

Modeling Frequently Accessed Wireless Data With Weak Consistency

YI-BING LIN AND YUNG-CHANG CHANG*

Department of Computer Science and Information Engineering

National Chiao Tung University

Hsinchu, 300 Taiwan

E-mail: liny@csie.nctu.edu.tw

**Far Eastone Telecommunications Co., Ltd.*

To reduce the response times of wireless data access in a mobile network, caches are utilized in wireless handheld devices. If the original data entry has been updated, the cached data in the handheld device becomes stale. Thus, a mechanism is required to predict when the cached copy will expire. This paper studies a weakly consistent data access mechanism that computes the time-to-live (TTL) interval to predict the expiration time. We propose an analytic model to investigate this TTL-based algorithm for frequently accessed data. The analytic model is validated against simulation experiments. Our study quantitatively indicates how the TTL-based algorithm reduces the wireless communication cost by increasing the probability of stale accesses. Depending on the requirements of the application, appropriate parameter values can be selected based on the guidelines provided in this paper.

Keywords: cache, mobile network, time-to-live, weak consistency, wireless data

1. INTRODUCTION

Modern mobile networks support wireless data applications. As shown in Fig. 1, a mobile customer may use a wireless handheld device (e.g., a wireless PDA) to access data services from an application server through a mobile network. An example of a wireless data architecture can be found in [11-13]. An application running on such a wireless handheld device may repeatedly access a data entry received from the application server. If the data entry is not sensitive to time, then the customer may access the data stored in the wireless handheld device instead of querying the application server, and the expense of the wireless transmission overhead will be reduced. If the data entry is sensitive to time, then the current data entry should be provided by the application server. An example is a stock quote. This application is *strongly consistent*, which means that it must guarantee that the data presented to the client is the same as that in the server. Such applications typically result in large amounts of wireless traffic and long response times, and may be charged by each data access. Some time-sensitive wireless applications can tolerate a certain degree of inaccuracy. For this type of application, we can set an expiration period t to predict when the data entry will be updated. During period t , the data entry in the handheld device can be used. When t expires, the next data access will result in a query to the mobile network. In this case, the application is *weakly consistent*, and the wireless handheld device may occasionally access stale data. A mecha-

Received July 13, 2001; revised November 23, 2001; accepted December 6, 2001.
Communicated by Chu-Sing Yang.

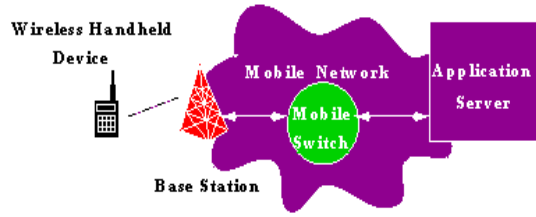


Fig. 1. Mobile network supporting wireless data applications.

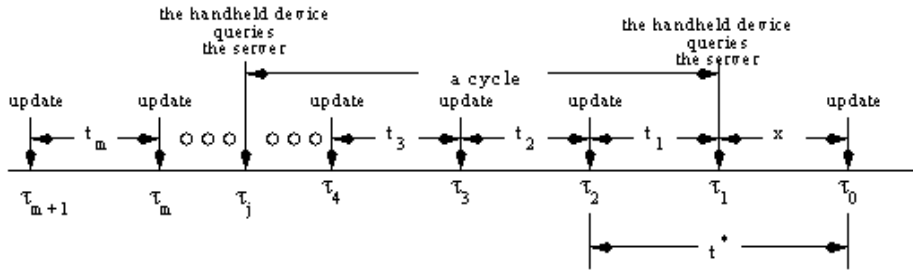


Fig. 2. Timing diagram I.

nism is required to predict when a data entry will expire. In a TTL-based consistency algorithm, a time-to-live (TTL) interval t is defined for a data entry stored in the wireless handheld device. Consider the timing diagram shown in Fig. 2. In this figure, the updates to the data entry occur at time τ_i (for $i \geq 0, i \neq 1$), where $\tau_i > \tau_{i+1}$. In Apache [1] and Squid [9], when the server is queried for a data entry at time τ_i , the TTL interval t for the data entry is set to

$$t = c_f (\tau_1 - \tau_2), \tag{1}$$

where τ_2 is the time for the previous update to the data entry (see Fig. 2) and c_f is a system defined fudge factor. In Jigsaw [3], the TTL interval is set to a fixed value of 24 hours. In LNC-R-W3-U [8], the TTL interval t is determined by the sliding window size m . That is,

$$t = \frac{\tau_1 - \tau_{m+1}}{m}, \tag{2}$$

where τ_m is the time of the m th most recent distinct update to the data entry. In Fig. 2, for a sliding window size $m = 3$,

$$t = \frac{\tau_1 - \tau_4}{3}.$$

Note that the TTL-based algorithm is typically implemented with a cache replacement scheme such as LRU (least recently used) or LFU (least frequently used) [2, 8] in a proxy cache for WWW accesses. Since the storage memory of a handheld device is

limited, the application may determine that no cache replacement algorithm is implemented for frequently accessed data (or they are likely to be replaced by infrequently accessed data). That is, when a wireless handheld device runs a particular application, some data used by this application may be considered as “frequently accessed,” and will always be kept in the handheld device until it expires. This is especially true for some location dependent services provided by mobile operators. A customer may also enable a data entry as “frequently accessed,” and the handheld device will not perform cache replacement for this data entry until the frequently accessed indication is disabled.

In this paper, we investigate the performance of the TTL-based algorithm for frequently accessed wireless data with weak consistency. We propose an analytic model for the TTL-based algorithm. Then we use several examples to analyze the algorithm. Our results provide guidelines for implementing the weakly consistent mechanism on wireless data services.

2. INPUT PARAMETERS AND OUTPUT MEASURES

Several trace-driven approaches have been proposed for Internet web performance. In a wireless environment, especially for location dependent services, no such traces exist. Therefore, we make the following assumptions to investigate frequently accessed wireless data. We assume that there is a sufficient number of accesses and a sufficient number of updates to a frequently accessed data entry. Thus the accesses can be approximated as a Poisson stream with rate λ , and the updates can be approximated as a Poisson stream with rate μ . With the above assumptions, we can conduct mean value analysis [6] on wireless data access / update algorithms and obtain useful primary results. We will consider the following input parameters:

- λ : the data access rate.
- μ : the data update rate.
- δ : the query / acknowledgement cost. Suppose that the cost for a query with data transmission is one unit. Then the cost for a query with acknowledgment (without data transmission) is $0 < \delta \leq 1$.

