E-MACSC: A Novel Dynamic Cache Tuning Technique to Maintain the Prescribed Minimum Hit Ratio Consistently for Internet/WWW Applications

Richard S. L. Wu[1], Allan Kang Ying Wong[1] and Tharam S. Dillon[2]
[1]Department of Computing, Hong Kong Polytechnic University, Hong Kong SAR, PRC

[2]Faculty of Information Technology, University of Technology, Sydney Broadway, N.S.W. 2000

Abstract: The novel E-MACSC technique is for efficacious dynamic cache tuning. It is an enhanced version of the previous MACSC (*model for adaptive cache size control)* approach by eliminating the unpredictable time elements. The enhancement is achieved by substituting the *point-estimate* method adopted by the MACSC with the M^{3}RT, which is a *micro* IEPM (*Internet End-to-End Performance Measurement*) technique. All the preliminary test results show that the E-MACSC not only upkeeps the prescribed minimum hit ratio consistently but also produces up to 5% better hit ratio than the MACSC. This makes the E-MACSC more suitable for time-critical applications because it requires shorter execution time, which means better timeliness.
Keywords: E-MACSC, dynamic cache tuning, point-estimate, M^{3}RT, IEPM, popularity ratio (PR)

1 Introduction

Browsing and data retrieval are normal World Wide Web (WWW) operations in this Internet era. Yet, studies (e.g. [1, 2]) show that: a) the size of a web page increases around 15% monthly, and b) comparatively the Internet backbone capacity improves only about 60% yearly. If this trend continues the Internet will come to a standstill due to congestion. An effective solution to alleviate WWW congestion is caching [3]. For example, if a cache has a 75% hit ratio and the average data retrieval time (latency without caching) is T, then the speedup by caching is S = T/(0.75*0 + 0.25*T) or 400%. Since the data found in the cache needs no long-haul data transfer across the Internet, more backbone bandwidth is freed for public sharing. The data to be retrieved may have time constraints, for instance, making a sound real-time investment decision before a deadline. Then, the data timeliness or freshness on top of correctness is significant [4]. This is well reflected by the following Zurich IBM comment: "... Distance still matters in the Information Age. Geographically distributed caching servers exist because the Internet alone cannot satisfy the desire to provide content quickly…".

The logical solution to secure timely data retrieval is to monitor and control the *roundtrip time* (RTT), which is the latency for sending a request and getting the result successfully. The CARP (*Cache Array Routing Protocol* [17]) is such an algorithm that cuts RTT by maximizing the hit ratio in a cluster by load balancing. In the E-MACSC context WWW data retrieval is a client/server relationship [6]. The proxy needs to approach the web-server only when it cannot find the data object in its local data cache (LDC), as shown in Figure 1. The RTT between points (A) and (C) is broken down in this paper into several basic elements: *client/server communication time* (T_{comm}), *queuing delay at the server side* (TQD), *service time by the server (*T_{serv}), and *context switching delay* (T_{CS}) *by the operating system at the server side*. The delay for queuing and service is the server's "*Local response time*". The details of these elements are: *a)* T_{comm} is the average time for the end-to-end "*Send*" and "*Response*" path (across the channel), *b)* T_{QD} depends on the rate of the incoming requests and the current server loading, and *c)* T_{serv} has two time components: T_{locate} for locating the data object and $T_{retrieve}$ to retrieve it; $T_{serv} = T_{locate} + T_{retrieve}$.

