A fuzzy inference system design for ICP protocol optimization in cache appliances hierarchies

Oscar Linares
Engineering Faculty, Universidad Distrital "Francisco José de Caldas"
Bogotá D.C., Cundinamarca, Colombia

and

Camilo Orozco
Engineering Faculty, Universidad Distrital "Francisco José de Caldas"
Bogotá D.C., Cundinamarca, Colombia

ABSTRACT

A cache appliance is a network terminal which provides cache memory functions, such as object queries service from a user; such objects could be stored in one cache or in a cache hierarchy, trying to avoid carry out service from an origin server. This cache appliance structure improves network performance and quality of service. These appliances use ICP protocol (Internet Cache Protocol) to support interoperation between existing cache hierarchies and web servers, through implementation of a message format to communicate web caches. One cache sends an ICP query to its neighbors. The neighbors send back ICP replies indicating "HIT" or "MISS". When one cache faces an excessive traffic situation, that is, a very high number of service queries from users, ICP protocol may allocate the service to cache which has the desired object. Because of traffic conditions, specific appliance may congests and the requests may be refused, which can decrease network's quality of service. So, a system designed for optimizing cache allocation, considering factors as traffic and priority could be useful. This paper presents a fuzzy inference system design, which uses entries such as number of queries over a time interval and traffic tendency, and as output, the web cache allocation decision that will provides the service; this design is proposed to optimize allocation of caches into a hierarchy for network services performance, so balancing out requesting among hierarchy members and improving services performance.

Keywords
ICP, Web cache, fuzzy inference, Web proxy, hierarchy, traffic control.

1. INTRODUCTION

Internet Cache Protocol (ICP) was designed by D. Wessels y K. Claffy, and published on 1997[1]. This protocol is used for WWW distributed caching systems management and it allows to create relations between caches, like siblings and parents for constitution of a searching hierarchy.

Experience has showed that beyond distributing load far of server's access hot areas, caching documents also can save band weight, reduce latency and protect networks from clients lead to generate cycles and wrong requests several times [1].

Proposed works have included hardware and software solutions. In their evaluation of caching applications performance [2], Lindermann and Waldhorst indicated that hardware commercial solutions include, among others, Cisco's CacheEngine, DynaCache's InforLibria, and Intel's Net Cache Network Appliance. These products have differences regarding to cache size, disk storage and performance. In spite of being different on term of device, all solutions use the same disk replacement scheme, Least Recently Used (LRU) [2]. However, there are other schemes like Segmented Least Recently Used (SLRU), or an algorithm frequency-based called Least Frequently Used with Dynamic Aging (LFU-DA). These algorithms just pretend to allocate or delete cached documents depending on how recent it is or how frequently has been used.

Other software solutions include Inktomi, Novell and Squid. The last is a free source available for academic institutions and may be configured to incorporate adaptations like replacement algorithms – above mentioned – [2, 3]. An approach proposed by Cheung & Kwok [4] uses a fuzzy logic-based load balancing service; however, it's not oriented to caching systems; in fact, it emphasizes on object distributed computing systems.

In terms of protocol, there also exists Cache Array Routing Protocol (CARP) [5], a design based on a membership list of known proxies and a hash function for URL space distribution among those proxies. CARP defines a Membership Table for Proxy Array using an ASCII plain text file. Hash function together with routing algorithm takes the member proxies described on the table and determines which list member should be the appropriate receptor for a cached version of a URL-based key document. Subsequent requests of the document can access the cached object sending an HTTP packet requesting that resource to the adequate member of Proxy Array.

The same year, Cache Digest's specification was published – Version 5 [6]. Digest is a data format and data exchange protocol with implementation capabilities on Squid server Version 2. This protocol supports peer tasks between caching servers with no request – answer exchange. In contrast, servers share an abstract of server contents (the Digest).

In 1999, Cooper issued the extensions draft for Inter Cache Co-operation Protocol [7]; these extensions tried to fix a few number of inter-operation limiting issues of caching proxies. They include additional information for debugging commentaries field on HTTP headers, trying to get objects' route, deleting messages inside a cache and ICP response messages optimization.