Define a *cycle* as the interval between two consecutive queries from a wireless handheld device to a server. In Fig. 2, $[\tau_j, \tau_1)$ is a cycle. An access at time τ_j results in a query to the server. During (τ_j, τ_1) , the handheld device returns the cached copy to all the accesses to the data entry. We derive the following primary output measures:

- The expected number $E[K^*]$ of *stale* accesses in a cycle: For a stale access, when the access occurs, the data entry is found in the cache, but the copy in the server has already been updated. For the cycle $[\tau_j, \tau_1)$ shown in Fig. 2, the cache returns a stale data entry for any access between τ_4 and τ_1 . We also derive $V[K^*]$, the variance of K^* .
- The expected number $E[K]$ of accesses in a cycle: This number includes the stale and the non-stale accesses in the cache plus the access resulting in a query from the handheld device to the server (for the cycle $[\tau_j, \tau_1)$ shown in Fig. 2, this query occurs at τ_j). Thus, $K \geq 1$ always holds. We also derive $V[K - 1]$, the variance of $K - 1$ (without considering the access that results in a query to the server).

- The probability β that when the handheld device queries the server, the data entry will be valid (i.e., the data entry has not been modified since the last query).

It is clear that the handheld device communicates with the server for every $E[K]$ access. Based on $E[K^*]$, $E[K]$ and β , two major output measures are derived:

- The *staleness ratio* p_s [8] is the probability that the handheld device will return a stale data entry for an access. That is,

$$p_s = \frac{E[K^*]}{E[K]}. \quad (3)$$

- The server query cost or wireless transmission cost C : When the handheld device queries the server, if the data entry is valid, then the server returns a positive acknowledgement. If the data has been modified, then the server returns the updated data entry to the handheld device. Note that on average, a query to the server occurs for every $E[K]$ accesses. Thus, if we normalize the cost (e.g., the wireless transmission delay) for a query with data transmission as one unit, then the server query cost can be expressed as

$$C = \frac{\delta\beta + (1-\beta)}{E[K]} = \frac{1-(1-\delta)\beta}{E[K]}. \quad (4)$$

As previously defined, δ is the cost of delivering a query-acknowledgement message if the cost of delivering the whole data entry is one unit.

3. MODELING THE TTL-BASED ALGORITHM

This section models the TTL-based algorithm. As mentioned in section 2, the accesses are a Poisson stream with rate λ , and the updates are a Poisson stream with rate μ . Thus, t^* , t_2 , t_3 , ..., t_m in Fig. 2 have the identical exponential distribution with mean $1/\mu$. Since the accesses are a Poisson stream, the access at time τ_1 is a random observer of the inter-update intervals. From the memoryless property of the exponential distribution [7], t_1 has the same distribution as t^* . From (2), the TTL value computed in LNC-R-W3-U [8] with a sliding window size of m is

$$t = \frac{t_1 + t_2 + \dots + t_m}{m}.$$

To generalize our derivation, the fudge factor c_f is also used to determine the TTL interval. That is,

$$t = c_f \left(\frac{t_1 + t_2 + \dots + t_m}{m} \right). \quad (5)$$

When $c_f = 1$, the TTL interval t computed in (5) is the one generated by LNC-R-W3-U [8]. When $m = 1$, t is generated by Apache [1] and Squid [9]. When m

$\rightarrow \infty$, t is a fixed value. Since t_i (where $1 \leq i \leq m$) are exponentially distributed, t has an Erlang density function $f_{TTL}(m, t)$ with mean $1/\mu_f$ [4], where

$$\mu_f = \frac{\mu}{c_f}. \tag{6}$$

That is,

$$f_{TTL}(m, t) = \left[\frac{(m\mu_f)^m t^{m-1}}{(m-1)!} \right] e^{-m\mu_f t}. \tag{7}$$

In Fig. 2, the next update after time τ_1 occurs at time τ_0 . Let $x = \tau_0 - \tau_1$. Then the cached copy of the data entry is valid during the period x . If $x > t$, then all accesses to the cached copy are valid before the TTL interval t expires. On the other hand, if $t > x$, then during the period $y = x - t$, all accesses to the cached entry are stale. The relationship between x , y , and t is shown in Fig. 3. The density function $f_Y(y)$ for y is derived as follows. Since $t = x + y$, and since x is exponentially distributed with mean $1/\mu$, from (7), we have

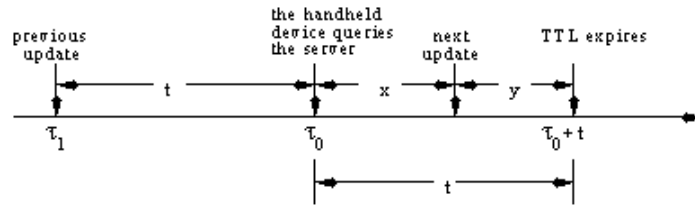


Fig. 3. Timing diagram II.

$$\begin{aligned} f_Y(y) &= \int_{x=0}^{\infty} \mu e^{-\mu x} f_{TTL}(m, x+y) dx \\ &= \int_{x=0}^{\infty} \mu e^{-\mu x} \left[\frac{(m\mu_f)^m}{(m-1)!} \right] \left[\sum_{j=0}^{m-1} \binom{m-1}{j} x^j y^{m-j-1} \right] e^{-m\mu_f(x+y)} dx \\ &= \left[\frac{\mu(m\mu_f)^m e^{-m\mu_f y}}{(m-1)!} \right] \left\{ \sum_{j=0}^{m-1} \binom{m-1}{j} \left[\frac{j! y^{m-j-1}}{(\mu + m\mu_f)^{j+1}} \right] \right\} \\ &= \sum_{j=0}^{m-1} \left[\frac{\mu(m\mu_f)^m y^{m-j-1}}{(m-j-1)! (\mu + m\mu_f)^{j+1}} \right] e^{-m\mu_f y}. \end{aligned} \tag{8}$$

Let $i = m - j - 1$. Then (8) can be re-written as

$$f_Y(y) = \sum_{i=0}^{m-1} \left[\frac{\mu(m\mu_f)^m y^i}{i! (\mu + m\mu_f)^{m-i}} \right] e^{-m\mu_f y}. \tag{9}$$

It is clear that $\Pr[t < x] = \Pr[y < 0]$ is the probability that all accesses to the cached copy will be valid during the TTL interval t . From (9),

$$\begin{aligned} \Pr[y < 0] &= 1 - \int_{y=0}^{\infty} f_Y(y) dy \\ &= \left(\frac{m}{m + c_f} \right)^m. \end{aligned} \tag{10}$$

When $m \rightarrow \infty$,

$$\lim_{m \rightarrow \infty} \Pr[y < 0] = \lim_{m \rightarrow \infty} \left(\frac{m}{m + c_f} \right)^m = e^{-c_f}$$

is the case when the TTL interval has a fixed value of $1/\mu_f = c_f/\mu$.

