An Analytical Model for the LISP Cache Size

Florin Coras, Jordi Domingo-Pascual, and Albert Cabellos-Aparicio

Universitat Politècnica de Catalunya
Barcelona, Spain
{fcoras, jordi.domingo, acabello}@ac.upc.edu

Abstract. Concerns regarding the scalability of the inter-domain routing have encouraged researchers to start elaborating a more robust Internet architecture. While consensus on the exact form of the solution is yet to be found, the need for a semantic decoupling of a node's *location* and *identity* is generally accepted as the only way forward. One of the most successful proposals to follow this guideline is LISP (Loc/ID Separation Protocol). Design wise, its aim is to insulate the Internet's core routing state from the dynamics of edge networks. However, this requires the introduction of a mapping system, a distributed database, that should provide the binding of the two resulting namespaces. In order to avoid frequent lookups and not to penalize the speed of packet forwarding, map-caches that store temporal bindings are provisioned in routers. In this paper, we rely on the working-set theory to build a model that accurately predicts a map-cache's performance for traffic with time translation invariance of the working-set size. We validate our model empirically using four different packet traces collected in two different campus networks.

Keywords: LISP, Internet architecture, cache modeling, working set

1 Introduction

The fast paced growth [8] of the global inter-domain routing table and infrastructure has recently raised a series of concerns and spurred debate regarding the Internet's architectural inability to scale. Reasons for the growth are partly organic in nature, as new domains are continuously added to the topology, but also related to operational practices that defeat provider-based aggregation of prefixes. In this sense, multihoming, traffic engineering and allocations of provider-independent prefixes are the main drivers behind prefix de-aggregation [16].

In a recent Internet Advisory Board workshop [16] participants deemed the routing table growth as one of the Internet's most important problems and tracked down its origins to the semantic overloading of the IP addresses with both *location* and *identity* information. As a result, a set of architectures inspired by the loc/id split paradigm have been proposed [14] to overcome the limitation. Most notably, LISP [5] aims to alleviate the routing state pressure exerted by edge networks on core networks by means of an *identity-location* separation at network level. It is designed for an incremental deployment and it relies on a pilot network [1] for experimentation.

For the binding of identifiers and locators, LISP introduces a distributed database called the *mapping-system*. As LISP routers are typically domain border routers, the identifier resolution delays through the mapping system (hundreds of ms [11]) are several orders of magnitude higher than packet inter-arrival times (ns). Therefore, to speed up intra-router packet forwarding and to protect the mapping system from floods of resolution requests, the LISP routers are provisioned with mapping caches (*map-caches*) that temporarily store in use mappings. Given the map-cache's primordial role in assuring LISP's data-plane (packet forwarding) and control-plane (mapping-system) scalability, a better understanding of its behavior is crucial.

In this paper we aim to improve the understanding of the map-cache's performance by developing an analytical model that links miss rate with cache size. Specifically, we rely on the working-set theory [3] to build a model that accurately predicts the performance based on simple and measurable parameters of packet traces that possess time translation invariance of the working-set curves. We validate our model empirically using four different packet traces collected in two different campus networks. Our findings show that the miss rate decreases at an accelerated pace with the cache-size, and that even for a relative small size (5% of the Internet's aggregated routing table) the performance is acceptable (below 0.6% miss-rate).