Later, experimental work was developed, called Hypertext Caching Protocol (HTCP) [8]. In particular, this protocol expands the domain of cache management to include monitoring remote cache's additions and deletions, requesting immediate deletions, and sending hits about web objects such as the third party locations of cacheable objects.

In spite of ICP is an efficient protocol, it generates a high number of messages, without criteria which allow it to send messages in a more appropriated way. In this paper, we present a fuzzy control system, pretending to optimize the allocation of the cache that will render the service, and message sending, considering factors as occupation rate and traffic.

The fuzzy control system is used as tool for controlling the communication system, trying to improve ICP packets transmission process. This type of system allows allocating resources more adequately than other types of approaches.

This paper is organized as follows. In Section 2 we describe the communication process using ICP. In Section 3, we show the issues regarding to ICP messages. In Section 4, we present our fuzzy control system design and conceptual considerations about its development. In Section 5, we discuss implementation, contributions and preliminary results about the fuzzy control system. Finally, in Section 6, we summarize our findings and present possible future work lines.

2. INTERNET CACHE PROTOCOL

As we indicated in Section 1, ICP is used for WWW distributed caching systems management [1] and it allows to create relationships between caches, like siblings and parents for constitution of a searching hierarchy.

With ICP is possible create two types of relationship between caches: parents and siblings [9]. Parents are located in the highest level of the caches' scheme and they send ICP requests to the siblings to search a possible cached document and siblings answer with hit or miss messages. In case of miss in the whole hierarchy or the maximum waiting time has been exceeded, the miss is forwarded to client by origin server and a copy is sent to hierarchy for future accesses. Siblings are located in the same hierarchy level.

This protocol is implemented over UDP [9]. In addition to its use as object location protocol, ICP messages may be used for cache selection. ICP answer may include information possibly to help in the selection process of the more appropriated object source from which an object may be retrieved.

A caching device is a network terminal that provides cache memory functions, as object requesting; such objects may be stored in a cache or in a hierarchical scheme of caches, avoiding the server to render the service by itself. These structures improve network performance and quality of service.

A server with caching devices stores WWW objects when clients requests it for the first time and returns the stored objects when subsequent requests arrive without retrieving it from origin server [10].

These devices use ICP for supporting caches and web servers' interoperation, through a message format implementation to communicate caches. A cache sends an ICP request to its neighbors; the neighbors returns ICP answer messages indicating hit or miss. In this type of systems, a cache hit may be local or remote [10]. Wherever the type, a large quantity of messages over small IP packets is sent when a HTTP request does not find the requested object. Several studies have showed that remote hit rate is quite lesser than local hit rate [10, 11].

There are two caching schemes categorized by caches organization: hierarchical and distributed [9]. In hierarchical scheme, client caches are located in the lowest level of the hierarchy. This is a geographical location-based scheme, considering importance of client access in the hierarchy. In this type of caching, if document is not found, origin server sends a copy to every cache down in the hierarchy.

In distributed caching scheme, intermediate caches don't need to be configured; every single cache is in the same network level, and simultaneously in the network border gateway, cooperating to serve misses. Searching hash algorithms may be used and with this, it's not necessary to store copies of documents in the caches.

Analyzing architectures [12], regarding to latency, hierarchical caching has lower connection times than distributed caching. The last one has lower transmission times than hierarchical caching.

Hierarchical scheme uses lower bandwidth than distributed caching, improves access latency to web services and to possible documents, and the caches distribution in several geographical sites. A hierarchical caching scheme, using support for intermediate caches, it is quite efficient in terms of bandwidth than distributed scheme which only has caches in the border gateway. However, distributed caching "distributes" traffic better, using more bandwidth in lower network levels.

Disk requirements for distributed caching are much lesser than in hierarchical caching. An institutional cache only need some Gigabytes for store all accessed documents, while one cache in the highest level of a hierarchy requires hundreds of Gigabytes to satisfy required capacity. Distributed caching shares total system load very well and doesn't yield hot areas with high load, but hierarchical caching does.