For the TTL algorithm with a sliding window size of m , we use the symbols $E_m[K^*]$, $V_m[K^*]$, $E_m[K]$, $V_m[K - 1]$, β_m , $p_{s,m}$, and C_m to represent the measures $E[K^*]$, $V[K^*]$, $E[K]$, $V[K - 1]$, β , p_s , and C (defined in the previous section). Thus, for a sliding window size of m , β_m is the probability that when the handheld device queries the server, the data entry will be valid. The probability β_m is derived as follows. Consider Fig. 4. Suppose that after the TTL interval expires at time $\tau_0 + t$, the next data access time occurs at $\tau_0 + t + t_1$ (which results in a query to the server). Also, after τ_0 , the next update occurs at $\tau_0 + x$. Then from the memoryless property of the exponential distribution, both t_1 and x will also be exponentially distributed with rates λ and μ , respectively. In other words,

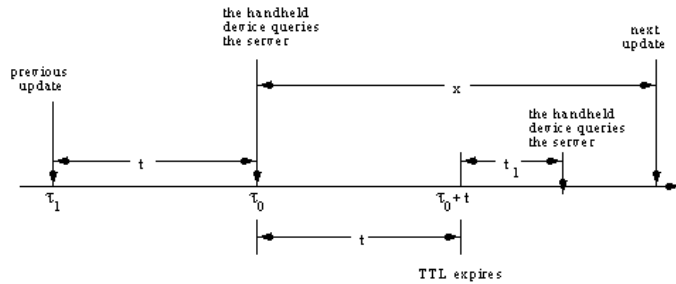


Fig. 4. Timing diagram III.

$$\begin{aligned} \beta_m &= \Pr[t < t + t_1 < x] \\ &= \int_{t=0}^{\infty} \int_{t_1=0}^{\infty} \int_{x=t+t_1}^{\infty} f_{TTL}(m,t) \lambda e^{-\lambda t_1} \mu e^{-\mu x} dx dt_1 dt \\ &= \left(\frac{\lambda}{\lambda + \mu} \right) \left(\frac{m}{m + c_f} \right)^m. \end{aligned} \tag{11}$$

From (10) and (11), we have

$$\beta_m = \Pr[y < 0] \left(\frac{\lambda}{\lambda + \mu} \right). \tag{12}$$

The intuition behind (12) is given as follows: β_m is the probability that the TTL interval will expire before the next update arrives (with probability $\Pr[y < 0]$), and that after it expires, the first event will be an access (with probability $\frac{\lambda}{\lambda + \mu}$).

Let $\Pr[N = n | T = y]$ be the probability that there will be n accesses to the data entry during period y . Since the data accesses are a Poisson stream with rate λ , we have [7]

$$\Pr[N = n | T = y] = \left[\frac{(\lambda y)^n}{n!} \right] e^{-\lambda y}. \tag{13}$$

Let $P_m(n)$ be the probability that there will be n stale accesses to the data entry during the TTL interval t with a sliding window size of m . Then from (9) and (13),

$$\begin{aligned} P_m(n) &= \int_{y=0}^{\infty} \Pr[N = n | T = y] f_Y(y) dy \\ &= \int_{y=0}^{\infty} \left[\frac{(\lambda y)^n}{n!} \right] e^{-\lambda y} \left\{ \sum_{i=0}^{m-1} \left[\frac{\mu(m\mu_f)^m y^i}{i!(\mu + m\mu_f)^{m-i}} \right] e^{-m\mu_f y} \right\} dy \\ &= \sum_{i=0}^{m-1} \left[\frac{\mu(m\mu_f)^m \lambda^n}{i!(\mu + m\mu_f)^{m-i} n!} \right] \left[\frac{(n+i)!}{(\lambda + m\mu_f)^{n+i+1}} \right] \\ &= \left(\frac{m\mu_f}{\mu + m\mu_f} \right)^m \left(\frac{\mu}{\lambda + m\mu_f} \right) \left[\sum_{i=0}^{m-1} \binom{n+i}{i} \left(\frac{\mu + m\mu_f}{\lambda + m\mu_f} \right)^i \right] \left(\frac{\lambda}{\lambda + m\mu_f} \right)^n \\ &= A_1 \left\{ \sum_{i=0}^{m-1} A_2^i \binom{n+i}{i} z^n \right\}, \end{aligned} \tag{14}$$

where

$$A_1 = \left(\frac{m\mu_f}{\mu + m\mu_f} \right)^m \left(\frac{\mu}{\lambda + m\mu_f} \right), A_2 = \frac{\mu + m\mu_f}{\lambda + m\mu_f}, \text{ and } z = \frac{\lambda}{\lambda + m\mu_f}. \tag{15}$$

Let $E_m[K^*]$ be the expected number of stale accesses during a TTL interval t (i.e., in a cycle) with a sliding window size of m . Then from (29) in Appendix A,

$$\begin{aligned} E_m[K^*] &= \sum_{n=1}^{\infty} n P_m(n) \\ &= \left(\frac{\lambda}{\mu} \right) \left[\left(\frac{m}{m + c_f} \right)^m - 1 + c_f \right]. \end{aligned} \tag{16}$$

For a sliding window size of m , let $E_m[K]$ be the number of accesses to the cached entry during the TTL interval t plus one (representing the query to the server at the beginning of the cycle). Then from (13) and (39) in Appendix C,

$$\begin{aligned} E_m[K] &= 1 + \sum_{n=1}^{\infty} n \left\{ \int_{t=0}^{\infty} \Pr[N = n \mid T = t] f_{TTL}(t) dt \right\} \\ &= \frac{c_f \lambda + \mu}{\mu} \end{aligned} \quad (17)$$

