E-MACSC: A NOVEL DYNAMIC CACHE TUNING TECHNIQUE TO MAINTAIN THE HIT RATIO PRESCRIBED BY THE USER IN INTERNET APPLICATIONS

Richard S.L. Wu and Allan K.Y. Wong
Department of Computing, Hong Kong Polytechnic University, Hong Kong SAR, PRC
Email: csslwu@comp.polyu.edu.hk, csalwong@comp.polyu.edu.hk

Tharam S. Dillon
Faculty of Information Technology, University of Technology, Sydney Broadway, N.S.W. 2000
Email: tharam@it.uts.edu.au

Keywords: E-MACSC, dynamic cache tuning, popularity ratio, point-estimate, IEPM.

Abstract: The E-MACSC (*Enhanced Model for Adaptive Cache Size Control*) is a novel approach for dynamic cache tuning. The aim is to adaptively tune the cache size at runtime to maintain the prescribed hit ratio. It works with the popularity ratio (PR), defined by the standard deviations sampled for the relative popularity profile of the data objects at two successive time points. The changes in the PR value reflect the shifts of users' preference toward certain data objects. The E-MACSC makes use of the *Convergence Algorithm* (CA), which is an IEPM (*Internet End-to-End Performance Measurement*) technique that measures the mean of a waveform quickly and accurately. Accuracy of the measurement is independent/insensitive to the waveform pattern because the CA is derived from the *Central Limit Theorem*.

1 INTRODUCTION

The E-MACSC (*Enhanced Model for Adaptive Cache Size Control*) model proposed in this paper is a novel approach for dynamic cache tuning. It maintains the prescribed hit ratio for the local cache adaptively and consistently by adjusting the cache size on the fly. The adjustment is performed according to the current popularity ratio (PR). The E-MACSC is more efficacious than its MACSC (*Model for Adaptive Cache Size Control*) predecessor (Allan, 2003). The E-MACSC and MACSC tuners are especially suitable for small caching systems of limited memory resources. In fact, in the field the number of small caching systems, which usually cost less than USD$1000 (Wessels, 2001), is substantial. In these systems poor caching will lead to excess memory consumption and poor performance because of frequent task suspensions. The E-MACSC is good news for e-business applications because it shortens the RTT and keeps the customers happy. Its rationale is: "*optimal memory usage to maintain the given hit ratio*". For example, maintaining a 70% hit ratio means a RTT (*roundtrip time*) speedup of $S = RTT/(0*0.7 + RTT*0.3) \approx 3.33$. Such speedup inspires the widespread quest for different solutions to yield high hit ratios, such as the replacement strategies (Aggarwal, 1999).

2 RELATED WORK

The E-MACSC is the deeper work that bases on our previous research MACSC experience in the area of dynamic cache tuning. The focus is especially on leveraging the relative object popularity profile as the sole control metric.

2.1 Popularity Ratio

The dynamic adjustment of the cache size by the E-MACSC and MACSC models is based on the *popularity ratio* (PR) (Allan, 2003). It is the ratio of standard deviations or variances of the current popularity profile of the data objects at two consecutive time points. It is derived from the Zipf-like behaviour (Breslau, 1999; Zipf) intrinsic to cached data objects. The behaviour is represented by

J. Ascenso et al. (eds.), e-Business and Telecommunication Networks, 65–72.

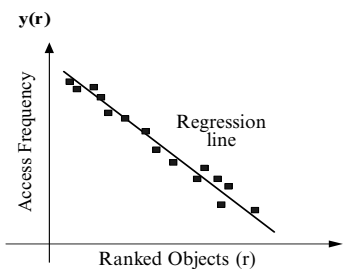

Figure 1a: Zipf-like distribution (log-log).