2 LISP Overview

LISP [5] is one [14] of the recently emerged architectural solutions to the Internet's scalability problem [16]. Its main goal is that of splitting the semantics of IP addresses with the aim of forming two namespaces that unambiguously identify core (locators) and edge (identifiers) network objects. To facilitate transition from the current Internet infrastructure, both of the resulting namespaces use the existing IP addressing scheme. As a result, the split does not affect routing within existing stub or transit networks. Nevertheless, as identifiers and locators bear relevance only within their respective namespaces, a form of conversion, from one to the other, has to be performed at border points between core and edge networks. Traditionally, the two techniques that may be employed for a fast translation are address rewriting [17] and map-and-encap [7]. Unfortunately, besides their need for data plane modifications, both also require the introduction of a new control-plane *mapping function* able to provide bindings that link identifiers to locators. Out of the two, LISP enabled border routers make use of the latter. Therefore, prior to forwarding a host generated packet, a LISP router maps the destination address, or Endpoint IDentifier (EID), to a corresponding destination Routing LOCator (RLOC) by means of a LISP specific distributed database, called the *mapping system* [2, 19, 11, 15]. Once a mapping is obtained, the border router tunnels the packet from source edge to corresponding destination edge network by means of an encapsulation with a

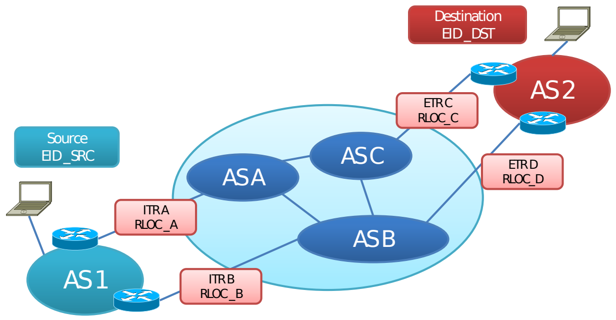


Fig. 1: Example LISP Architecture

LISP-UDP-IP header. The outer IP header addresses are the RLOCs pertaining to the corresponding border routers (see Fig. 1). At the receiving router, the packet is decapsulated and forwarded to its intended destination. In LISP parlance, the source router, that performs the encapsulation, is called an Ingress Tunnel Router (ITR) whereas the one performing the decapsulation is named the Egress Tunnel Router (ETR). One that performs both functions is referred to as an xTR.

Because all packets need to be encapsulated, the packet throughput of an ITR is highly dependent on the time needed to obtain a mapping. In consequence, all ITRs are provisioned with a cache, called map-cache, that stores the recently used EID-prefix-to-RLOC mappings to avoid frequent EID resolutions through the mapping system. Stale entries are avoided with the help of validity periods (timeouts) that mappings carry as attributes. One can easily observe that the map-cache is most efficient in situations when destination EIDs present high temporal and/or spatial locality and that its size depends on the diversity of the visited destinations. As a result, a cache's performance depends entirely on its provisioned size and on traffic characteristics. Given the map-cache's critical role in the LISP architecture, a good understanding of the properties linking its size and performance with the traffic's characteristics is fundamental.

3 LISP Cache Modeling

This section presents the theoretical background and methodology used to model the LISP map-cache's performance. After a brief presentation of the working-set model we perform an analysis of the traffic pertaining to several network traffic traces. The observed traffic properties enable us to analytically link cache size and miss rate.

3.1 Working-Set Theory

As our goal is to determine suitable caching strategies and to better model their performance for LISP routers, we leverage previous resource allocation and cache

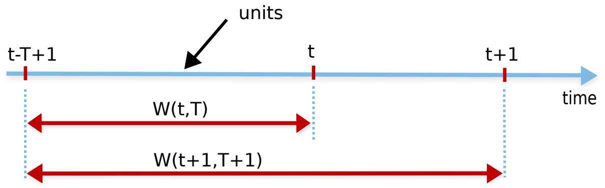


Fig. 2: The Working Set Model

performance research carried out in the field of operating systems. Consequently, in order to unify language, we are forced to draw parallels between concepts pertaining to our two disciplines. We associate to what systems understand by a *program* a network *traffic trace*, and in a similar vein, to a *page* an *address prefix*. Although seemingly unrelated, from an abstract point of view, the properties of the above pairs of objects influence in similar ways a cache's performance. We depict this point further in what follows.