3. ISSUES ABOUT ICP USE

A large ICP packets number increases routing load, especially between LAN and WAN, and network's border gateway, where all ICP packets are forwarded. Router's load depends not only of total traffic, but also of total number of processed packets. Consequently, a lot of ICP packets generates overload in the network's border gateway.

Because of ICP uses UDP and its design is simple, it's considered an efficient protocol [9]. However, even though its message size is small, it is fairly expensive to provide any improvement in terms of WWW traffic reduction, and ICP

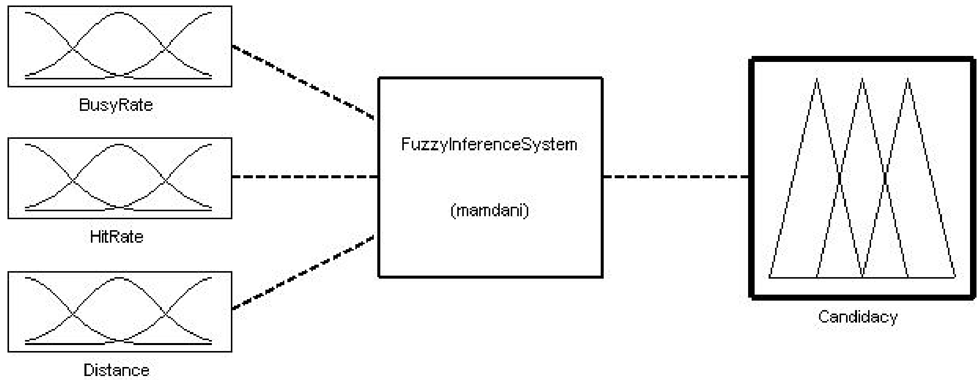

Figure 1. Fuzzy inference system.

messages may be considered network resources waste on Internet [10].

4. FUZZY CONTROL SYSTEM DESIGN

In this section, we describe the design of our fuzzy control system proposed. In first place, we underline several conceptual considerations which warrant the system development. Below, we describe the system architecture, that has four components: variables, rules base, membership functions, and inference engine. Input variables are measurements of real system; these allow to know its behavior in a point time. Output variables are measurements resulting of fuzzy inference process; these can be used to control the system. Rules base is a IF-THEN-form rules set; these rules become numeric inputs in semantic values (via logical operations) to obtain a consequent also semantic, trying to model the real system behavior to lead it to a desired behavior. Membership functions allow match numeric input with a membership extent to defined true values. Inference engine applies fuzzy logic methods (fuzzification and defuzzification) to calculate a fuzzy values for specific inputs, based on rules set and yields outputs also fuzzy, according with rules base consequents, defuzzifies [4] this value and use it to control the system and lead its behavior to a desired state. Figure 1 shows the design of whole scenario.

Conceptual considerations

To manage complex systems, like computer networks, it is necessary to consider several parameters with high uncertainty extent. For modeling this type of systems there are several complex and non-linear relationships between variables, making a conventional control theory-based approach intractable.

To overcome this problem, fuzzy logic control theory may be applied [13], replacing conventional theories. Fuzzy logic-based control pretends to capture intuition in form of IF-THEN rules, and conclusions may be obtained from these rules. Based on intuitive and expert knowledge, system parameters can be modeled as linguistic variables and their corresponding membership functions can be designed. Hence, non-linear system with high level of complexity and uncertainty may be effectively controlled based on fuzzy rules without deal with complex models [13].

Variables

In our design, we consider three input variables: distance, busy rate and hit rate.