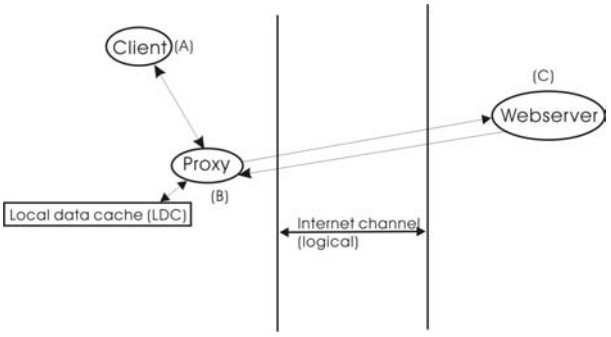

Figure 1. Data retrieval over an Internet channel

Since T_{serv} involves the Internet DNS (*Domain Name System*) and data transfer, it incurs a significant delay. If the proxy server is equipped with an efficient local cache so that the data object can be retrieved directly, then T_{serv} is shortened due to T_{locate} and $T_{retrieve}$ elimination. The MACSC maintains the minimum hit ratio prescribed to the cache so that shorter RTT is guaranteed. This kind of technique is essential for good performance by small caching systems [13]. It helps paid-ISP's (*Internet Service Providers*) to deliver fast data retrieval to keep customers happy [5]. The need for a high hit ratio incessantly inspires different relevant areas of research. The more popular topics include replacement strategies [3], caching of dynamic files [8], leveraging object popularity as an extra parameter [9], and adaptive caching (or *caching adaptivity* [10]). The previous MACSC technique for on-line dynamic cache tuning is a form of caching adaptivity. It leverages the relative data object popularities as the only parameter [18]. Our continued study of the MACSC reveals that it has a timeliness problem for serious real-time applications. To resolve this problem the E-MACSC (*Enhanced MACSC*), which uses the M^3RT *micro* IEPM or $M-IEPM$ (*Internet End-to-End Performance Measurement*) technique [19,20,21], is proposed. This $M-IEPM$ technique is predictive, fast, accurate and insensitive to traffic patterns because it is derived from the *Central Limit Theorem*. In operation the two E-MACSC constituent objects run and collaborate in a concurrent manner; that is, "$E-MACSC = MACSC + M^3RT$".

2 Related Work

The E-MACSC is an enhancement of the previous MACSC (*model for adaptive cache size control*) [18], which computes the *standard deviation* (SD) of the current *relative popularity* (RP) among the data objects on-line. The RP is embedded in the plot of "*access frequencies versus the data objects*", known as the *popularity distribution* (PD). The data objects in the LDC belong to the "*currently hot*" subset of the repertoire in the web-server. The PD shape changes according to the seasonal user preference, and so does the SD of the PD. The MACSC model makes use of the difference between two consecutive SD values in two ways, as shown by equations (1) and (2), which compute the cache size (CS) adjustment differently. The following two terms: $(SD_{ThisSample}/SD_{LastSample})^2$ and $(SD_{ThisSample}/SD_{LastSample})$ are the *popularity ratios* (PR).

$$CS_{Adjusted\_VR} = CS_{Old\_VR} * (\frac{SD_{ThisSample}}{SD_{LastSample}})^2 .....(1)$$

$$CS_{Adjusted\_SR} = CS_{Old\_SR} * (\frac{SD_{ThisSample}}{SD_{LastSample}}).....(2)$$

The initial cache size for the MACSC operation is $C_{SD} = N_{objects} * S_{AverageObjectSize}$, where:

a) C_{SD} is the initial cache size for the prescribed confidence level or hit ratio, in terms of the number of standard deviations.

b) $N_{objects}$ is the number of data objects in the central region of the initial PD chosen to satisfy the prescribed confidence.

c) $S_{AverageObjectSize}$ is the average size of the data objects in the repertoire.

The MACSC efficacy depends on its capability to precisely capture the continuous changes in the relative probability (RP) profile. Under all operational conditions the MACSC should strive to maintain the prescribed minimum hit ratio (e.g. one SD or 68.4%) by adjusting the cache size adaptively. A larger PD (probability distribution) variability needs a larger cache for the given hit-ratio because it includes more data objects. The *popularity ratios* (PR) by equation (1) and equation (2) produce very different effects. The suffixes: "*Adjusted _VR*" and "*Adjusted _SR "* differentiate the PR by the VR (*variance ratio*) from that by the SR (*standard deviation ratio*). If S_b=$SD_{ThisSample}$ and S_a =$SD_{LastSample}$ are assumed, $(S_b/S_a)^2$ and (S_b/S_a) are the VR and SR ratios respectively. The MACSC estimates the SD or $\delta_{\bar{x}}$ and the mean (i.e. $\bar{x}$) by direct data measurement to get the shape of the current PD. This is carried out with the "*N-equation"*: $E\mu = k\delta_{\bar{x}} = k(\delta_x/\sqrt{N})$ or $N = (k\delta_x/E\mu)^2$, which, like the M^3RT, is derived from the *Central Limit Theorem* [7]. The meanings of the parameters are as follows: a) E is the fractional error between $\bar{x}$ and the *true mean* μ, b) k is the number of standard deviations for $\bar{x}$ to deviate from μ but within an acceptable tolerance, c) N is the number of x data