For operating systems, a general resource allocation treatment was possible after it was observed that programs more or less obey the so called *principle of locality*. In such situations, a program's past referenced pages may be used as a good predictor for near future, to be re-referenced pages. We speculate that network traffic traces might roughly obey the same principle mainly because of flow burstiness in time but also due to traffic aggregation. Consequently, we try to evaluate the feasibility of in-router *prefix caching* by first building and then using a working-set model for network traffic.

In what follows, in order to avoid repetitive text, we shall be using the term *unit* of reference as a substitute for both pages and prefixes; and *reference set* to represent the set of all referenced pages or prefixes. Further, given a *reference set* N, we define a *reference string* as the sequence $\rho = r_1 r_2 \ldots r_i \ldots$ where each unit r_i belongs to N.

As explained by Denning in [4], a working-set analysis in this context may be performed only if based on three constraints that provide for a more rigorous definition for *locality of reference*:

1. Reference strings are unending
2. The stochastic mechanism underlying the generation of a reference string is stationary, i.e. independent of absolute time origin.
3. For all $t > 0$, r_t and r_{t+x} become uncorrelated as $x \to \infty$

The first of the constraints, though obviously not fulfillable, introduces an insignificant error because the reference strings generated by practical programs or traces are long from a statistical standpoint. The third of the assumptions requires that the references become uncorrelated as the distance (in units) goes to infinity. This can usually be asserted as being true in practice. The most restrictive of the three is the second assumption which limits the analysis to a locality where all three constraints, including stationarity are satisfied.

If all of the above requirements are satisfied, and t is a measure of time in *units*, then we can formally define the working-set $W(t,T)$, as: the set of distinct *units* that have been referenced among the T most recent references, or in the time interval $[t-T+1,T]$ (see Fig. 2). In accordance with [4] we shall refer to T as the *window size*. Also, we will be using the term *working-set curve* when referring to $W(t,T)$ as a function of T, when t is constant.

We exploit the above definition to present some of the working-set's properties of further use in our analysis. For a broader scope discussion of the subject and complete proofs, the interested reader is referred to [4]. The first of the properties we are interested in is the *average working-set size*, *s(T)*. It measures the growth of the working-set with respect to the size of the window T, extending in the past, but independent of absolute time *t*. If *w(t,T)* measures the number of units in *W(t,T)* then, considering the first locality property, one can compute *s(T)* as the averaged working-set size over an infinite number of windows:

$$s\left(T\right)=\lim_{k\to\infty}\frac{1}{k}\sum_{t=1}^{k}w(t,T) \tag{1}$$

It is also shown in [4] that the the *miss rate, m(T)*, which measures the number of units of reference per unit time returning to the working-set, is the derivative of the previous function:

$$m(T)=s(T+1)-s(T) \tag{2}$$

In other words, if both $s(T)$ and $m(T)$ are viewed as functions of T, the miss rate can be regarded as the slope of the average working-set size. Finally, the sign inverted slope of the miss rate function, the second slope of $s(T)$, represents the *average interreference density* function.

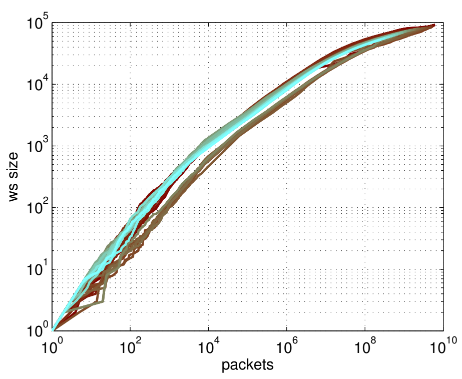
$$-f(T+1)=m(T+1)-m(T) \tag{3}$$

3.2 Traffic Properties

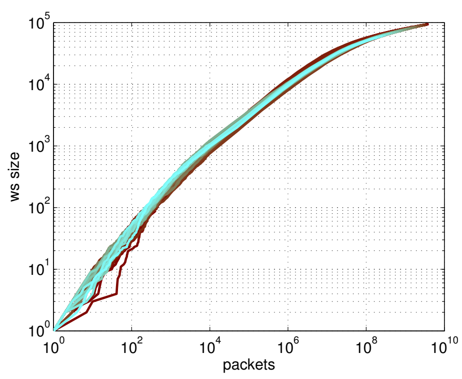
In order to ascertain the practical feasibility of using the working-set model to evaluate a LISP cache's performance we applied the theory presented in Section 3.1 to several network traffic traces. Details regarding the network captures can be found in Section 4.1.