Busy rate (B_r). It measures quantity of clients a cache is attending in a point time over its maximum attendance capacity. The last value depends on technical characteristics of device. HTTP and ICP packets are sent during the attending process in full-duplex communication between cache, device which has fuzzy controller (DFC) and/or client. Knowing cache's workload supports decision of distribute or not ICP packets trying to, a certain degree, balancing out load in the caching scheme [see Eq. (1)].

$$\mathbf{B_r} = \frac{\textbf{Quantity of clients in attention (by time unit)}}{\textbf{Maximum attendance capacity (by time unit)}} \quad (1)$$

Hit rate (H_r). It's the ratio between ICP hit number provided by a cache and the total number of ICP requests sent to it in a time interval. This variable provides knowledge based on historical data; this information allows to establish, indirectly, a possibility degree that cache has the document; there are other measures for indicate other probability degrees of contain the document [14, 15] [see Eq. (2)].

$$\mathbf{H_r} = \frac{\textbf{ICP hit number in the cache}}{\textbf{Total number of ICP requests sent to the cache}} \quad (2)$$

Distance (D): It's the Round-Trip Time (RTT); this variable measures amount of time it takes a packet to go from DFC to a cache and for the acknowledgment to be returned. Taking into account ICP is a protocol that retrieves an object and send it to client from first cache that responds with hit message [1], a local but distant cache may take more time responding; hence, while it possibly has the document, it will has no chance to serve it because, probably, the object has already been served by server or a cache nearer to the (DFC) or that responded in lesser time.

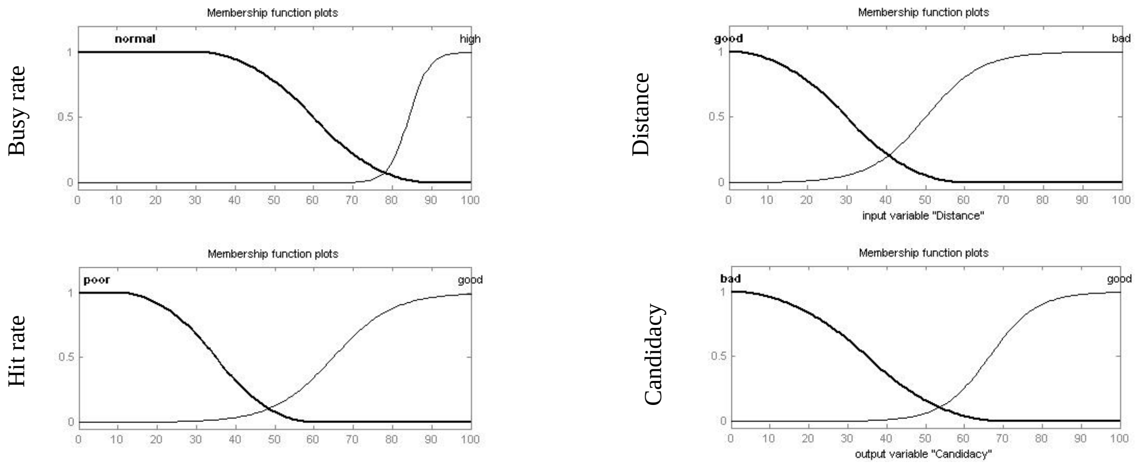

Figure 2. Fuzzy sets of busy rate, hit rate, distance and candidacy.

Candidacy (C). Once fuzzification process has been carried out, there is a consequent called *candidacy*. This indicates how viable is send a request to a cache, taking into account three variables above; the higher its value, the more appropriated to send the request to that specific cache.

Rules base

Based on previous considerations, we elaborate a series of rules to control system's behavior. This rules base is presented in Table 1.

Table 1. Fuzzy control system's rules base.

If busy rate of cache X is	∧	Hit rate of cache X is	∧	Distance to cache X is	⇒	X Cache's cand. is
High	∧	Good	∧	Near	⇒	Poor
High	∧	Good	∧	Distant	⇒	Poor
High	∧	Poor	∧	Near	⇒	Poor
High	∧	Poor	∧	Distant	⇒	Poor
Normal	∧	Good	∧	Near	⇒	Good
Normal	∧	Good	∧	Distant	⇒	Good
Normal	∧	Poor	∧	Near	⇒	Good
Normal	∧	Poor	∧	Distant	⇒	Poor

Membership functions

We defined two membership functions for each fuzzy set. These were represented using sigmoidal and z-moidal sets, to achieve high degree of quality in decision-making. In case of busy rate, we defined two membership categories, normal and high. Normal rate has the form of z-moidal set; this category allows DFC sends requests to that cache and high rate indicates the cache's attending capacity reached a busy threshold; it means DFS can't send more requests to that cache; the last category is also in form of sigmoidal set (see Figure 2).