Equations (16) and (17) indicate that $E_m[K^*]$ and $E_m[K]$ are not affected by the distributions of y , t , and the inter-access intervals, but are only affected through their mean values (see (30) in Appendix A and (39) in Appendix C). The variance of K^* is derived in Appendix B as

$$\begin{aligned} V_m[K^*] &= \sum_{n=1}^{\infty} n^2 P_m(n) - (E_m[K^*])^2 \\ &= \left(\frac{\lambda}{\mu} \right)^2 \left\{ c_f^2 + \frac{c_f^2}{m} + \left(\frac{\mu}{\lambda} - 2 \right) \left[\left(\frac{m}{m+c_f} \right)^m - 1 + c_f \right] \right\} - (E_m[K^*])^2. \end{aligned} \quad (18)$$

The variance of $K - 1$ (by excluding the access that results in a query to the server) is derived in Appendix C as

$$\begin{aligned} V_m[K - 1] &= \sum_{n=1}^{\infty} n^2 \left\{ \int_{t=0}^{\infty} \Pr[N = n \mid T = t] f_{TTL}(t) dt \right\} - (E_m[K - 1])^2 \\ &= \left(\frac{1}{m} \right) \left(\frac{c_f \lambda}{\mu} \right)^2 + \frac{c_f \lambda}{\mu}. \end{aligned} \quad (19)$$

From (16) and (17), p_s for the TTL-based algorithm with a sliding window size of m is

$$p_{s,m} = \left(\frac{\lambda}{c_f \lambda + \mu} \right) \left[\left(\frac{m}{m+c_f} \right)^m - 1 + c_f \right]. \quad (20)$$

Let C_m be the server query cost (see (4)) for the TTL-based algorithm with a sliding window size of m . Then from (11) and (17),

$$C_m = \left(\frac{\mu}{c_f \lambda + \mu} \right) \left[1 - (1 - \delta) \left(\frac{\lambda}{\lambda + \mu} \right) \left(\frac{m}{m+c_f} \right)^m \right]. \quad (21)$$

The equations derived in this section have been validated against simulation experiments. In all the test cases, the errors were within 0.5% as shown in Table 1.

Table 1. A comparison of the analytic and simulation results for $p_{s,m}$ ($c_f=1$).

Access Rate λ	5μ	10μ	15μ	20μ
Simulation ($m = 1$)	0.416248	0.454732	0.468268	0.475432
Analytic ($m = 1$)	0.417	0.455	0.469	0.476
Simulation ($m = 2$)	0.369821	0.403122	0.417460	0.423964
Analytic ($m = 2$)	0.370	0.404	0.417	0.423
Simulation ($m = 3$)	0.350942	0.383360	0.395460	0.401911
Analytic ($m = 3$)	0.352	0.384	0.396	0.402
Simulation ($m = 4$)	0.340774	0.372161	0.384785	0.389087
Analytic ($m = 4$)	0.341	0.372	0.384	0.390

4. NUMERICAL EXAMPLES

This section provides numerical examples for investigating the TTL-based algorithm. We have generated test cases with large ranges of input parameters. This section presents the results obtained with selected input parameter values. Similar conclusions are drawn for other test cases not shown in this paper.

Effects of the access rate λ : Fig. 5 and 8 plot $p_{s,m}$, β_m and C_m for $\lambda = 5\mu, 10\mu, 15\mu$, and 20μ , respectively. These figures indicate that $p_{s,m}$ and β_m increase as λ increases, and that C_m decreases as λ increases.

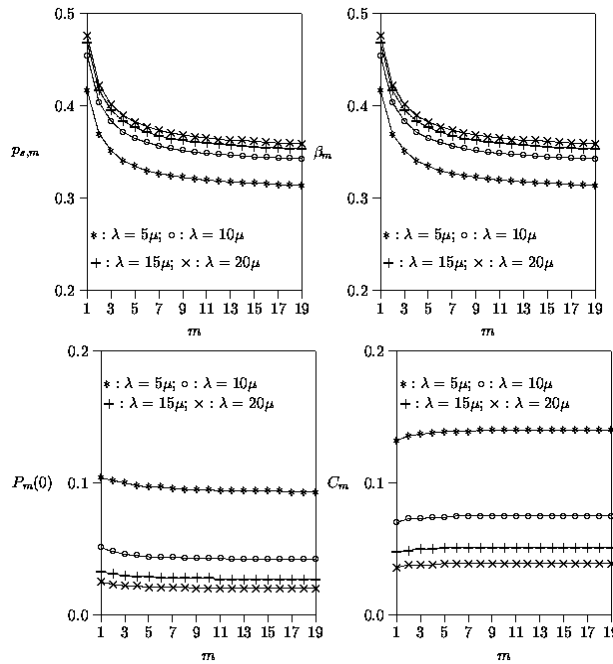


Fig. 5. Effects of the sliding window size on the TTL-based algorithm ($c_f=1$, and $\delta=0.5$).

The β_m curves can be explained using (12). Suppose that when the TTL interval expires, the next update has not yet arrived (with probability $\Pr[y < 0] = \left(\frac{m}{m+c_f}\right)^m$, which is independent of λ). Then for a large λ value, it is more likely that the next event will be a data access, and that β_m will increase as λ increases.

The impact of λ on $p_{s,m}$ is explained as follows. In a cycle, the first access is a query to the application server, which always returns the current data entry. If we exclude this access, then from (17), the expected number of remaining accesses in the cycle will be

$$E_m[K-1] = E_m[k] - 1 = \frac{c_f \lambda}{\mu}. \quad (22)$$

Note that for these accesses, the handheld device always returns the cached copy. Among these accesses, the portion of stale accesses is (cf. (16) and (22))

$$\frac{E_m[K^*]}{E_m[K-1]} = \left(\frac{1}{c_f}\right) \left[\left(\frac{m}{m+c_f}\right)^m - 1 + c_f \right],$$

which is independent of the access rate λ . Thus, λ only affects $p_{s,m}$ through the first access in the cycle, i.e., the query to the application server. Since the first access is always valid, the first-access effect reduces $p_{s,m}$. For a large λ (e.g., $\lambda > 10\mu$ in Fig. 8), there are more accesses in a cycle, and the first-access effect becomes insignificant.

The effect of λ on C_m is trivial. As λ increases, there are more accesses in a cycle, and the server query cost per cycle is reduced. A non-trivial observation is that C_m is more sensitive to the change of λ than $p_{s,m}$ (and β_m) is. Fig. 5 plots the $p_{s,m}$ and C_m curves for $c_f=1$ and $\delta=0.5$. For $m=3$, when λ increases from 5μ to 20μ , C_m decreases by almost 40%, while $p_{s,m}$ increases by 14%.