Foremost, we were interested in discovering if our traces satisfy the three constraints introduced in Section 3.1. As previously explained, the first and third introduce negligible errors and thus present no practical limitations. In order to verify the second, and therefore determine the trace segments generated by distinct stationary processes, we devised a simple experiment. For each trace we computed multiple empirical destination prefix working-set curves with start times spanning one day and spaced by half an hour. Intuitively, we were expecting to discover that curves with close start times have a similar growth shape (cluster), whereas those separated by larger time gaps behave differently.

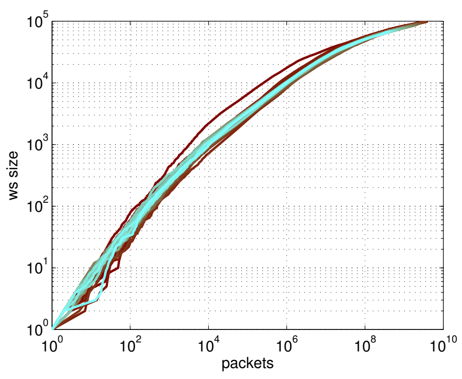
By necessity [4], if working-set curves are generated by the same stationary process then, they will tend to cluster. For a certain window size the working-set



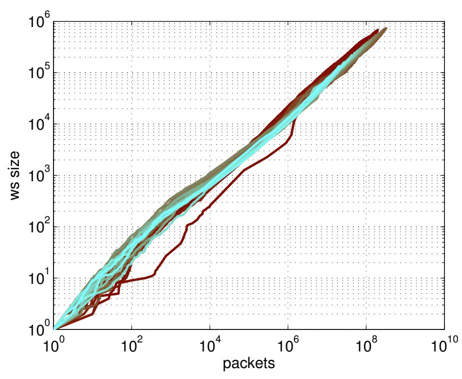
(a) *upc*_09



(b) *upc*_11*a*



(c) *upc*_11*b*



(d) *auck*_08

Fig. 3: Empirical working-set curves. Closeness of color nuance reflects closeness of start time.

size should be normally distributed. In other words, one may use the clusters of the working-set curves to discriminate between trace segments with different underlying generating processes. Figure 3 presents the results obtained in log-log plots. Surprisingly, all traces exhibit a strong clustering and sublinear growth, due to locality, of the working-set curves. We also tested and confirmed (not shown here) that the interreference interval and the relative frequency of reference distributions for our traces are approximately time translation invariant. Only *upc_09* presents a subtle time-of-day behavior with two perceptible clusters while *upc*_11*b* and *auck*_08 each have one discordant curve that seems to be due to localized scanning attacks. Nevertheless, the results suggest that one could approximate the traces as being generated by a stationary process and as an implication, that any conclusions drawn after analyzing the average working set size function would characterize the whole trace. Similarly, Kim et al. observed in [12] that the working-set size for prefixes tends to be highly stable with time for traffic pertaining to a large ISP. It's worth noting that we do not assume that all Internet traffic possesses this approximate time translation invariance of the working-set curves. And in fact, we require that the model we develop further be applied only to traces that have it.

