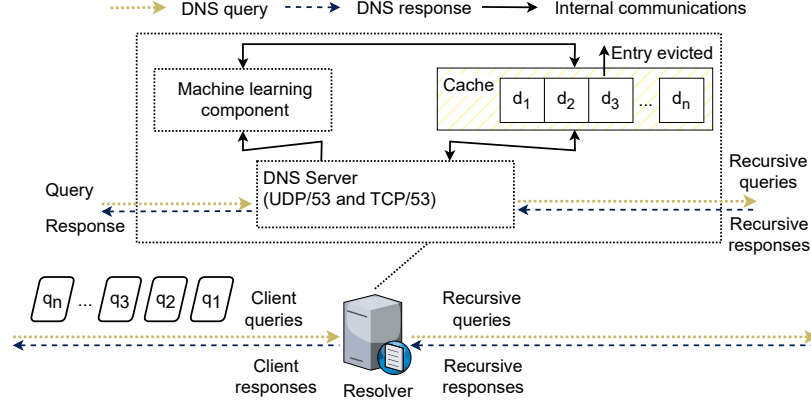


Fig. 2. Extended DNS resolver components architecture. The machine learning component receives information from DNS and interacts with the cache components to improve its working.

III. METHODOLOGY

This study will use state-of-the-art results in the caching area, domain resolution processes, and user-driven machine-learning prediction strategies as the basis for its development. In addition, the studies cited here and their discussion will be detailed during study development. The remaining directions of this proposed study will be divided into five stages detailed below and are depicted in Table I.

TABLE I
ACTIVITY SCHEDULE. 36-MONTH TIME RANGE GUIDED THIS TIMETABLE TO DEVELOP THIS PROPOSED STUDY.

Stage	Semester					
	1st	2nd	3rd	4th	5th	6th
Requirements^a	X				X	X
First	X	X				
Second		X	X			
Third			X	X		
Fourth				X	X	
Fifth					X	X

^aMandatory procedures involved in Ph.D. program.

The first stage focuses on modeling a domain name resolution service architecture to account for user request patterns, DNS working, and cache efficiency, then developing its components and employing them in the decision about DNS cache. The second stage develops ML to identify DNS entry trends concerning the extracted information about potential future accesses. In the third stage, algorithms with custom cache management policies, besides those already well-known in the literature, are implemented using user trends from the ML component. The fourth stage concentrates on integrating ML predictions into DNS resolution servers and creating a simulation model to evaluate it, using self-development and well-known tools from caching-related work. The final stage (fifth) targets a proof-of-concept of the DNS resolution cache system developed in some selected edge network partners, using the best-performed policies from simulated experiments.

IV. CONCLUSION

Although the cache improves the DNS resolution service, its management requires comprehensive care of resources to improve its performance, which can minimize other issues associated with the service. To address this, we introduced a simple architecture proposal to enhance the current resolver service cache by integrating the machine learning component with cache algorithms for better performance and reduced latency bottleneck. However, the subsequent steps involve designing an efficient algorithm based on machine learning for DNS resolver cache management mechanisms. It includes incorporating user traffic trends to fill it with the required data before necessary, which involves evaluation of performance and applicability in real environments and the delimitation of features needed for this decision online in the machine learning model. The results of this study can improve QoE for users by implementing better cache management policies in the DNS cache of global access infrastructures. However, accurately measuring the impact of the solution on user experience is a challenge, and appropriate measurement strategies are crucial to assess the influence of the solution confidently.