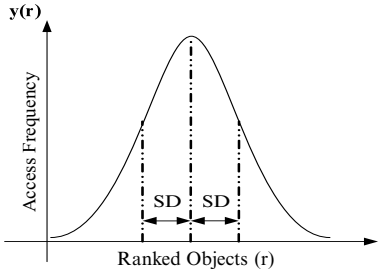

Figure 1b: Bell shape distribution. (SD – standard deviation)

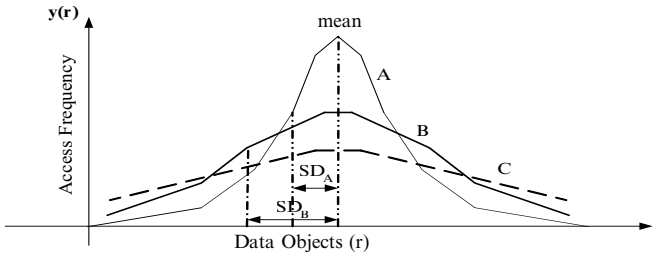

Figure 1c: Popularity distribution changes over time and reflects the change in user preference.

the log-log plot in Figure 1a, which shows that the chance (Y-axis) for the j^{th} popular object in the *sorted/ranked list* (X-axis) to be accessed is proportional to $(1/j)^{\beta}$, for $0 < \beta \leq 1$.

The original plot of the raw scattered data in Figure 1a can be approximated by the linear regression: $y(r)=f_{highest} - \gamma(r-1)$, where γ is a curve fitting parameter, r the ranked position of the object, and $f_{highest}$ the highest access frequency for the "*ranked-first*" object in the set. If this regression is mapped into the bell curve in Figure 1b, it becomes the *popularity distribution*, which shows the current profile of the relative popularity of the data objects. The central region of this curve includes the more popular objects, and $f_{highest}$ is the "*mean of the popularity distribution*" in the E-MACSC context (Allan, 2003). The shape of the popularity distribution changes over time due to the shifts in the user preference towards specific data objects. The shift is immediately reflected by the current standard deviation. For example, the three curves: A, B and C in Figure 1c mimic the different popularity distribution shapes of three different time points, and SD_A and SD_B are the standard deviations of A and B (at different time points) respectively.

2.2 The MACSC Predecessor

The MACSC tuner leverages the relative popularity of the data objects as the sole control parameter to achieve dynamic cache size tuning over the web. Leveraging the relative object popularity to heighten the hit ratio in a timely manner is a recent concept. For example, it is used as an additional parameter for the first time in the "*Popularity-Aware Greedy Dual-Size Web Proxy Caching Algorithms*" (Jin, 2000). The issue of how to utilize the relative object popularity alone for gaining higher hit ratio was never addressed before the MACSC model. As an additional parameter in a replacement algorithm (Stefan, 2003) the potential benefits from leveraging it are easily offset by the long algorithm execution time due to heavy parameterisation.

The running MACSC tuner traces all the popularity distribution changes continually and uses them timely to adjust the cache size adaptively. This tuning process maintains the prescribed minimum hit ratio consistently, and the *cache adjustment size* (CAS) is based on one of the following two equations:

$$CAS_{VR} = CacheSize_{old} * \left\{ \frac{SD_{current}}{SD_{last}} \right\}^2 \quad \text{.....(2.1)}$$

$$CAS_{SD} = CacheSize_{old} * \left(\frac{SD_{current}}{SD_{last}} \right) \quad \text{.....(2.2)}$$

The popularity ratios for equation (2.1) and equation (2.2) are the *variance ratio* (VR), and the *standard deviation ratio* (SR) (i.e. $SD_{current}$ over SD_{last}) respectively. Although the VR-based tuner (equation (2.1)) is more effective in maintaining the given hit ratio, it consumes too much memory and this makes it impractical for small caching systems with limited memory resources (Wessels, 2001; Allan, 2003). The focus here is the SR approach.