3.3 Analytical Model

Considering that the working-set, $W(t, T)$, always contains the $w(t, T)$ most recently referenced units then, the time translation invariance of $w(t, T)$ for a constant T implies that the working-set actually models a cache of fixed size. Similarly, a cache of size c that uses the Least Recently Used (LRU) eviction policy will always store the c most recently referenced units. Therefore, if $s(T) = c$, the working-set simulates a LRU eviction policy. In other words, due to the time translation invariance of the working-set curves the working-set can be seen as modeling a LRU cache.

With hindsight, one can recognize that the empirical working-set curves from Fig. 3 are piecewise linear when depicted in log-log scale. The observation enabled us to compute for each trace the average working-set size function by means of a piecewise linear fit of the log-log scale plot. As a result, we obtained estimates of both the slope α and the y-intercept β for all of a working-set's segments. We limited the number of knots to just three, however if better fits are desired more knots can be used. Figure 4 presents the goodness of fit for two of our traces. Next, through conversion to linear scale the equations become piecewise power law of the type:

$$s_{fit}(u) = \exp(\beta(u))u^{\alpha(u)} \tag{4}$$

Where u represents the number of referenced destination prefixes, $s_{fit}(u)$ is the fitted working-set size function and $\alpha(u)$ and $\beta(u)$ are piecewise constant functions obtained through fitting. With the help of (2) one can estimate the miss rate for a trace by computing the derivative of s_{fit} like:

$$m_{est}(u) = \exp(\beta(u))\,\alpha(u)\,u^{\alpha(u)-1} \tag{5}$$

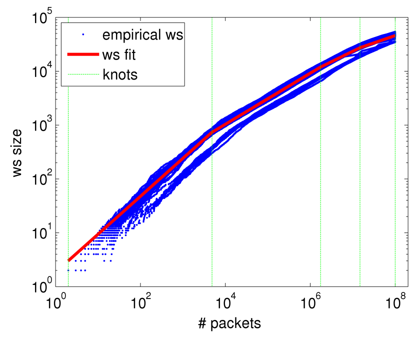
Combing the last two equations one obtains an analytical relation that links the cache size and the estimated miss rate.

$$m_{est}(c) = \exp\left(\frac{\beta^*(c)}{\alpha^*(c)}\right)\,\alpha^*(c)\;c^{1-\frac{1}{\alpha^*(c)}} \tag{6}$$

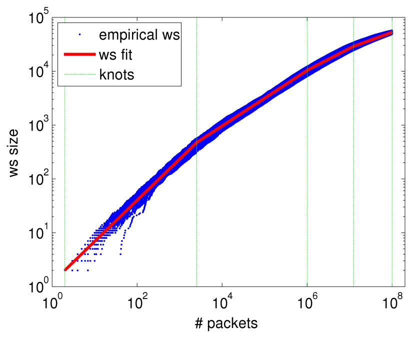
Where, c represents the cache size in number of entries and $\alpha^*(c)$ and $\beta^*(c)$ are piecewise constant functions with knees dependent on c. We put to test the accuracy of the above equation in Section 4.3.

4 Evaluation

In this section, we present the four data sets used to build and evaluate the map-cache model. We then describe the simulator employed in the empirical



(a) *upc*_09



(b) *upc*_11*a*

Fig. 4: Fit of empirical working-set curves

evaluation of the cache and we finish by comparing the theoretical and empirical cache performance results.

4.1 Traces and IP Aggregation

Four one-day packet traces were used for the purpose of our experiments. Three of them have been captured at the 2Gbps link connecting several of our University's campus networks to the Catalan Research Network (CESCA) and consist only of egress traffic. UPC Campus has more than 36k users. One dates back to May 2009 while the other two are from October 2011. The fourth trace belongs to the University of Auckland and has been downloaded from a public trace repository [20]. Its capture date is March 2008 and it consists of University egressing traffic. Further, the addresses have been anonymized and one hour's worth of midday traffic, from 12 to 13, is missing. We shall further refer to these four traces as *upc*_09, *upc*_11*a*, *upc*_11*b* and *auck*_08 respectively.