Effects of m on $p_{s,m}$, β_m , and C_m : Fig. 5 shows that $p_{s,m}$ decreases as m increases. This phenomenon is due to the fact that as m increases, the predicted TTL interval moves closer to the mean of the inter update interval, which results in a smaller $p_{s,m}$. This result is consistent with the observation of the trace-driven WWW access study in [8]. The $p_{s,m}$ improvement is most significant when m increases from 1 to 3. For $\lambda = 10\mu$, this improvement is about 16%. On the other hand, when m increases from 3 to 19, the improvement is less than 10%. In [8], $m=2$ or 3 is suggested.

Fig. 5 shows that, like $p_{s,m}$, β_m decreases as m increases. Intuitively, we expect that if stale accesses (i.e., a large $p_{s,m}$) are likely to occur in a cycle, then the handheld device is unlikely to have a valid cached entry when it queries the server (i.e., a small β_m is expected). Fig. 5 illustrates opposite result for the following reason. The variances of K^* and K are larger for smaller m (as will be shown in Fig. 7), which implies that for the same $p_{s,m}$ value, more cycles without any stale accesses or more cycles with large numbers of stale accesses will be observed for smaller m . Indeed, from (14), we have

$$P_m(0) = \left(\frac{m\mu_f}{m\mu_f + \mu} \right) \left[1 - \left(\frac{m\mu_f}{\lambda + m\mu_f} \right)^m \right],$$

which is a decreasing function of m as shown in Fig. 5. A larger $P_m(0)$ implies that there is a larger probability that the cached copy will be valid when the handheld device queries the server. Hence, β_m increases as m increases.

Since $E_m[K]$ is not affected by m (see (17)), the cost C_m is affected by m through β_m and δ (see (21)). From the above discussion for β_m , it follows that C_m increases as m increases. The effect of δ will be discussed next.

Effects of δ : It is clear that C_m decreases when δ decreases, as observed in Fig. 6. From (4), we have

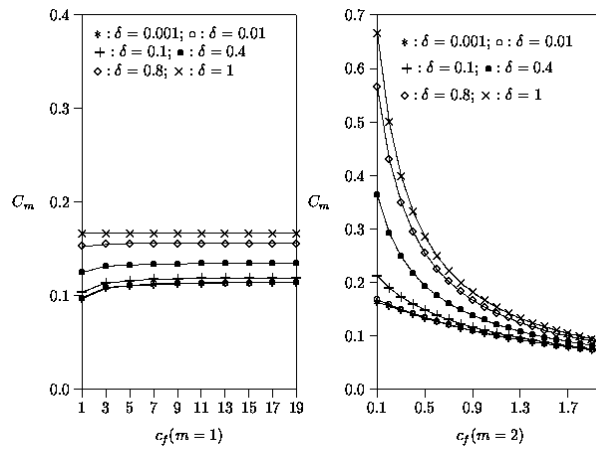


Fig. 6. Effects of δ on the TTL-based algorithm ($\lambda = 5\mu$).

$$C_m = \begin{cases} \frac{1}{E_m[K]}, & \text{for } \delta = 1 \\ \frac{1 - \beta_m}{E_m[K]}, & \text{for } \delta = 0. \end{cases}$$

That is, when $\delta = 1$, C_m is only affected by $E_m[K]$. When $\delta = 0$, C_m is also affected by $1 - \beta_m$. Thus, as δ decreases, β_m has more impact on C_m . For $\delta > 0.4$, when m increases from 1 to 3, C_m increases by less than 5%. Also, for $m > 3$, the effects of m on C_m can be ignored. This figure also indicates that when $\delta < 0.1$, C_m is insensitive to the change of δ .

Effects of m on $V_m[K^*]$ and $V_m[K - 1]$: Fig. 7 indicates that both $V_m[K^*]$ and $V_m[K - 1]$ decrease as m increases. Note that when $V_m[K^*]$ and $V_m[K - 1]$ become larger, this implies that there will be more cycles with large numbers of stale accesses and more cycles without any stale accesses. In other words, bursty stale accesses will be observed for

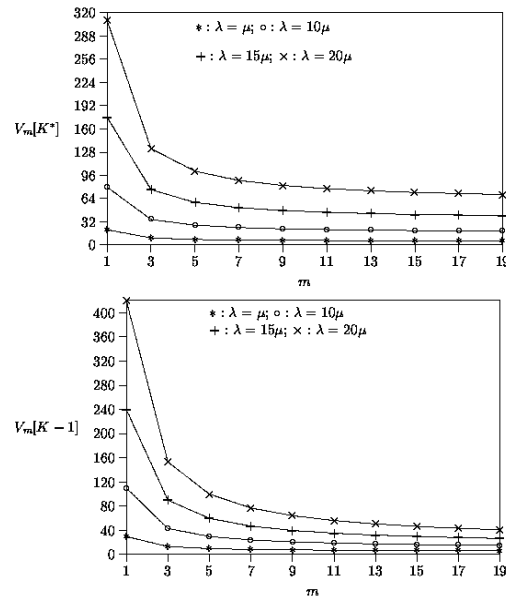


Fig. 7. Variances of K^* and $K - 1$ ($cf+1$).

small m . In some wireless data applications, regular stale access patterns may be desirable. Consider a positioning application as an example, where the location data are used to predict the position of a moving object. During a TTL interval, the cached location data entry may be periodically used together with a time function to predict the current position of the object. If there are many consecutive stale accesses to the data entry, the error caused by prediction may be large. In this example, small $V_m[K^*]$ and $V_m[K - 1]$ are desirable, and a larger m should be selected for the TTL-based algorithm.

Effects of c_f : Fig. 8 shows how the fudge factor c_f affects $p_{s,m}$, β_m and C_m , where $m = 2$ and $\delta = 0.5$. It is clear that for a large c_f , a large TTL interval be expected, and that there will be more accesses in a cycle (i.e., large $E_m[K^*]$ and $E_m[K]$). In this case, large $p_{s,m}$ and small β_m will be observed.

The cost C_m is affected by the conflicting factors β_m and $E_m[K]$, and the net effect shown in Fig. 8 is that C_m decreases as c_f increases. This result holds for all δ values (see also Fig. 6).

In Fig. 8, if we limit $p_{s,m}$ to be less than 0.1 by selecting $c_f = 0.2$, then the server query cost C_m will be less than 0.35. In other words, when 65% of the wireless transmission cost is saved, the customer pays the price of one stale access per 10 data accesses. The factor c_f can be dynamically adjusted to limit the wireless transmission cost C_m or the stale penalty $p_{s,m}$. As observed in both Figs. 5 and 8, C_m is more sensitive to λ than $p_{s,m}$ is, and it is probably better to adjust c_f based on C_m . In other words, (21) can be used as a heuristic to dynamically determine the c_f value.