samples to satisfy the *N-equation*, and d) δ_x is the true (population's) SD. The analysis of the preliminary MACSC performance data indicates that the speed and accuracy for estimating $\delta_{\bar{x}}$ and $\bar{x}$ should be improved. It is otherwise difficult to apply the MACSC techniques in real-time applications due to the unpredictable nature of the two time elements: $T_{Sampling}$ and T_{PE} (to be explained later). The $\bar{x}$ inaccuracy is caused by the absence of historical $\bar{x}$ data in the PE estimation process. The key issue here is how to incorporate past performance into the current $\bar{x}$ estimation. A thorough explanation of the literature convinces us that the M^{3}RT technique is suitable for the job. In its $M-IEPM$ form this technique operates as an independent logical entity/object to be invoked for estimation service by message passing. The preliminary experimental E-MACSC results with different values for the damping factor P in equation (3) show that with the $M-IEPM$ support the $\bar{x}$ accuracy is significantly improved. This can yield up to 5% more hit ratio than the MACSC with the *point-estimate* method. The correlation among the P value, the hit ratio and cache size will be explored thoroughly in the next phase of the research.

3 The E-MACSC Details

The E-MACSC has four main operational modules (Figure 2):

a) M1-The CS (cache size) adjustment computation module that calculates by either the VR or the SR approach (i.e. equation (1) or equation (2)).
b) M2 - The $\bar{x}$ (sample mean) predicted by the M^{3}RT $M-IEPM$ (i.e. $\bar{x} \approx A_i$ in equation (3)) technique that includes the past $\bar{x}$ in the current prediction.
c) M3 – It carries out the actual dynamic cache tuning with the adjustment computed above.
d) M4 - Memory recycling to support the dynamic cache tuning process.

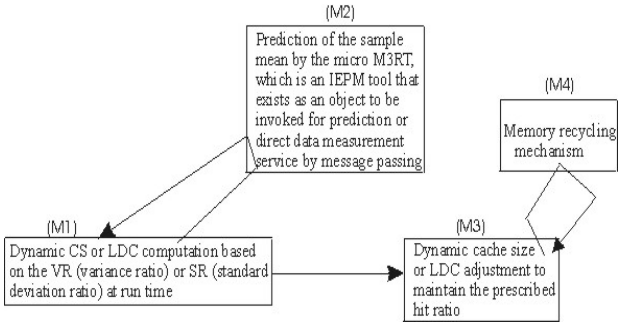

Figure 2. The high-level E-MACSC model

The core of the M^{3}RT is the *Convergence Algorithm* (CA) represented by equation (3). The parameters are: a) A_i is the predicted $\bar{x}$ for i^{th} cycle, with $\bar{x} \approx \mu$ [19,20], b) m_i^j is the j^{th} sample mean in the i^{th} cycle, c) f is the *flush limit* for fast convergence to the predicted mean (i.e. A_i), and there are $(f-1)$ number of m_i^j in every i^{th} cycle ($j = 1,2,...,(f-1)$), and d) P is the damping factor chosen for smooth A_i convergence. The optimal range for f is always between 9 and 16 as previously confirmed empirically [9].

$$A_i = \frac{PA_{i-1} + \sum_{j=1}^{f} m_i^j}{P+f} \quad \text{.....(3)}$$

It is expected the M^{3}RT provides three advantages. Firstly, it should make the E-MACSC yield a higher hit ratio than the previous MACSC that has *point-estimate* or PE rather than IEPM support. This is due to the fact that the M^{3}RT yields more accurate $\bar{x}$ or A_i predictions than the PE approach. Secondly, the M^{3}RT estimation is faster with predictable execution time. For example, if it is assumed that for a sample size of 50 (i.e. $SS = 50$) the PE process yields 25 and 7 for $\bar{x}$ and $\delta_{\bar{x}}$ respectively, for $N = (k\delta_x / E\bar{x})^2 \approx 126$. This implies that the sample size of $SS = 50$ is incorrect and more data should be sampled (126-50=76) to recalculate $\bar{x}$ and $\delta_{\bar{x}}$. The recalculation repeats until N is acceptable, and this gives $T_{Sampling}$ and T_{PE} the unpredictable nature as explained later. The M^{3}RT does not have this problem because at the system

steady state the current $\bar{x}$ (or A_i) and $\delta_{\bar{x}}$ are always computed accurately with the f (*flush limit*) value chosen from the range: 9 to 16. Thirdly, the M^{3}RT accuracy eliminates T_{PE} repetition because $A_i = \bar{x}$ is very close to the true mean μ [19] and is always available to immediately satisfy the *N-equation*. Conceptually $\delta_{\bar{x}}$ can now be computed in a single step by $\delta_{\bar{x}} = \frac{EA_i}{k}$, and $A_i = \bar{x} \approx \mu$ implies $\delta_{\bar{x}} \approx \delta_x$. The convergence speed to get A_i is associated with the P value, which smoothens out the oscillations in the convergence process [19]. Ideally the "*dynamic cache tuner*" should detect and capture the smallest changes in the relative popularity profile of the data objects quickly and accurately so that the seasonal change in user preference can be optimally reflected.