Both for the evaluation of the working-set in Section 3.3 and for the cache performance evaluation to be presented in Section 4.3, IP addresses had to be mapped to their corresponding prefixes. We considered EID-prefixes to be of BGP-prefix granularity. In the case of UPC traces, we used BGP routing tables downloaded from the RouteViews archive [18] that matched the exact capture date of the traffic traces. The only preprocessing we performed was to filter out more specific prefixes. Generally, such prefixes are used for traffic engineering purposes however, the LISP protocol provides mechanisms for a more efficient management of these functions that do not require EID-prefix de-aggregation. In the case of *auck*_08, the anonymized traffic is prefix preserving but the lack of an anonymized routing table made it impossible for us to determine the exact clustering of anonymized IPs into prefixes. Consequently, we constructed a virtual routing table made up of all prefixes with a 24 bit netmask. We chose the 24 bit limit as it is the longest netmask (corresponding to the most specific prefix) currently accepted for BGP announcements in the Internet's DFZ.

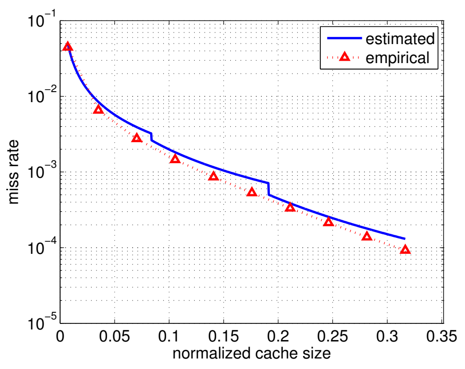
4.2 Simulator

We indirectly assess the effectiveness of the working-set model as a tool for LISP cache performance evaluation by practically testing the soundness of equation (6). For this we implemented a packet trace based simulator that mimics a subset of a LISP ITR's functionality. In short, for each processed packet, the simulator maps the destination IP address to a prefix in accordance with the methodology presented in Section 4.1. If this prefix is already stored in the ITR's cache, no further processing is done and the simulation continues with the next packet. Should the prefix not yet be stored in the cache, two possibilities arise. First, if the cache is not full, the destination prefix is stored in and the processing proceeds to the next packet. Second, if the cache is full, an entry is evicted, the new prefix is stored in and then the simulator moves to the next packet. The entry to be evicted is chosen according to the LRU eviction policy. We use LRU because, as mentioned in Section 3.3, its performance should be appropriately described by equation (6) for our traffic.

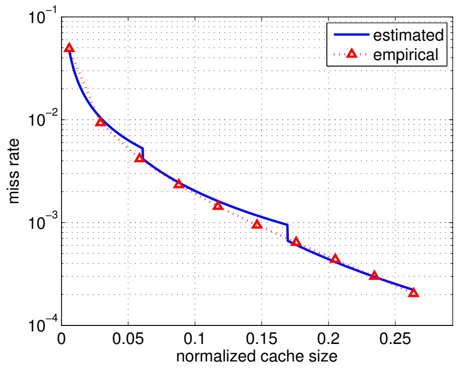
4.3 Results

We ran simulations with several cache sizes for all traces. Figure 5 presents the results for just two, *upc*_09 and *upc*_11*a*, of our traffic captures as they are representative for all four. Cache size is normalized with routing table size. It can be noticed that the absolute error of the estimation is negligible when compared to simulation results. Even in the case of *upc*_09, the only trace to present a more pronounced time-of-day behavior, the absolute error is of small proportions. Further, the model appropriately predicts that even for small LISP cache sizes the performance still stays acceptable, fact also observed by [9, 13, 11]. This is even more remarkable as we do not consider TTL, a timeout after which stale entries are removed, in our simulation.