Note that the $p_{s,m}$ value can be controlled by adjusting m or c_f . Fig. 9 plots the $C_m - p_{s,m}$ curves for various m values. For example, the “*” curve is for $m = 1$, where a point in the curve represents the $(C_m, p_{s,m})$ pair when a particular c_f value is chosen. We can ad-

just c_f so that the same $p_{s,m}$ value can be obtained for different sliding window sizes. In this case, Fig. 9 indicates that a larger m value results in a small C_m value. Thus, a larger sliding window size will yield better performance. This advantage become insignificant when $m > 3$.

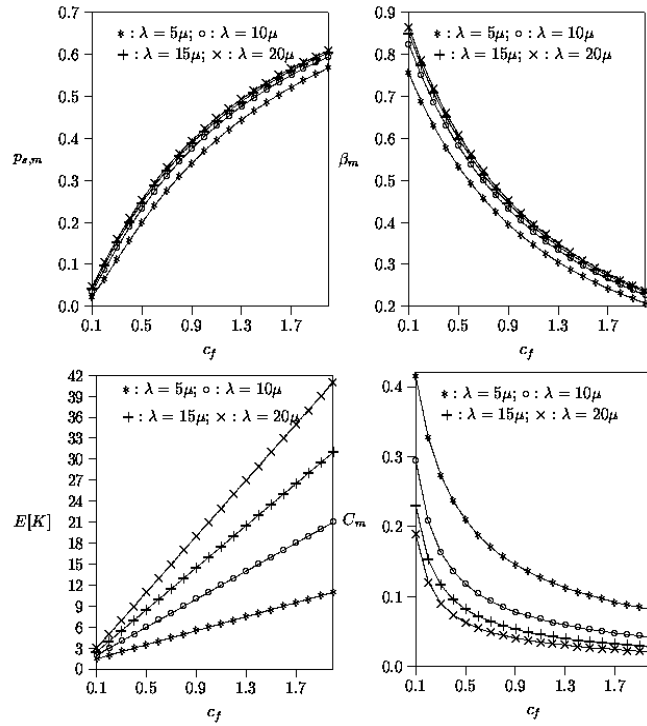


Fig. 8. Effects of the fudge factor on the TTL-based algorithm ($m = 2$).

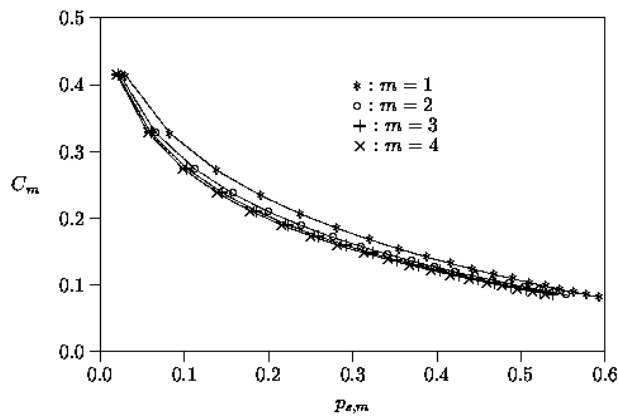


Fig. 9. The $C_m - p_{s,m}$ curves for various m values ($\lambda = 5\mu$, $\delta = 0.5$).

5. CONCLUSIONS

To reduce data access times and wireless traffic in a mobile network, frequently accessed data are typically cached in a wireless handheld device. The cached copy becomes stale if the data entry has been updated in the application server, and a mechanism is required to predict when the cached copy will expire. A popular approach is the TTL-based algorithm, which computes the time-to-live (TTL) interval to predict the expiration time. Several trace-driven approaches have been proposed to improve Internet web performance. In a wireless environment, especially for location dependent services, no such traces exist. Therefore, we make Poisson assumptions for data accesses and updates. Based on these assumptions, we have proposed an analytic model for investigating the TTL-based algorithm with various sliding window sizes for frequently accessed data. In our study, we have conducted mean value analysis of TTL-based algorithms. The analytic model has been validated against simulation experiments, giving useful primary results.

Our study has indicated that when the sliding window size m is increased from 1 to 3, the stale probability $p_{s,m}$ can be moderately reduced by slightly increasing the wireless communication cost (i.e., the application server query cost) C_m . For $m > 3$, both $p_{s,m}$ and C_m are insignificantly affected by the change of m . This result is consistent with the trace-driven WWW access study in [8]. Since the storage overhead for estimating the TTL interval increases as m increases (see [8] for a storage cost analysis), $m = 2$ or 3 is suggested.

We have also observed that by adjusting the fudge factor c_f , $p_{s,m}$ can be balanced against C_m . For the cases investigated in this paper, if we select $c_f = 0.2$, then 65% of the wireless communication overhead can be saved at the cost of one stale access per 10 data accesses. We have found that C_m is more sensitive to the change of c_f than $p_{s,m}$ is, which suggests that it is more appropriate to adjust c_f based on the measured C_m than on the measured $p_{s,m}$.

ACKNOWLEDGMENT

We would like to thank the three anonymous reviewers. Their valuable comments have significantly improved the quality of this paper. Lin's work was sponsored in part by the MOE Program of Excellence Research under contract 89-E-FA04-4, the TAHOE Network, Ericsson, InterVideo, FarEastone, the National Science Council under contract NSC 90-2213-E-009-156, and the Lee and MTI Center for Networking Research, NCTU.