4 Experimental Results

Many experiments were carried out with the Java-based E-MACSC prototype over the Aglets mobile agent platform. This stable Java-based platform [10] yields credible and scalable experimental results, which can be repeated in the open Internet environments. In the experiments the client and servers in Figure 1 become Java objects known as the *aglets* (*agile applets*). The essence of the setup for the experiments is shown in Figure 3. Figure 4 shows one of the experimental E-MACSC results, with $P = 1$, for demonstration purposes. In this case, the driver *aglet* operates at the rate of 0.67 requests per second (rate of the clock interrupts). The requested data object is determined by generating a random number first with the chosen distribution. Then, the data object is interpolated along the X-axis (e.g. the normal PD distribution illustrated in Figure 3). The proxy *aglet* finds the requested data by both the PE and M^{3}RT approaches running concurrently. If the data object cannot be found in the cache, then the proxy *aglet* invokes the I/O *aglet* to find it from the repertoire. This mimics the presence of a web server that handles 40,000 different files of an average size of 5K bytes. The size of the LDC is initialised to 50 Megabytes for the prescribed minimum hit ratio of one standard deviation (SD = 1) or 68.3%. The replacement algorithm for the LDC is the basic LRU (*least frequently used*) approach. In all the experiments 250,000 data requests (i.e. 250,000 clock interrupts) were generated. Figure 4 shows how the E-MACSC controls the LDC size optimally and yields better hit ratio than the previous MACSC model. The analyses of all the test cases show that the hit ratio by the E-MACSC is up to 5% higher, as demonstrated in Figure 4.

The computation time for the PE approach (T_{Total}) has several basic time components: a) $T_{Sampling}$ for collecting enough samples to satisfy the N-equation, b) T_{PE} for computing $\bar{x}$ and $\delta_{\bar{x}}$ in order to satisfy the same criterion above, and c) T_{PR} for computing the adjustment and carrying out the physical dynamic cache tuning; $T_{Total} = T_{Samping} + T_{PE} + T_{PR}$. The time $T_{Sampling}$ is a variable because it depends on the