The MACSC efficacy depends on the accuracy of the popularity distribution's current standard deviation. The MACSC uses the *Point-Estimate* (PE) approach because it is derived from the Central Limit Theorem and therefore its accuracy is insensitive to traffic patterns. The $\sqrt{N}-equation$ (Chis, 1992) means the following statistical relationship:

$$E\lambda = k\delta_{\bar{x}} = k(\delta_x / \sqrt{N})$$

and the parameters are:

a) *Fractional error tolerance* (*E*): It is the error between λ (ideal/population mean value) and *m* (the mean estimated from a series of sample means $\bar{x}$ of sample size $n \geq 10$, on the fly).

b) *SD tolerance* (*k*): It is the number of standard deviations (SD) that *m* is away from the true mean λ but still be tolerated (same tolerance connotation as *E*).

c) *Predicted standard deviation* ($\delta_{\bar{x}}$): It is estimated from the same series of sample means $\bar{x}$ of sample size $n \geq 10$ by the following:

$\delta_{\bar{x}} = \delta_x / \sqrt{n}$ that fits the Central Limit Theorem.

d) *Minimum N value*: From the relationship:

$$E\lambda = k\delta_{\bar{x}} = k(\delta_x / \sqrt{N})$$

the minimum sample size *N* to compute the acceptable λ and δ_x with respect to the given *k* and *E* can be estimated. In practice $\bar{x}$ and s_x (standard deviation), estimated from the current data samples, substitute λ and δ_x as follows:

$$N = (\frac{k\delta_x}{E\lambda}) \rightarrow N = (\frac{ks_x}{E\bar{x}})^2$$

In every iteration that estimates *N* with *n* samples until $n \geq N$ convergence has occurred, the sample standard deviation s_x is estimated at the same time as $\bar{x}$. The *n* value increases with the number of estimation iterations involved till the condition: $n \geq N$ is satisfied. The PE estimates $\bar{x}$ statistically from the *n* samples first and then the standard deviation:

$$s_x = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N-1}}$$

where x_i is a data item in the i^{th} sampling round.

The following example illustrates how the PE iterative process satisfies the $n \geq N$ criterion for the $\sqrt{N}-equation$:

a) It is assumed that the initial 60 samples (i.e. sample size of n = 60) have yielded 15 and 9 for $\bar{x}$ and s_x respectively.

b) The given SD tolerance is 2 (i.e. k = 2 or 95.4%), and the fractional tolerance *E* is therefore equal to 4.6% (*E = 0.046*). Both *E* and *k* mean the same error tolerance by the *Central Limit Theorem*.

c) The minimum *N* estimate is now

$$N = (\frac{2*9}{0.046*15})^2 \approx 680$$

The value $N \approx 680$ implies that the initial sample size $n = 60$ is insufficient. To rectify the problem, one of the following methods can be adopted:

a) The first one is to collect (680 – 60) or 620 more samples and re-calculate $\bar{x}$ and s_x. There is no guarantee, however, the estimation would converge to $n \geq N$ and the same process has to be repeated.

b) The second method is to collect another 60 samples and re-calculate $\bar{x}$ and s_x from the total of 120 samples (i.e. $n = 120$ for the 2nd trial). The process repeats with 60 additional new data samples until $n \geq N$ is satisfied.

Practical experience shows that the second method converges much faster because it is common for $\bar{x}$ and s_x to stabilize in the second or third trial. This method is the basis of the core of the PE operation in the MACSC model.

The drawback of the PE process is its unpredictable time requirement to satisfy $n \geq N$ in real-life applications because of the changing IAT between any two samples. Collecting the 680 samples in the example above may take seconds, hours or even days. This kind of time un-predictability reduces the tuning precision of MACSC and makes the hit ratio oscillate in the steady state (Richard 2003). The E-MACSC model, however, does not have such unpredictability in terms of the number of data samples to satisfy $n \geq N$. Even though the IAT of the data samples can vary, the degree of severity on timeliness unpredictability is lessened because the number of data samples is fixed at *F* (the *flush limit* (equation (3.1)). This is made possible by replacing the PE process with the novel M^{3}RT *micro* IEPM or simply referred to as the

μ-IEPM mechanism. The choice of *F* is important for the fastest M³RT convergence, and the best range is: *9 ≤ F ≤ 16* (Allan, 2001). Being *micro* the mechanism operates as a logical object in the E-MACSC framework. The M³RT is a realization of the *Convergence Algorithm*, which is an IEPM technique (Cottrel, 1999) based on the *Central Limit Theorem*. For this reason the M³RT prediction accuracy is insensitive to the waveform/distribution being worked on. Previous Internet experience confirms that the M³RT mechanism always yields consistent performance even when the traffic pattern is changing continuously, for example, switching among the following patterns: Poisson, heavy-tailed, and self-similar.