REFERENCES

- [1] P. Mockapetris, "Domain names - concepts and facilities," pp. 1–55, Nov. 1987. [Online]. Available: <https://www.rfc-editor.org/info/rfc1034>
- [2] —, "Domain names - implementation and specification," pp. 1–55, Nov. 1987. [Online]. Available: <https://www.rfc-editor.org/info/rfc1035>
- [3] A. Vulimiri, P. B. Godfrey, R. Mittal, J. Sherry, S. Ratnasamy, and S. Shenker, "Low latency via redundancy," *Conex. 2013 - Proc. 2013 ACM Int. Conf. Emerg. Netw. Exp. Technol.*, pp. 283–294, 2013.
- [4] I. N. Bozkurt, A. Aguirre, B. Chandrasekaran, P. B. Godfrey, G. Laughlin, B. Maggs, and A. Singla, "Why is the internet so slow?!" in *Passiv. Act. Meas.*, M. A. Kaafar, S. Uhlig, and J. Amann, Eds. Cham: Springer International Publishing, 2017, pp. 173–187.
- [5] R. de Oliveira Schmidt, J. Heidemann, and J. H. Kuipers, "Anycast Latency: How Many Sites Are Enough?" in *Passiv. Act. Meas.*, M. A. Kaafar, S. Uhlig, and J. Amann, Eds. Cham: Springer International Publishing, 2017, pp. 188–200.
- [6] ETSI, "Enhanced DNS support towards distributed MEC environment," *ETSI White Pap. No. 39*, no. 39, 2020.
- [7] E. Cohen and H. Kaplan, "Prefetching the means for document transfer: A new approach for reducing Web latency," *Proc. - IEEE INFOCOM*, vol. 2, no. 4, pp. 854–863, 2000.
- [8] C. E. Wills and H. Shang, "The contribution of DNS lookup costs to web object retrieval," Citeseer, Tech. Rep., 2000.
- [9] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, "DNS performance and the effectiveness of caching," *IEEE/ACM Trans. Netw.*, vol. 10, no. 5, pp. 589–603, Oct. 2002.
- [10] R. Liston, S. Srinivasan, and E. Zegura, "Diversity in DNS performance measures," in *Proc. 2nd ACM SIGCOMM Work. Internet Meas.*, 2002, p. 19.
- [11] Z. Wang, "Analysis of DNS cache effects on query distribution," *Sci. World J.*, vol. 2013, pp. 1–8, 2013.
- [12] X. Yuchi, X. Lee, and L. Pan, "Dealing with temporary domain name issues in the DNS," *Proc. - IEEE Symp. Comput. Commun.*, vol. 2016-Augus, pp. 778–783, 2016.
- [13] S. Hao and H. Wang, "Exploring domain name-based features on the effectiveness of DNS caching," *Comput. Commun. Rev.*, vol. 47, no. 1, pp. 36–42, 2017.
- [14] G. Yu, Y. Zhang, H. Cui, X. Yang, and Y. Li, "Mitigating Negative Impacts on DNS Caches Caused by Disposable Domain Names," *Proc. - IEEE Symp. Comput. Commun.*, vol. 2019-June, no. July, pp. 1–23, 2019.
- [15] G. C. Moura, J. Heidemann, R. d. O. Schmidt, and W. Hardaker, "Cache me if you can: effects of DNS time-to-live," in *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*. USC/Information Sciences Institute, 2019, pp. 101–115.
- [16] I. Alayoff and G. Einziger, "Optimizing DNS resolvers for high loads," *2023 IFIP Netw. Conf. IFIP Netw. 2023*, 2023.
- [17] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing DNS resolvers in the wild," in *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*. Assoc. for Comput. Machinery, 2010, pp. 15–21.
- [18] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante, "Content delivery and the natural evolution of DNS: Remote DNS trends, performance issues and alternative solutions," in *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, 2012, pp. 523–536.
- [19] H. Hours, E. Biersack, P. Loiseau, A. Finamore, and M. Mellia, "A study of the impact of DNS resolvers on performance using a causal approach," in *Proc. - 2015 27th Int. Teletraffic Congr. ITC 2015*. Institute of Electrical and Electronics Engineers Inc., Sep. 2015, pp. 10–18.
- [20] S. Zhuang, J. H. Wang, P. Zhang, and J. Wang, "Understanding the latency to visit websites in China: An infrastructure perspective," *Comput. Networks*, vol. 169, 2020.
- [21] Google, "Google public DNS: Performance benefits," 2021. [Online]. Available: <https://developers.google.com/speed/public-dns/docs/performance>
- [22] Cloudflare, "How to make the Internet faster for everyone," [Online]. Available: <https://cloudflare.com/learning/performance/more/speed-up-the-web>