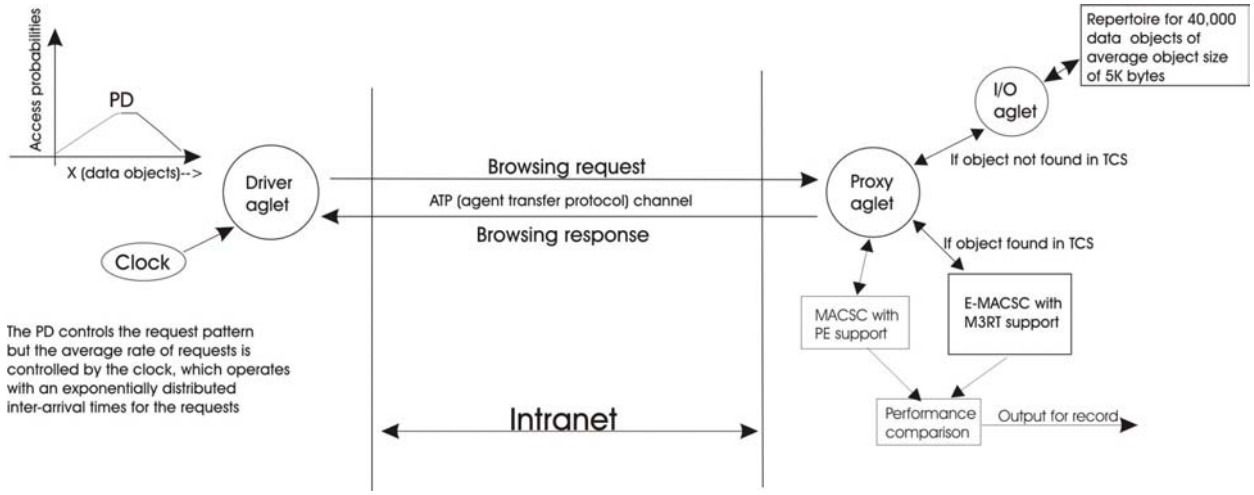

Figure 3. The testing environment for E-MACSC dynamic cache tuning

unpredictable *inter-arrival time* (IAT) between two samples. For one of the experiments, with $SS = 50$, the timing analysis of T_{PE} alone by the *Intel's VTune Performance Analyzer* [22] shows that it needs an average of 1,673,100 clock cycles just for one computation pass (collecting 50 samples to compute $\bar{x}$ and $\delta_{\bar{x}}$). In fact, it is normal for such a calculation pass to be repeated until the N-equation is satisfied. The *VTune* shows that the $M - IEPM$ running on the same platform alone requires only 211 clock cycles on average to predict $\bar{x} \approx A_i$ and about 110 clock cycles to compute $\delta_{\bar{x}}$ by: $\delta_{\bar{x}} = \frac{EA_i}{k}$. In the light of this experiment, the M^{3}RT saves roughly $(16731000 - 321)$ or 16730679 clock cycles in a predictable manner. The problem with the PE approach is that it may need P passes to compute $\bar{x}$ and $\delta_{\bar{x}}$ that satisfies the N-equation. Therefore, the number of clock cycles saved by the E-MACSC compared to the MACSC is conceptually around Z*16730679. In all the test cases so far, Z varies from 10 to 40 depending on the average IAT. On a platform such as the Sony R505CT that operates at 143MHz, the *point-estimate* method with no M^{3}RT support is about 16730679*(1/143)*10^{-6}≈11.7ms slower per pass for the $\bar{x}$ and $\delta_{\bar{x}}$ computations. Therefore, the E-MACSC model is more applicable for time-critical applications for better timeliness.

The E-MACSC experimental results show that this novel technique yields a higher LDC hit ratio with more cache space consumption. For example, in the case shown in Figure 4, the average 5% higher hit ratio consumes 14 megabytes more cache space in the dynamic cache tuning process. The higher cache space consumption is the side effect from the fast and accurate $\bar{x}$ and δ_x predictions by the $M - IEPM$. This effect is closely associated with the choice of the value for the damping factor P, and the rationale behind this association will be thoroughly explored in the near future. The general conclusion from the preliminary E-MACSC experimental results are as follows: a) it is essential to get $\bar{x}$ accurately so that $\delta_{\bar{x}}$ can be computed to gauge the popularity ratio(s) correctly for meaningful cache adjustment computation, b) the P factor filters out the x oscillations to yield a smooth and accurate A_i convergence, but meanwhile this leads to small variations for $\delta_{\bar{x}}$ (i.e. $\delta_{\bar{x}} = \frac{EA_i}{k}$) and thus less sensitivity for the E-MACSC in response to small changes in the relative popularity profile of the data objects, c) less sensitivity yields less system responsiveness and more inertia, and d) the impact by the inertia are: i) a previous cache elongation leads to the higher hit ratio because after a cache elongation any change in user preference does not shrink $\delta_{\bar{x}}$ immediately, and ii) a previous cache shrinkage would lower the hit ratio because of slow response by the dynamic cache tuning process. The third and fourth observations above imply that the sensitivity and responsiveness by the E-MACSC to small changes need improvement.

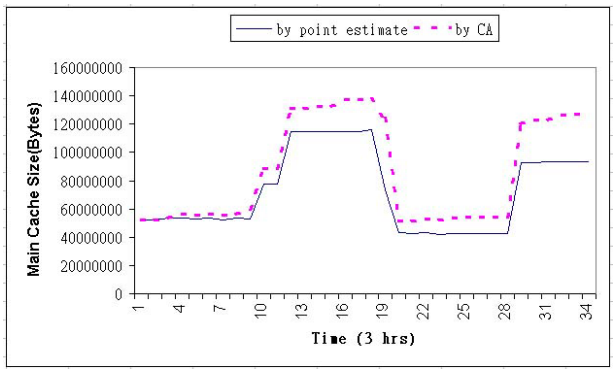

Figure 4. A case of more accurate LDC size control supported by M^{3}RT (or CA) with P = 1

5 Conclusion

The novel E-MACSC technique for dynamic cache tuning is an enhanced version of the previous MSCSC model. The enhancement is achieved by replacing the point-estimate method for computing $\bar{x}$ and $\delta_{\bar{x}}$ with the M^{3}RT $M - IEPM$ technique. In the E-MACSC the two component modules: the dynamic cache tuner (i.e. the MACSC without PE) and the M^{3}RT run concurrently. The tuner gets the accurate $\bar{x}$ and $\delta_{\bar{x}}$ predictions immediately from the M^{3}RT by message passing whenever it needs them. The accuracy of these predictions supported by the chosen f (*flush limit*) value eliminates the unpredictable $T_{Sampling}$ and T_{PE} time elements from the MACSC. It

allows the E-MACSC to gain up to a 5% higher hit ratio by using less cache memory as demonstrated by the case in Figure 4. The next phase planned for the research is to explore and define: a) the association between the P factor and the gain in hit ratio by the E-MACSC, and b) the correlation between Z and the IAT.