2 THE E-MACSC DETAILS

The E-MACSC is an enhancement of the MACSC predecessor, which has unpredictable computation time due to: i) the unpredictable number of data samples needed by its statistical PE approach to satisfy the *N* value of the $\sqrt{N}-equation$, and ii) the unpredictable Inter-Arrival Times (IAT) among the samples (data requests). In reality, the IAT pattern affects the accuracy of the computed result because the traffic pattern can be Poisson, heavy-tailed, self-similar, or multi-fractal (Paxson, 1995).

The E-MACSC damps hit-ratio oscillation in dynamic cache tuning by replacing the PE approach with the M³RT μ-IEPM mechanism (summarized by the equations (3.1) and (3.2)), which uses *f =(F-1)* data samples to compute s_x and $\bar{x}$. The preliminary E-MACSC results indicate that the flush limit range: $9 \le F \le 16$ also yields σ_i (equation (3.6)) quickly and accurately. To enhance the sensitivity of equation (3.1) it is transformed through equation (3.3) into the equation (3.4). By arranging $\alpha = \frac{p}{p+f}$ and $(1-\alpha) = \frac{f}{p+f}$ the alternative equation (3.5) is obtained. This means that the previous $\delta_{\bar{x}}$ computation is now replaced by σ_i (equation (3.6)), and thus the PR computation is also σ_i based.

$$M_i = \frac{p * M_{i-1} + \sum_{j=1}^{j=F-1} m_i^j}{p+f} \quad \text{.....(3.1)}; i \ge 1$$

$$M_0 = m_{i=1}^{j=1} \quad \text{.....(3.2)}$$

$$\mathrm{M_i} = \frac{p}{p+f} * \mathrm{M_{i\text{-}1}} + \frac{1}{p+f}\left(\sum_{j=1}^{j=f} \mathrm{m_i^j}\right) \quad \text{.....(3.3)}$$

$$\mathrm{M_i} = \frac{p}{p+f} * \mathrm{M_{i\text{-}1}} + \frac{1}{p+f}(f)\left(\frac{1}{f}\right)\left(\sum_{j=1}^{j=f} \mathrm{m_i^j}\right) \quad \text{.....(3.4)}$$

$$\theta \quad \frac{p}{p+f} + \frac{f}{p+f} = 1$$

$$\Rightarrow \quad \mathrm{M_i} = \alpha * \mathrm{M_{i\text{-}1}} + (1-\alpha) * \left(\frac{\sum_{j=1}^{j=f} \mathrm{m_i^j}}{f}\right) \quad \text{.....(3.5)}$$

$$\sigma_i = \alpha * \sigma_{i\text{-}1} + (1-\alpha) * \sqrt{\frac{\sum_{j=1}^{j=f} \left(\mathrm{m_i^j} - \mathrm{M_i}\right)^2}{f-1}} \quad \text{..... (3.6)}$$