A Derivation for $E_m[K^*]$

Let $E_m[K^*]$ be the expected number of stale accesses during a TTL interval t (i.e., a cycle) with a sliding window size of m . Then from (14)

$$\begin{aligned}
 E_m[K^*] &= \sum_{n=1}^{\infty} nP_m(n) \\
 &= A_1 \left\{ \sum_{i=0}^{m-1} A_2^i \left[\sum_{n=1}^{\infty} n \binom{n+i}{i} z^n \right] \right\}. \tag{23}
 \end{aligned}$$

Since

$$n \binom{n+i}{i} = (i+1) \binom{n+i}{n-1},$$

(23) can be re-written as

$$E_m[K^*] = A_1 \left\{ \sum_{i=0}^{m-1} (i+1) A_2^i \left[\sum_{n=1}^{\infty} \binom{n+i}{n-1} z^n \right] \right\}. \tag{24}$$

Let $k = n - 1$. Then (24) can be re-written as

$$E_m[K^*] = A_1 z \left\{ \sum_{i=0}^{m-1} (i+1) A_2^i \left[\sum_{k=0}^{\infty} \binom{k+i+1}{k} z^k \right] \right\}. \tag{25}$$

Define a special case of the generalized binomial series [5]:

$$\beta(z) = \left[\sum_{k=0}^{\infty} \binom{k+i}{i} z^k \right]^{-(i+1)}. \tag{26}$$

From [10], it can be shown that

$$\beta(z) = \frac{1}{1-z}. \tag{27}$$

Substituting (26) into (25), we have

$$\begin{aligned}
 E_m[K^*] &= [\beta(z)]^2 A_1 z \left\{ \sum_{i=0}^{m-1} (i+1) [A_2 \beta(z)]^i \right\} \\
 &= \left\{ \frac{A_1 z [\beta(z)]^2}{1 - A_2 \beta(z)} \right\} \left\{ \frac{1 - [A_2 \beta(z)]^m}{1 - A_2 \beta(z)} - m [A_2 \beta(z)]^m \right\}. \tag{28}
 \end{aligned}$$

From (15), (27) and (28) can be re-written as

$$E_m[K^*] = \left(\frac{\lambda}{\mu}\right) \left[\left(\frac{m}{m+c_f}\right)^m - 1 + c_f \right]. \quad (29)$$

Note that (29) can also be derived from intuition:

$$E_m[K^*] = \frac{E[y]}{E[\text{inter-access interval}]}. \quad (30)$$

From (9),

$$\begin{aligned} E[y] &= \int_{y=0}^{\infty} y f_Y(y) dy \\ &= \int_{y=0}^{\infty} y \left\{ \sum_{i=0}^{m-1} \left[\frac{\mu(m\mu_f)^m y^i}{i!(\mu+m\mu_f)^{m-i}} \right] e^{-m\mu_f y} \right\} dy \\ &= \left(\frac{1}{\mu}\right) \left[\left(\frac{m}{m+c_f}\right)^m - 1 + c_f \right]. \end{aligned} \quad (31)$$

Since the expected inter-access interval is $1/\lambda$, by substituting (31) into (30), we have the same result as in (29). Equation (30) implies that $E_m[K^*]$ is only affected by the expected values of the interval y and the access rate, and is independent of the distributions of y and the inter-access times.

Note that the tedious derivation from (23) to (29) will be used in Appendix B.

B Derivation for $V_m[K]$

From (14), the variance of K^* is

$$\begin{aligned} V_m[K^*] &= \sum_{n=1}^{\infty} n^2 P_m(n) - (E_m[K^*])^2 \\ &= A_1 \left\{ \sum_{i=0}^{m-1} A_2^i \left[\sum_{n=1}^{\infty} n^2 \binom{n+i}{i} z^n \right] \right\} - (E_m[K^*])^2. \end{aligned} \quad (32)$$

Since

$$\begin{aligned} \sum_{n=1}^{\infty} n^2 \binom{n+i}{i} z^n &= (i+1) \left[\sum_{n=1}^{\infty} n \binom{n+i}{n-1} z^n \right] \\ &= (i+1) \left[\sum_{n=2}^{\infty} (n-1) \binom{n+i}{n-1} z^n \right] + (i+1) \left[\sum_{n=1}^{\infty} \binom{n+i}{n-1} z^n \right] \end{aligned} \quad (33)$$

and

$$\begin{aligned} \sum_{n=2}^{\infty} (n-1) \binom{n+i}{n-1} z^n &= (i+2) \left[\sum_{n=2}^{\infty} \binom{n+i}{n-2} z^n \right] \\ &= (i+2) \left[\sum_{k=0}^{\infty} \binom{k+i+2}{k} z^{k+2} \right] \\ &= (i+2) z^2 [\beta(z)]^{i+3}, \end{aligned} \tag{34}$$

equation (35) can be derived from (34) by using (26). From (33) and (35), (32) can be re-written as

$$\begin{aligned} V_m[K^*] &= A_1 \left\{ \sum_{i=0}^{m-1} A_2^i (i+1)(i+2) z^2 [\beta(z)]^{i+3} \right\} + E_m[K^*] - (E_m[K^*])^2 \\ &= A_1 z^2 [\beta(z)]^3 \Theta_m(A_2 \beta(z)) + E_m[K^*] - (E_m[K^*])^2, \end{aligned} \tag{36}$$

where

$$\begin{aligned} \Theta_m(w) &= \sum_{i=0}^{m-1} (i+1)(i+2) w^i \\ &= \left(\frac{1}{w} \right) \left[\sum_{j=1}^m j(j+1) w^j \right] \\ &= \left(\frac{1}{w-1} \right) \left[m^2 w^m + \left(\frac{w-3}{w-1} \right) m w^m + \frac{2}{(w-1)^2} (w^m - 1) \right]. \end{aligned} \tag{37}$$

Substituting (37) and (15) into (36), we have

$$V_m[K^*] = \left(\frac{\lambda}{\mu} \right)^2 \left\{ c_f^2 + \frac{c_f^2}{m} + \left(\frac{\mu}{\lambda} - 2 \right) \left[\left(\frac{m}{m+c_f} \right)^m - 1 + c_f \right] \right\} - (E_m[K^*])^2.$$

C Derivations for $E_m[K^*]$ and $V_m[K - 1]$

For a sliding window size of m , let $E_m[K^*]$ be the number of accesses to the cached entry during the TTL interval t plus one (representing the query to the server at the beginning of the cycle). Then from (13),

$$E_m[K] = 1 + \sum_{n=1}^{\infty} n \left\{ \int_{t=0}^{\infty} \Pr[N = n \mid T = t] f_{TTL}(t) dt \right\}. \tag{38}$$

Apply the same reasoning to (30), $E_m[K]$ can also be expressed as

$$E_m[K] = 1 + \frac{E[t]}{E[\text{inter-access interval}]} = \frac{c_f \lambda + \mu}{\mu}. \tag{39}$$