Because of its proven good performance, we can now recommend the use of the LRU eviction policy in LISP caches, at least for traces that present an approximate time translation invariance of the working-set curves. Still, the diversity of our data sets and previous results from Kim [12] suggest that this property is not due to an isolated event. In this situation, equation (6) may be used to dimension the cache size according to the desired miss rate. Further, the prediction of its mathematical expression, due to discontinuity at the knots, is that miss rate decreases at an accelerated pace with cache size and finally settles to a power law decrease. However, the speed of the decrease depends on the degree of locality present in the trace. This leads us to conclude that cache sizes need not be very large for obtaining good performance. For instance, in the case of *upc*_11*a*, for a cache having 8.4k entries (0.05% of the Internet's aggregated DFZ routing table) the miss rate is around 0.6%.



(a) *upc_09*



(b) *upc_11a*

Fig. 5: Comparison of empirical and estimated miss rates

5 Related Work

Feldmeier [6] and Jain [10] were among the first to evaluate the possibility of performing destination address caching by exploiting the locality of the reference strings in network environments. Feldmeier analyzed traffic on a gateway router and showed that caching of both flat and prefix addresses can significantly reduce routing table lookup times. Jain however performed his analysis in an Ethernet environment. He was forced to concede that despite the locality observed in the traffic, small cache performance was struggling due to traffic generated by protocols with deterministic behavior. Both works were fundamental to the field however their results were empirical and date back to the beginning of the 1990s, years when the Internet was still in its infancy. Recently, Kim et al. [12] showed the feasibility of route caching after performing a measurement study within the operational confinement of an ISP's network. They show by means of an experimental evaluation that LRU performs close to an optimal eviction policy and that working-set size is generally stable with time. We also observe the stability of the working-set for our data sets but we further leverage it to build a LRU model instead of just empirically evaluating its performance.

Iannone et al. [9] performed an initial trace driven study of the LISP map-cache performance. Instead of limiting the cache size by using an eviction policy, their cache implementation evicted stale entries after a configurable timeout value. Further, Kim et al. [13] have both extended and confirmed the previous results with the help of a larger, ISP trace and by considering LISP security aspects in their evaluation. Ignoring security concerns, which we did not consider, and despite differences regarding the cache eviction policy, the results of these last two works seem to be in agreement with ours. Zhang et al. [21] performed a trace based Loc/ID mapping cache performance analysis assuming a LRU eviction policy. They used two 24-hour traffic traces captured at two egressing links of the China Education and Research Network backbone network. They concluded that a small cache can offer good results. Finally, using trace *upc_09*, Jakab et al. [11] analyzed the performance of several LISP mapping systems and, without focusing on a cache analysis, also observed very low miss rates for a cache model similar to that used in [9]. Our work confirms previous LISP cache analysis results however, it also tries to provide a better understanding of the reasons behind the relatively good performance of the LISP cache. In this sense it introduces a cache model that could be used to theoretically evaluate or dimension for operational needs the caching performance.

6 Conclusions

The location/identity split paradigm has been recently recommended as the solution to Internet's scalability problems. Out of the proposals to follow its guidelines, LISP seems to be the one to have drawn the largest amounts of attention and support. In the set of newly introduced architectural components, the map-cache is one of the most important to LISP's own scalability and also the subject of our analysis. Given that caches have their performance driven by the characteristics of the traffic they serve, we proposed an evaluation that links the LISP map-cache performance with measurable, intrinsic properties of the traffic processed by a LISP router. In this sense, we advanced the use of the working-set model [3] as a tool to capture the mentioned traffic properties but also as a performance predictor. Accordingly, we also noted that the time-translation invariance of the working-set curves is a constraint that traffic traces need to observe for the theory to be applicable. After we established that our data sets are amenable to working-set modeling we deduced an analytical link between the cache size and miss rate. Finally, we validated our theoretical result by means of simulation and with the help of traffic traces collected in two different campus networks.