3 E-MACSC VERIFICATION

Many simulations were carried out with the E-MACSC prototype implemented in Java over the controlled Internet environment, as illustrated in Figure 2. The intention is to verify that the M³RT-based E-MACSC model has: a) more stable control, and b) predictably shorter execution time than its PE-based MACSC predecessor. These E-MACSC tests were carried out on the Java-based Aglets mobile agent platform [Aglets], which is chosen for its stability, rich user experience, and scalability. The Aglets platform is designed for applications over the Internet, and this makes the experimental results scalable for the open Internet. The replacement algorithm used in the simulations is the basic LRU (*Least Recently Used*) approach with the "*Twin Cache System* (TCS)" [Aggarawal99]. The TCS was used successfully in the previous MACSC verification and its function is to filter the "*one-timers*", which are considered as caching "noise". The filtration makes the hot data in the cache more concentrated by reducing the noise (Allan, 2003). One-timers are unpopular data objects that are accessed only once over a long period. The driver and the E-MACSC tuner for the proxy server in Figure 2 are aglets (*aglet applets*) that interact in a client/server relationship. The driver generates the input traffic for the E-MACSC operation, with a chosen pattern (e.g. heavy-tailed, self-similar, Poisson, and multi-fractal). The input traffic is generated by one of the following methods: a) choosing a distribution from the table (Figure 2), b) interleaving different waveforms, c) using a pre-collected data trace, or d) collecting actual data

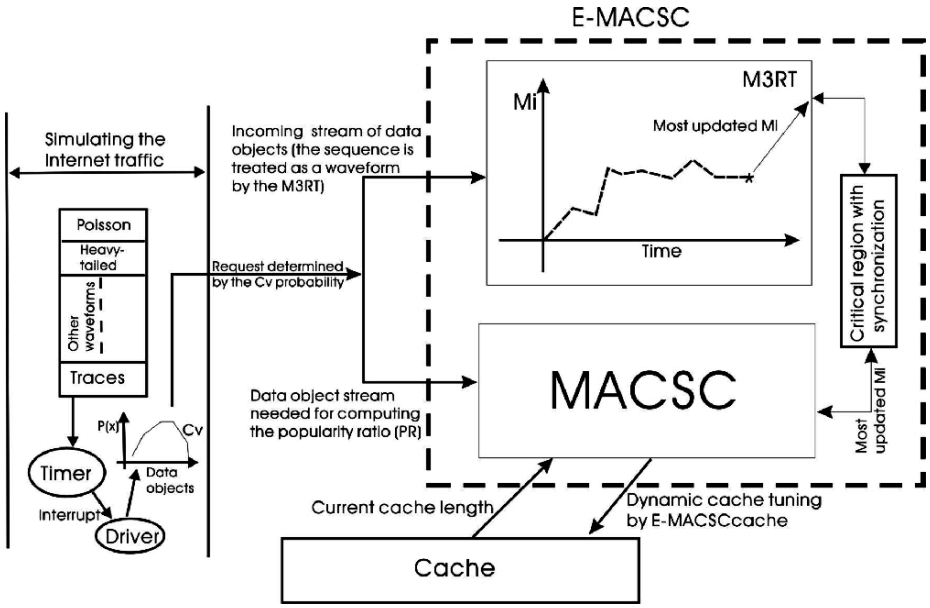

Figure 2: Verification set up for the E-MACSC tuner (for the proxy) by simulation.

object samples on the fly (Allan, 2003)). The simulation results presented in this paper are produced with the second method in two steps. The first step is to choose the waveform to drive the timer interrupt, which generates the IAT intervals. The second step is let the timer interrupt the driver so that it interpolates the unique object identifier from the X-axis of the *Cv* curve. The object identifier is then sent as the request to the server for data retrieval. The *Cv* curve is generated with either one of the four methods for the input traffic. It correlates the access probabilities/frequencies with the corresponding data objects. The object identifier is an integer (e.g. 1, 2, 3…) that uniquely identifies the specific data object.

$$CacheSize_i = CacheSize_{i-1} * \left(\frac{\sigma_i}{\sigma_{i-1}} \right) \quad \text{.....(3.7)}$$

In the verification experiments at least 40,000 data objects of various sizes are used. For example, the simulation results presented here are produced with 40,000 data objects of average size 5k bytes and 1 million data retrieval transactions generated by the driver. The cache size is first initialised to meet the prescribed hit ratio. For example, if the given hit ratio is one popularity-distribution standard deviation or 68.3%, then the initial cache size should be 5k*0.683*40,000 bytes (or 136.6MB). The basic LRU replacement algorithm deletes aged objects in the cache to accommodate the new comers. The simulation results in this paper are *SR* based (equation (2.2)), and in fact, the primary goal of the verification is to confirm that the E-MACSC tuner is more efficacious than its predecessor for small caching system applications. The preliminary results confirm that it is indeed faster and yields higher hit ratio. The popularity ratio in the E-MACSC case is computed with σ_i as shown by equation (3.7). If the given hit ratio is 68.3%, then the cache size *Z* should be initialised to $Z \approx Q * Sz$, where *Sz* is the average object size and Q equal to the number of objects that represents one standard deviation (i.e. 68.3%).