- [23] A. J. Smith, "Cache memories," *ACM Comput. Surv.*, vol. 14, no. 3, pp. 473–530, 1982.
- [24] S. Pires, A. Ziviani, and L. N. Sampaio, "Contextual dimensions for cache replacement schemes in informationcentric networks: a systematic review," *PeerJ Comput. Sci.*, vol. 7, pp. 1–51, 2021.
- [25] G. Hasslinger, M. Okhovatzadeh, K. Ntougias, F. Hasslinger, and O. Hohlfeld, "An overview of analysis methods and evaluation results for caching strategies," *Comput. Networks*, vol. 228, no. August 2022, p. 109583, 2023.
- [26] L. A. Belady, "A study of replacement algorithms for a virtual-storage computer," *IBM Syst. J.*, vol. 5, no. 2, pp. 78–101, 1966.
- [27] R. Mattson, J. Gecsei, D. R. Slutz, and I. L. Traiger, "Evaluation techniques for storage hierarchies," *IBM Syst. J.*, vol. 9, no. 2, pp. 78–117, 1970.
- [28] D. S. Berger, N. Beckmann, and M. Harchol-Balter, "Practical bounds on optimal caching with variable object sizes," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 2, no. 2, pp. 1–38, 2018.
- [29] D. D. Sleator and R. E. Tarjan, "Amortized efficiency of list update rules," in *Proc. Annu. ACM Symp. Theory Comput.*, vol. Part F1303, 1984, pp. 488–492.
- [30] T. Johnson and D. Shasha, "2Q: A low overhead high performance management replacement algorithm," *Proc. 20th VLDB Conf.*, pp. 439–450, 1994.
- [31] L. Cherkasova, "Improving WWW proxies performance with Greedy-Dual-Size-Frequency caching policy," *HP Lab. Tech. Rep.*, no. 98 -69, 1998.
- [32] N. Megiddo and D. S. Modha, "Outperforming LRU with an adaptive replacement cache algorithm," *Computer (Long. Beach. Calif.)*, vol. 37, no. 4, pp. 58–65, 2004.
- [33] Q. Huang, K. Birman, R. Van Renesse, W. Lloyd, S. Kumar, and H. C. Li, "An analysis of Facebook photo caching," *SOSP 2013 - Proc. 24th ACM Symp. Oper. Syst. Princ.*, pp. 167–181, 2013.
- [34] G. Einziger and R. Friedman, "TinyLFU: A highly efficient cache admission policy," *Proc. - 2014 22nd Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2014*, pp. 146–153, 2014.
- [35] C. Li and A. L. Cox, "GD-wheel: A cost-aware replacement policy for key-value stores," in *Proc. 10th Eur. Conf. Comput. Syst. EuroSys 2015*, 2015.
- [36] D. S. Berger, R. K. Sitaraman, and M. Harchol-Balter, "AdaptSize: Orchestrating the hot object memory cache in a content delivery network," *Proc. 14th USENIX Symp. Networked Syst. Des. Implementation, NSDI 2017*, pp. 483–498, 2017.
- [37] D. S. Berger, B. Berg, T. Zhu, M. Harchol-Balter, and S. Sen, "Robinhood: Tail latency-aware caching - dynamically reallocating from cache-rich to cache-poor," *Proc. 13th USENIX Symp. Oper. Syst. Des. Implementation, OSDI 2018*, pp. 195–212, 2018.
- [38] Y. Zhang, J. Yang, Y. Yue, Y. Vigfusson, and K. V. Rashmi, "SIEVE is simpler than LRU: An efficient turn-key eviction algorithm for web caches," *Nsdi*, 2023.
- [39] J. Yang, Y. Zhang, Z. Qiu, Y. Yue, and R. Vinayak, "FIFO queues are all you need for cache eviction," in *Proc. 29th Symp. Oper. Syst. Princ.* New York, NY, USA: ACM, Oct. 2023, pp. 130–149.
- [40] G. Einziger, O. Himelbrand, and E. Waisbard, "Boosting cache performance by access time measurements," *ACM Trans. Storage*, vol. 19, no. 1, 2023.
- [41] D. S. Berger, "Towards lightweight and robust machine learning for CDN caching," *HotNets 2018 - Proc. 2018 ACM Work. Hot Top. Networks*, pp. 134–140, 2018.
- [42] G. Vietri, L. V. Rodriguez, W. A. Martinez, S. Lyons, J. Liu, R. Rangaswami, M. Zhao, and G. Narasimhan, "Driving cache replacement with ML-based LeCaR," in *10th USENIX Work. Hot Top. Storage File Syst. (HotStorage 18)*. Boston, MA: USENIX Assoc., July 2018.
- [43] Z. Song, D. S. Berger, K. Li, W. Lloyd, I. Nsdi, Z. Song, D. S. Berger, K. Li, and W. Lloyd, "Learning relaxed belady for content distribution network caching," *Proc. NSDI*, 2020.
- [44] E. Liu, M. Hashemi, K. Swersky, P. Ranganathan, and J. Ahn, "An imitation learning approach for cache replacement," in *Proc. 37th Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 6237–6247.