We note that the predictions of our model fit with previous observations however, we also remark the versatility of our result. Besides the possibility of using it as a cache dimensioning tool in operational environments it may also be used in more complex models that evaluate the scalability of LISP or other loc/id architectures. Another interesting application could be to evaluate the effects of trashing attacks on the LISP map-cache.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Education under scholarship number AP2009-3790, research project TEC2011-29700-C02, Catalan Government under project 2009SGR-1140 and a Cisco Unsolicited Research Proposal Grant.

References

1. LISP Testbed, http://www.lisp4.net/
2. D. Farinacci, V. Fuller, D. Meyer, and D. Lewis: LISP Alternative Topology (LISP+ALT). draft-ietf-lisp-alt-10 (Dec 2011), work in progress
3. Denning, P.J.: The working set model for program behavior. Commun. ACM 11(5), 323–333 (1968)
4. Denning, P.J., Schwartz, S.C.: Properties of the working-set model. Commun. ACM 15(3), 191–198 (1972)
5. Farinacci, D., Fuller, V., Meyer, D., Lewis, D.: Locator/ID Separation Protocol (LISP). draft-ietf-lisp-17 (Dec 2011), work in progress
6. Feldmeier, D.: Improving gateway performance with a routing-table cache. In: INFOCOM'88. Networks: Evolution or Revolution, Proceedings. Seventh Annual Joint Conference of the IEEE Computer and Communcations Societies, IEEE. pp. 298–307. IEEE (1988)
7. Hinden, R.: New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG. RFC 1955 (Informational) (Jun 1996)
8. Huston, G.: BGP Report, http://bgp.potaroo.net/
9. Iannone, L., Bonaventure, O.: On the Cost of Caching Locator/ID Mappings. In: Proceedings of the 3rd International Conference on emerging Networking EXperiments and Technologies (CoNEXT'07). pp. 1–12. ACM (Dec 2007)
10. Jain, R.: Characteristics of destination address locality in computer networks: A comparison of caching schemes. Computer networks and ISDN systems 18(4), 243–254 (1990)
11. Jakab, L., Cabellos-Aparicio, A., Coras, F., Saucez, D., Bonaventure, O.: LISP-TREE: A DNS Hierarchy to Support the LISP Mapping System. Selected Areas in Communications, IEEE Journal on 28(8), 1332 –1343 (october 2010)
12. Kim, C., Caesar, M., Gerber, A., Rexford, J.: Revisiting Route Caching: The World Should Be Flat. In: Proceedings of the 10th International Conference on Passive and Active Network Measurement. pp. 3–12. PAM '09 (2009)
13. Kim, J., Iannone, L., Feldmann, A.: A deep dive into the LISP cache and what ISPs should know about it. In: Proceedings of the 10th international IFIP TC 6 conference on Networking - Volume Part I. pp. 367–378. NETWORKING'11 (2011)
14. Li, T.: Recommendation for a Routing Architecture. RFC 6115 (Informational) (Feb 2011)
15. Menth, M., Hartmann, M., Hofling, M.: FIRMS: a future Internet mapping system. Selected Areas in Communications, IEEE Journal on 28(8), 1326–1331 (2010)
16. Meyer, D., Zhang, L., Fall, K.: Report from the IAB Workshop on Routing and Addressing. RFC 4984 (Informational) (Sep 2007)
17. O'Dell, M.: GSE - An Alternate Addressing Architecture for IPv6. draft-ietf-ipngwg-gseaddr-00.txt (1997)
18. University of Oregon: RouteViews Project, http://www.routeviews.org
19. V. Fuller, D. Farinacci, and D. Lewis: LISP Delegated Database Tree (LISP-DDT). draft-ietf-lisp-ddt-00 (Nov 2011), work in progress
20. WITS: Waikato Internet Traffic Storage: http://www.wand.net.nz/wits/catalogue.php
21. Zhang, H., Chen, M., Zhu, Y.: Evaluating the performance on ID/Loc mapping. In: Global Telecommunications Conference (GLOBECOM 2008). pp. 1–5 (2008)