The simulation results shown in Figure 3 are produced with: 40,000 data objects of an average 5k byte size, the given hit ratio of 68.3%, 1 million data E-MACSC tuner with M^3RT support yields the highest hit ratio as compared to the "*fixed cache system* (FCS)" that works with a static cache size and the PE based MACSC tuner. Figure 4 shows the impacts by the α values, where *p* for proportional (damping) control:

$$\alpha = \frac{p}{p+f}$$

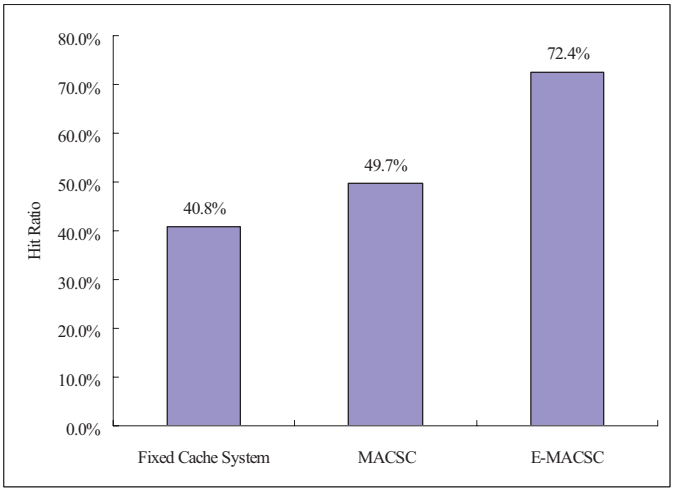

Figure 3: The comparison of the hit ratio between different algorithms. (Input traffic cyclical sequence: 3k→5k→4k→8k, α = 0.99).

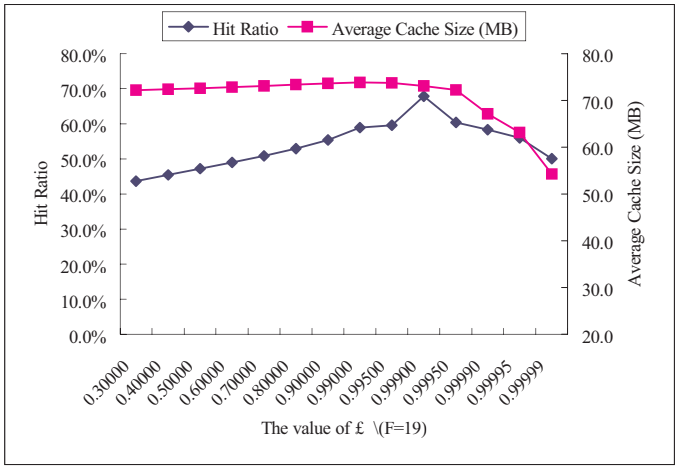

Figure 4: Hit ratio, cache size and α; F = 19, α = 0.99.

F=19 (i.e. *f=F-1=18*) produces the fastest $n \geq N$ convergence. The input traffic cyclical sequence and the given hit ratio are: *2k→6k→4k* and 68.3% (one standard deviation about the "f_{highest} *mean*" (Figure 1c)) respectively. The E-MACSC always maintains the prescribed hit ratio consistently for $a \leq 0.999$. For any α value larger than this threshold, the hit ratio drops steeply together with the memory consumption. The cause is the sudden loss of PR sensitivity because the emphasis is now on the past performance represented by α rather the current changes indicated by the *(1-α)* factor as shown in equation (3.6). Figure 5 shows the impact of different *flush limits* on the hit ratio with $a \leq 0.999$. The flush limit range that yields the highest hit ratio has shifted to $17 \leq F \leq 22$ from the best original $9 \leq F \leq 16$ range for the M_i prediction. This shift is

caused by the integrative/cumulative computation of the σ_i component in equation (3.6).