With tedious derivation, it can be shown that (38) yields the same result as (39). The variance of $K - 1$ is derived as follows (we exclude the one access that results in a

query to the server). Similar to (38),

$$\begin{aligned} V_m[k-1] &= \sum_{n=1}^{\infty} n^2 \left\{ \int_{t=0}^{\infty} \Pr[N=n | T=t] f_{TTL}(t) dt \right\} - (E_m[k-1])^2 \\ &= \sum_{n=1}^{\infty} n^2 \left\{ \left[\frac{(\lambda t)^n}{n!} \right] e^{-\lambda t} \left[\frac{(m\mu_f)^m t^{m-1}}{(m-1)!} \right] e^{-m\mu_f t} dt \right\} - (E_m[k-1])^2 \\ &= \left(\frac{m\mu_f}{\lambda + m\mu_f} \right)^m \left[\sum_{n=1}^{\infty} n^2 \left(\frac{\lambda}{\lambda + m\mu_f} \right)^n \binom{n+m-1}{n} \right] - (E_m[k-1])^2. \end{aligned}$$

From (15),

$$z = \frac{\lambda}{\lambda + m\mu_f}$$

and

$$\begin{aligned} V_m[K-1] &= (1-z)^m \left[\sum_{n=1}^{\infty} n^2 \binom{n+m-1}{n} z^n \right] - (E_m[K-1])^2 \\ &= m(1-z)^m \left[\sum_{n=2}^{\infty} (n-1) \binom{n+m-1}{n-1} z^n \right] + E_m[K-1] - (E_m[K-1])^2 \end{aligned} \quad (40)$$

$$= m(m+1)(1-z)^m \left[\sum_{k=0}^{\infty} \binom{k+m+1}{k} z^{k+2} \right] + E_m[K-1] - (E_m[K-1])^2 \quad (41)$$

$$= m(m+1)(1-z)^m z^2 \left(\frac{1}{1-z} \right)^{m+2} + E_m[K-1] - (E_m[K-1])^2 \quad (42)$$

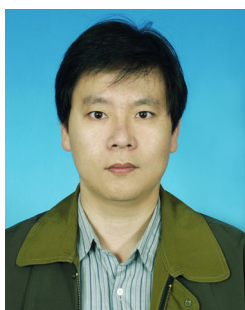
$$= \left(\frac{1}{m} \right) \left(\frac{c_f \lambda}{\mu} \right)^2 + \frac{c_f \lambda}{\mu}. \quad (43)$$

Note that (41) is derived form (40) using (26), and that (43) is derived form (42) using (15), (27) and (39).

REFERENCES

1. Apache 1.3. HTTP Server Document; <http://www.apache.org>, 2000.
2. P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," in *Proceedings of Usenix Symposium on Internet Technologies and Systems*, 1997.
3. Jigsaw 2.0.5. HTTP Server Document; <http://www.w3c.org/Jigsaw>, 2000.
4. N. L. Johnson, *Continuous Univariate Distributions-I*. John Wiley & Sons, 1970.
5. I. H. Lambert, "Observationes variae in Mathesin puram," *Acta Helvetica*, Vol. 3, 1758, pp. 128-168.

6. E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, *Quantitative System Performance*, Prentice Hall, 1984.
7. S. M. Ross, *Stochastic Processes*, John Wiley & Sons, 1996.
8. J. Shim, P. Scheuermann, and R. Vingralek, "Proxy cache algorithms: Design, implementation, and performance," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 11, 2000, pp. 549-562.
9. Squid 2.3. *Internet Object Cache Document*; <http://squid.nlanr.net/Squid>, 2000.
10. A. Tucker, *Applied Combinatorics*, John Wiley & Sons, 1984.
11. WAP Forum "Wireless application protocol architecture specification," Technical report, WAP Forum, 1998.
12. WAP Forum "Wireless application protocol white paper," Technical report, WAP Forum, 1999.
13. WAP Forum "Wireless application protocol over GSM unstructured supplementary service data," Technical report, WAP Forum, 1999.



Yi-Bing Lin (林一平) received his BSEE degree from National Cheng Kung University in 1983, and his Ph.D. degree in Computer Science from the University of Washington in 1990. From 1990 to 1995, he was with the Applied Research Area at Bell Communications Research (Bellcore), Morristown, NJ. In 1995, he was appointed as a professor of Department of Computer Science and Information Engineering (CSIE), National Chiao Tung University (NCTU). In 1996, he was appointed as Deputy Director of Microelectronics and Information Systems Research Center, NCTU. During 1997-1999, he was elected as Chairman of CSIE, NCTU. His current research interests include design and analysis of personal communications services network, mobile computing, distributed simulation, and performance modeling.

Dr. Lin is an associate editor of *IEEE Network*, an editor of *IEEE J-SAC: Wireless Series*, an editor of *IEEE Personal Communications Magazine*, an editor of *Computer Networks*, an area editor of *ACM Mobile Computing and Communication Review*, a columnist of *ACM Simulation Digest*, an editor of *International Journal of Communications Systems*, an editor of *ACM / Baltzer Wireless Networks*, an editor of *Computer Simulation Modeling and Analysis*, an editor of *Journal of Information Science and Engineering*, Program Chair for the 8th Workshop on Distributed and Parallel Simulation, General Chair for the 9th Workshop on Distributed and Parallel Simulation. Program Chair for the 2nd International Mobile Computing Conference, Guest Editor for the ACM/Baltzer MONET special issue on Personal Communications, a Guest Editor for *IEEE Transactions on Computers* special issue on Mobile Computing, and a Guest Editor for *IEEE Communications Magazine* special issue on Active, Programmable, and Mobile Code Networking. Lin is the author of the book *Wireless and Mobile Network Architecture* (co-author with Imrich Chlamtac; published by John Wiley & Sons). Lin received 1998 and 2000 Outstanding Research Awards from National Science Council, ROC, and 1998 Outstanding Youth Electrical Engineer Award from CIEE, ROC. Lin is an Adjunct Research Fellow of Academia Sinica.



Yung-Chang Chang (張永昌) is the Division Head of Information Technology for Far EasTone Telecommunications Co., Ltd. He joined FET at January 1st, 1999. Dr. Chang received his B.S. in Transportation & Communication and M.S. degree in Civil Engineering from National Cheng Kung University and his Ph.D from Chinese Culture University in International Business Management. The subject of his dissertation is the relationship between Corporate internet and e-commerce technique adaptation. Except having published a few articles on different releases regarding the subjects of transportation & communications management and financial management, he also serves as a lecturer, Associate Professor in the International Business Management and Transportation & Communication of Chinese Culture University and National Cheng Kung University respectively.

Presently Dr. Y. C. Chang is in charge of Information Technology related operations and development issues including Billing and Customer Care Systems, Management Information Systems, IT Infrastructure, IT Planning & Quality and IT Architecture etc. Being evolved in a violent competitive telecommunications environment, his major responsibilities are to fulfill the internal and external customers requirement in relation to information technology.