- [45] L. V. Rodriguez, F. Yusuf, S. Lyons, E. Paz, R. Rangaswami, J. Liu, M. Zhao, and G. Narasimhan, "Learning cache replacement with cacheus," in *19th USENIX Conf. File Storage Technol. (FAST 21)*. USENIX Assoc., Feb. 2021, pp. 341–354.
- [46] S. Pires, A. Ribeiro, and L. Sampaio, "A meta-policy approach for learning suitable caching replacement policies in information-centric networks," *Proc. ACM Symp. Appl. Comput.*, pp. 1950–1959, 2022.
- [47] Z. Song, K. Chen, N. Sarda, D. Altunbükten, E. Brevdo, J. Coleman, X. Ju, P. Jurczyk, R. Schooler, and R. Gummadi, "HALP: Heuristic aided learned preference eviction policy for YouTube content delivery network," *Proc. 20th USENIX Symp. Networked Syst. Des. Implementation, NSDI 2023*, pp. 1149–1163, 2023.
- [48] D. Yang, D. S. Berger, K. Li, and W. Lloyd, "A learned cache eviction framework with minimal overhead," Jan. 2023.
- [49] A. Antoniadis, C. Coester, M. Eliáš, A. Polak, and B. Simon, "Online metric algorithms with untrusted predictions," *ACM Trans. Algorithms*, vol. 19, no. 2, 2023.
- [50] A. Kulkarni and A. Seetharam, "Model and machine learning based caching and routing algorithms for cache-enabled networks," 2020.
- [51] G. Paschos, G. Iosifidis, and G. Caire, "Cache optimization models and algorithms," *Found. Trends Commun. Inf. Theory*, vol. 16, no. 3–4, pp. 156–345, 2020.
- [52] Y. Wang and V. Friderikos, "A survey of deep learning for data caching in edge network," *Informatics*, vol. 7, p. 43, 2020.
- [53] R. Aghazadeh, A. Shahidinejad, and M. Ghobaei-Arani, "Proactive content caching in edge computing environment: A review," *Softw. Pract. Exp.*, vol. n/a, no. August, pp. 1–45, Sep. 2021.
- [54] A. Narayanan, S. Verma, E. Ramadan, P. Babaie, and Z. L. Zhang, "Deepcache: A deep learning based framework for content caching," in *NetAI 2018 - Proc. 2018 Work. Netw. Meets AI ML, Part SIGCOMM 2018*. Assoc. for Comput. Machinery, Inc, Aug. 2018, pp. 48–53.
- [55] H. Chen and G. Tan, "A q-learning-based network content caching method," *Eurasip J. Wirel. Commun. Netw.*, vol. 2018, no. 1, pp. 1–10, 2018.
- [56] J. Griffioen and R. Appleton, "Reducing file system latency using a predictive approach," in *USENIX Summer 1994 Tech. Conf. (USENIX Summer 1994 Tech. Conf.* Boston, MA: USENIX Assoc., June 1994.
- [57] Z. Li, Z. Chen, S. M. Srinivasan, and Y. Zhou, "C-miner: mining block correlations in storage systems," in *Proc. 3rd USENIX Conf. File Storage Technol.*, ser. FAST '04. USA: USENIX Assoc., 2004, pp. 173–186.
- [58] G. Soundararajan, M. Mihailescu, and C. Amza, "Context-aware prefetching at the storage server," in *USENIX 2008 Annu. Tech. Conf.*, ser. ATC'08. USA: USENIX Assoc., 2008, pp. 377–390.
- [59] B. S. Gill and L. A. D. Bathen, "AMP: Adaptive multi-stream prefetching in a shared cache," in *5th USENIX Conf. File Storage Technol. (FAST 07)*. San Jose, CA: USENIX Assoc., Feb. 2007.
- [60] J. Yang, R. Karimi, T. Sæmundsson, A. Wildani, and Y. Vigfusson, "MITHRIL: Mining sporadic associations for cache prefetching," *Proc. 2017 Symp. Cloud Comput.*, vol. 63, no. 2, pp. 66–79, May 2017.
- [61] D. Zhu, H. Du, Y. Sun, X. Li, R. Qu, H. Hu, S. Dong, H. M. Zhou, and N. Cao, "Massive files prefetching model based on LSTM neural network with cache transaction strategy," *Comput. Mater. Contin.*, vol. 63, no. 2, pp. 979–993, 2020.
- [62] E. Cohen and H. Kaplan, "Proactive caching of DNS records: Addressing a performance bottleneck," in *Proc. - 2001 Symp. Appl. Internet, SAINT 2001*, 2001, pp. 85–94.
- [63] K. S. Park, V. S. Pai, L. Peterson, and Z. Wang, "CoDNS: Improving DNS performance and reliability via cooperative lookups," in *Proc. 6th Conf. Symp. Oper. Syst. Des. Implement. - Vol. 6*. USENIX Assoc., 2004, pp. 199–214.