To confirm that the PE approach indeed needs more data samples to be collected on the fly to satisfy the $E\lambda = k\delta_{\bar{x}}$ criterion than the M^{3}RT mechanism, some of the above E-MACSC simulations are repeated with the MACSC under the same conditions. The average number of data samples needed by each tuner is listed in Table 1. Consistently, the MACSC tuner needs an average of 110 data samples to reach $n \geq N$ convergence, but the E-MACSC tuner needs only 18 on average. That is, the MACSC uses $(110/18) \approx 6.1$ times more samples on average and a computation overhead of 16 times, $(0.96ms/0.06ms) = 16$.

If the IAT delay and the speed of the platform are taken into account, then the average physical times to satisfy the $n \geq N$ criterion by the MACSC and E-MACSC tuners are 0.96 ms (milliseconds) and 0.06 ms respectively. The timing analysis is done with the *Intel's VTune Performance Analyzer* (VTune) and the speed of the platform being considered is 1.5 GHz (G for giga). If the average IAT is getting shorter (e.g. IAT→0), as for those simulations with pre-collected data traces where the data samples are readily usable without delay, the speedup can get up to 16 times. Yet, this is difficult to achieve in real-life applications because the data items have to be sampled one by one on the fly. It is only normal to have an IAT delay between two samples. Different simulations were carried out to verify if E-MACSC indeed is more accurate and has less oscillation than the MACSC in maintaining the given hit ratio. In the simulations the cache size under the E-MACSC control changes responsively and always settles down to satisfy the given 68.3% hit ratio requirement. For the MACSC response, however, the cache size is more oscillatory and has the tendency to stay at the higher values. There is much less chance for these problems to happen with the E-MACSC tuner because of its capability of smoother, faster, and more accurate dynamic buffer tuning.

5 CONCLUSION

The E-MACSC tuner is an enhanced successor of the previous MACSC model. It is created when the M^{3}RT μ-IEPM mechanism replaces the PE or *point-estimate* approach in the predecessor. The M^{3}RT

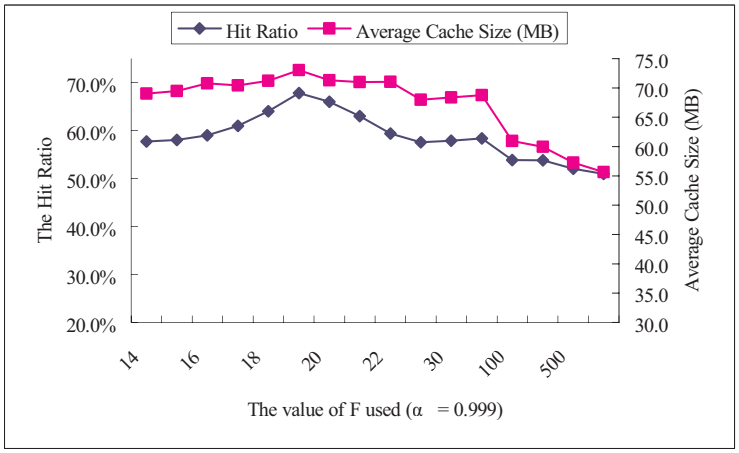

Figure 5: Correlation among hit ratio, cache size and F (α = 0.999).

Table 1: Average number of samples needed to compute the standard deviation (IAT = 0).