6 Acknowledgement

The authors thank the Hong Kong PolyU for the research grant: HZJ91

7 References

[1] K. Bharat and A. Broder, Measuring the Web, *http://www.research.digital.com/SRCwhatsnew/sem.html*

[2] J. Almeida and P. Cao, Measuring Proxy Performance with the Wisconsin Proxy Benchmark, Technical Report 1373, Computer Science Department, University of Wisconsin-Madison, 1998

[3] C. Aggarwal, J.L. Wolf and Philip S. Yu, Caching on the Word Wide Web, *IEEE Transactions on Knowledge and Data Engineering*, 11(1), 1999

[4] J. A. Stankovic et al, Deadline Scheduling for Real-Time Systems, *Kluwer Publishers*, 1998

[5] C. Kenyon, The Evolution of Web-Caching Markets, *IEEE Computer*, Nov. 2001, 128-135

[6] S.M. Lewandowski, Frameworks for Component-Based Client/Server Computing, *ACM Computer Survey*, March, 1998 [7] I. Mitrani, Probabilistic Modelling, *Cambridge University Press*, 1998

[8] M. Arlitt et al., Evaluating Content Management Techniques for Web Proxy Caches, *Proc. of the 2nd Workshop on Internet Server Performance*, 1999

[9] K. Cheng and Y. Kambayashi, LRU-SP: A Size-Adjusted and Popularity-Aware LRU Replacement Algorithm for Web Caching, *Proc. of Computer Software and Applications Conference (COMPSAC 2000)*, 2000

[10] S. Michel et al., Adaptive Web Caching: Towards a New Global Caching Architecture, *Journal of Computer Networks and ISDN Systems*, 1998

[11] L. Breslau et al., Web Caching and Zipf-like Distribution: Evidence and Implications, *Proc. of the IEEE INFOCOM*, 1999

[12] P. Cao and S. Irani, Cost Aware WWW Proxy Caching Algorithms, *Proc. of the USENIX Symposium on Internet Technology and Systems*, December, 1997

[13] D. Wessels, Web Caching, O'Reilly & Associates Inc., 2001

[14] T. Lakshman U. Madlow, The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss, *IEEE/ACM Transactions on Networking*, 5(3), June 1997

[15] H. Elaarag, Improving TCP Performance over Mobile Networks, *ACM Computing Surveys*, 34(3), 2002

[16] Allan K.Y. Wong, Wilfred W.K. Lin, May T.W. Ip and Tharam S. Dillon, Genetic Algorithm and PID Control Together for Dynamic Anticipative Marginal Buffer Management: An Effective Approach to Enhance Dependability and Performance for Distributed Mobile

[17] V. Valloppilli and K. W. Ross, Cache Array Routing Protocol v1.0, Internet Draft <darft-vinod-carp-v1-03.txt>

[18] Allan K.Y. Wong, May T.W. Ip and Richard S.L. Wu, A Novel Dynamic Cache Size Adjustment Approach for Better Data Retrieval Performance over the Internet, Computer Communications, vol. 26, 2003

[19] Allan K.Y. Wong, Tharam S. Dillon, Wilfred W.K. Lin and T.W. Ip, M2RT: A Tool Developed for Predicting the Mean Message Response Time for Internet Channels, The Journal Computer Networks (and ISDN Systems), vol. 36, 2001

[20] Allan K.Y. Wong, May T.W. Ip and Tharam S. Dillon, M3RT: An Internet End-to-End Performance Measurement Approach for Real-Time Applications with Mobile Agents, The Proc. of the ISPAN2002, the Philippines, 2002

[21] L. Cottrel, M. Zekauskas, H. Uijterwaal, and T. McGregor, Comparison of Some Internet Active End-to-End Performance Measurement Projects, http://www.slac.stanford.edu/comp/net/wan-mon/iepm-cf.html

[22] Intel@ VTune™ Performance Analyzer, www.intel.com/support/performancetools/vtune/v5/