For hit rate, we defined two membership categories, good and poor. Poor rate indicates a bad possibility to find the document in that cache; this rate is defined in z-moidal set form. Good rate is defined using a sigmoidal set and indicates a good possibility to find the document in that cache. The same type of sets was defined for distance (good, RTT lesser than the server's; bad, more than the server's) and for candidacy (good and bad).

Fuzzy Control System Operation

Cache's manufacturer guides provides data sheets of devices characteristics, including maximum attendance capacity in time units. Using Multi Router Traffic Grapher (MRTG) monitoring tool amount of clients in attention can be measured in the time unit given. With these two values busy rate is obtained.

Next, we calculate hit rate. This value also can be obtained from MRTG. If device has not this software, a meter script may calculate requests have been sent and hit rate among them. Distance is calculated sending RTT messages from DFC to every cache. This RTT is unique for each of them.

These three values are the input variables to fuzzy control system; for carry out fuzzification process, each variable enters in its respective fuzzy set and is matched with a membership value; this process shoots rules (inference) for yield a consequent (i.e., candidacy value). The last value undergoes defuzzification process using Center of Mass method, to yield a definitive clear value. This process is executed for each one cache.

Finally, clear values obtained are organized from higher to lesser in a list and document request packets are sent to caches with highest candidacy in a range defined by expert based on his/her knowledge of network design. For testing, we chosen ***n/2*** range, where ***n*** is total number of caches in hierarchy.

5. DISCUSSION

The fuzzy control system is useful because its implementation pretends to diminish quantity of packets forwarded in a normal ICP transaction in the document search phase; taking into account this excess of packets is considered a waste in the Internet, reducing packets may imply savings in several parameters of computational and network resources use and the access to them.

In this moment, we are testing the system using simulation processes. Nevertheless, our preliminary analysis indicates the following considerations: in normal communication process with ICP, regardless of hit or miss, ***2n*** ICP packets are sent, where ***n*** is the number of caches [10]; if we choose ***n/2*** of caches as requests message sending range, after fuzzy inference process, amount of packets may be reduced to ***n*** in whole communication process. Fuzzy inference system allows categorize caches by possibility of having a document, improving its searching process, in such a way that global hit rate may be increased. In spite of there are other approaches to measure such possibility [14, 15], its calculation may aggregate processing time in the moment of make a decision, making system inefficient in providing an timely decision. Mentioned approaches includes Zipf's law which states that the relative probability of a request for the ***i***'th most popular object is inversely proportional to ***i*** [14]; Zipf' law may be stated mathematically as:

$$\mathbf{f(k;\, s,\, N)=} \quad \frac{\mathbf{1}}{\mathbf{k^{s}H_{N,s}}} \tag{3}$$

Where $H_{N,s}$ is the Nth generalized harmonic number. This last is the sum of the reciprocals of the first n natural numbers. With a naked eye, Zipf's law calculation for the fuzzy control system may take a lot of mathematical operations. The other approach is temporal locality; it refers to the property that referencing behavior in the recent past is a good predictor of the referencing behavior to be seen in the near future. In this context, temporal locality refers to the property that the probability of referencing a particular object decreases with an increasing time of last reference to that object. Mahanti et al. [15] developed a quantitative measure of temporal locality defined for a particular document D_i at stack depth j as shows Eq. (4).

$$\mathbf{T_{ij}} = \frac{\text{Fraction of references to } D_i \text{ made when } D_i \text{ is at stack depth } j}{\text{Fraction of total references made when } D_i \text{ is at stack depth } j} \tag{4}$$

Furthermore, measuring distance in terms of time and its consideration in fuzzy controller also modifies searching behavior, differencing near from distant caches; thanks to this criteria, it is possible both avoid possible unnecessary packets waste send to distant caches and improves hierarchy design, changing host place making it more useful for caching service improvement.