	Range of data samples to satisfy $n \geq N$	**Average number of data samples needed for the $n \geq N$ convergence**	**Physical time to satisfy $n \geq N$ on the platform that operates at 1.5GHz**
MACSC (PE)	60 ~ 150	110	0.96 ms
E-MACSC (M^{3}RT)	Any choice from the range: 16 ~ 20 (F value)	18 (refer to Figure 4)	0.06 ms

predicts the mean, namely, M_i of the data distribution being worked on. It differs from the PE computation by the following: a) it is faster, smoother and more accurate, b) it works with a fixed F (*flush limit*) number of data samples, and c) it is integrative with the M_{i-1} (predicted mean in the last cycle) feedback but the PE has no feedback loop. The stability of the M_i convergence, however, reduces the sensitivity of the popularity ratio that is required for producing accurate, responsive dynamic cache size tuning. With the aim to improve this sensitivity the integrative equation (3.6) for σ_i is proposed. For E-MACSC the calculation of the popularity ratio is based on equation (3.7) instead of using M_i directly. The preliminary simulation results confirm that the E-MACSC is far more efficacious in maintaining the given hit ratio than the MACSC approach. In addition the hit ratio by the E-MACSC tends to be higher than the given value. This leads to: shorter information retrieval RTT, less timeouts and thus retransmissions by the clients, more network backbone bandwidth freed for public sharing, and better system throughput in general. In contrast the hit ratio by the MACSC tuner oscillates and can be much lower than the given value. The analysis of the preliminary E-MACSC experience confirms that its efficacy depends on a few factors though, namely, the α value, the choice of the *flush limit*, and the average IAT of the data samples. Therefore, the planned activity for the next phase in the research is to study the impacts of these factors thoroughly. Different possible ways to neutralize the negative effect of some of these factors on dynamic cache tuning performance will be explored and scrutinized.

ACKNOWLEDGEMENTS

The authors thank the Hong Kong Polytechnic University and the Department of Computing for the research funding: H-JZ91

REFERENCES

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

Allan, K.Y. Wong, May, T.W. Ip and Richard, S. L. Wu, 2003. A Novel Dynamic Cache Size Adjustment Approach for better Retrieval Performance over the Internet, *Journal of Computer Communications*, 26(14)

Allan, K.Y. Wong, Tharam, S. Dillon, Wilfred, W.K. Lin and May, T.W. Ip, 2001. M2RT: A Tool Developed for Predicting the Mean Message Response Time for Internet Channels, *Computer Networks*, Vol. 36

Breslau, L., Cao, P., Li, F., Phillips, G. and Shenker, S., 1999. Web Caching and Zipf-like Distributions: Evidence and Implications, In *INFOCOM'99*, Vol. 1

Chis, J.A., 1992. *Introduction to Simulation and Modeling - GPSS/PC*, Prentice Hall

Cottrel, L., Zekauskas, M., Uijterwaal, H. and McGregor, T., 1999. Comparison of Some Internet Active End-to-End Performance Measurement Projects, http: //www.slac.stanford.edu/comp/net/wanmon/iepm-cf. html

Jin Shudong and Bestavros, A., 2000. Popularity-Aware Greedy Dual-Size Web Proxy Caching Algorithms, *Proc. of the Int'l Conf. on Distributed Computing Systems*

Paxson, V., Floyd, S., 1995. Wide area traffic: The Failure of Poisson Modeling, In *IEEE/ACM Transactions on Networking*, 3(3)

Richard, S.L Wu, May, T.W. Ip and Allan, K.Y. Wong, 2003. LDC-CM: A Novel Model for Dynamic Cache Size Adjustment, In *Proceeding of the International Conference on Internet Computing*, Vol. 2, Las Vegas USA

Stefan Podlipnig and Laszlo Böszörmenyi, 2003. A Survey of Web Cache Replacement Strategies, In *ACM Compputing Surveys*, Vol. 35, NO. 4, 374-398

Intel Vtune, Retrieved from http://developer.intel.com/ software/ products/vtune/

Wessels, D., 2001. Web Caching, *O'Reilly & Associates Inc.*

Zipf Curves and Website Popularity, Retrieved from http://www.useit.com/alertbox/zipf.html