Finally, considering busy rate provides an indirect mechanism for load balancing, making the lesser busy caches take load trying to diminish it in more busy caches, distributing workload more appropriately for each device; this way lead hierarchy to act really as a team.

6. CONCLUSIONS

This experiment presents a series of contributions. First, the use of fuzzy logic as tool to control a caching system which needs improvements regarding to packets transmission; fuzzy controller allows allocate more appropriately the resources compared with other approaches. Intelligent techniques implementation for this type of protocols may be incorporate, in the future, in new generation devices as soft-switches with important processing capabilities, which allows an adequate execution of controller and algorithms for support it.

Three input variables together work in synergic way to approach different caching scenarios. These variables provide criteria to improve selection of candidacy of each cache. The expert plays a major role in design and implementation of this fuzzy control system, because there are fuzzy sets which ranges must be tuned depending on specific system knowledge (quantity of caches, architecture, clients to serve, etc.). This strategy allows improving the quality of service perceived by the user who access to documents in an institutional network, transparently, neither reaching high processing costs, nor money costs. Its implementation will not require significant changes in already existing devices or programs which carry out caching task.

7. REFERENCES

[1] D. Wessels, K. Claffy, "Internet Cache Protocol (ICP), version 2". **RFC2186**, 1997.

[2] C. Lindemann, O. Waldhorst. "Evaluating Hardware and Software Web Proxy Caching Solutions". **Report for Milestone 1 of the Project**. 2000.

[3] A. Chankhunthod, P.B. Danzig, C. Neerdaels, M.F. Schwartz, and K.J. Worrell. "A hierarchical Internet object cache". **In Proceedings of 1996 USENIX Technical Conference**, pages 153--163, January 1996.

[4] L. Cheung, Y. Kwok, "A Fuzzy Load Balancing Service for Network Computing Based on Jini". **Euro-Par 2001: Parallel Processing: 7th International Euro-Par Conference Manchester, UK August 28-31, 2001, Proceedings**. P. 183.

[5] V. Valloppillil. "Cache Array Routing Protocol v1.0". **Internet Draft**, 1998.

[6] M. Hamilton, A. Rousskov, D. Wessels. "Cache Digest specification – version 5". 1998.

[7] I. Cooper. "Inter Cache Co-operation, Protocol Extensions". **Internet Draft**, 1999.

[8] P. Vixie, D. Wessels. "Hyper Text Caching Protocol (HTCP/0.0)". **RFC2756**, 2000.

[9] D. Wessels, K. Claffy. "Application of Internet cache protocol (ICP), version 2". **Internet Engineering Task Force, Internet Draft: draft-wessels-icp-v2-appl-00**. May 1997.

[10] E. Kawai, K. Chinen, S. Yamaguchi, H. Sunahara. "An Analysis of the Number of ICP Packets on the Distributed WWW Caching System. Parallel Processing, 1999". **Proceedings. 1999 International Workshops on 21-24 Sept. 1999** Page(s):234 – 239.

[11] E. Kawai, K. Chinen, S. Yamaguchi, H. Sunahara. "A Quantitative Analysis of {ICP} Queries at the Distributed {WWW} Caching System". **Proceedings of th {INET}'99 Conference**.

[12] P. Rodriguez, C. Spanner, E. Biersack. "Analysis of Web Caching Architectures: Hierarchical and Distributed Caching". **IEEE/ACM TRANSACTIONS ON NETWORKING**, VOL. 9, NO. 4, AUGUST 2001.

[13] B. Kosko, "Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence", **Prentice Hall, New Jersey**, 1992.

[14] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "On the implications of Zipf's law for web caching," **Technical Report CS-TR-1998-1371**, University of Wisconsin, Madison, Apr. 1998.

[15] A. Mahanti, D. Eager, and C. Williamson. "Temporal locality and its impact on web proxy cache performance". **Performance Evaluation Journal: Special Issue on Internet Performance Modelling,** 42(2/3):187–203, September 2000.