植基於比例式延遲差異的封包排程演算法:最長等待時間優先

黃一泓 [1] 蔡安喬 [2] 朱延平 [3]
修平技術學院資訊網路技術系 [1] 中興大學資訊科學研究所 [2] 東海大學資訊工程與科學系 [3]
ehhwang@mail.hit.edu.tw, s9256020@cs.nchu.edu.tw, ypchu@thu.edu.tw

摘要

本文提出一個植基於比例式延遲差異(proportional delay differentiation，PDD)模組的封包排程演算法:最長等待時間優先(Longest Waiting Time First，LWTF)。本演算法的時間複雜度為 $O(N)$，優於 Advanced Waiting Time Priority (AWTP) [17] 的 $O(N^2)$。經由模擬實驗的結果，LWTF 不管在一般流量(moderate traffic load)以及高流量(high traffic load)的情況下，都可以獲得比 AWTP 更確實的延遲比例分配，讓不同等級之間產生相對延遲的差異來達成相對性差異式服務的需求。

關鍵詞：封包排程演算法、差異式服務、相對性差異的服務

Abstract

In this paper, we proposed a new packet scheduling algorithm for proportional delay differentiation (PDD) model called Longest Waiting Time First (LWTF). The time complexity of LWTF is $O(N)$, that is better than the time complexity of Advanced Waiting Time Priority (AWTP) algorithms [17], $O(N^2)$. Simulation results show that when the link utilization is moderate (60%~90%), LWTF obtains more accurate delay proportion than the Waiting Time Priority (WTP) [5] and AWTP scheduling algorithms.

Keywords: Packet Scheduling, DiffServ, relative service differentiation

1. 前言

由於網路技術的進步以及應用程式的需求，使用者對於網路的要求越來越高，傳統上以盡力而為(Best-Effort Service) [3,6]的方式來提供服務已無法滿足。為了達成不同使用者對於網路傳輸的需要，就必須為使用者設定不同的服務等級。因此，最近幾年已經有很多的方法被提出來解決這個問題以達成服務品質(Quality of Service，QoS)保證的需求[2,15,16]。

為此網際網路工程工作特別小組(Internet Engineering Task Force: IETF)提出了二種網路架構來提供 QoS，一種是整合式服務(Integrated Services) [1,8,12]，而另外一種是差異式服務(Differentiated Services， DiffServ)。在 DiffServ [7,9,11-14]服務中有二個方向，一個是絕對式服務差異(absolute service differentiation)，另一個是相對式服務差異(relative service differentiation)。在絕對式服務差異中，必須提供使用者絕對且穩定的效能保證。在相對式服務差異中，可以保證服務等級較高的使用者一定會比服務等級較低的使用者有較好的效能。而相對式服務差異又可分成四種模組：分別是嚴格的優先權、價錢的差異、能力的差異和比例式延遲差異(PDD)。其中比例式延遲差異模組，可以利用某些參數的控制來提供相對式服務差異。架構在比例式延遲差異模組[4,5]下的封包排程演算法，可以由計算封包的優先權來決定傳送順序使得不同等級之間的平均等待時間達到一定的比值，進而做到滿足相對式服務差異的需求[4]，這便是本篇論文所探討的主題。

接下來在第二節中將描述 PDD 模組的相關概念，以及探討兩個主要能達成 PDD 的封包排程演算法。第三節則提出我們的封包排程演算法 LWTF，第四節則藉著模擬實驗的展示來驗證 LWTF 如何達成比 WTP 及 AWTP 更優異的效能，最後提出我們的結論。

2. 相關文獻探討

Constantinos Dovrolis 等[5] 提出在 PDD 模組中的假設條件：

(1). 共有 N 個等級，每個等級都有一個佇列。封包排程演算法則用來決定要服務哪一個等級的第一個(head of line)封包。
(2). 每個等級的佇列(queue)都是採用先進先出(First-Come-First-Served)。
(3). 所有等級的封包進到佇列的速率總和會小於封包被處理的速度，也就是系統的使用率必須小於 1。
(4). 每個佇列的長度為無限的，也就是不會有封包被丟棄的情形產生。

PDD 模組希望不同等級之間的平均等待時間能達到一定的比值。假設 $\bar{d}_i$ 為第 i 個等級的平均等待時間，δ_i 為第 i 個等級的延遲差異參數(Delay Differentiation Parameter，DPP)。對任一對等級 i 及 j 都必須滿足下列的條件：

$$\frac{\bar{d}_i}{\bar{d}_j} = \frac{\delta_i}{\delta_j} \left(1 \le i, j \le N\right). \quad (1)$$

符號越低，代表等級得到的服務越好。也就是延遲(delay)時間越低。並且設定等級 1 的 DDP，$\delta_1 = 1$。然後根據不同的系統應用需求，設定其他等級的 DDP，來產生各個等級間延遲比例的差異。封包排程演算法藉著控制封包的傳輸順序，控制每個等級之間的平均等待時間來達到等級與等級間的延遲比例滿足 DDP 比值的需求。因此，接下來我們將介紹兩個主要的封包排程演算法。

2.1 Waiting Time Priority

當封包抵達時先記錄此封包的抵達時間，接著假設系統在時間 t 時空閒，此時要決定傳送哪一個封包。系統便計算每個等級 i 中，第一個封包的等待時間為 $W_i(t)$。然後算出等級 i 的優先權 $P_i(t)$：

$$P_i(t) = \frac{W_i(t)}{\delta_i}, \tag{2}$$

$$j = \arg \underset{i \in B_i(t)}{Max} P_i(t). \tag{3}$$

由(2)式算出在時間 t 時，每個有封包在佇列之等級 i 的優先權 $P_i(t)$。再由(3)式，取優先權最大的等級 j 的第一個封包來傳送，其中 $B_i(t)$ 是在時間 t 時有封包在佇列之等級形成的集合。每次在送封包之前都會再計算(2)和(3)一次。

2.2 Advanced Waiting Time Priority

WTP 由於只考慮第一個封包的等待時間，並沒有考慮到封包的傳送時間，由於每個封包的大小不同，因此封包的傳送時間也會不同，在封包傳送時自然會影響其它封包的等待時間，所以 Yuan-Cheng Lai 和 Wei-His Li 在[17]提出一個針對 WTP 改進的方法為 Advanced Waiting Time Priority (AWTP)。

WTP 原本由(2)算出每個等級的優先權，而 AWTP 則是先考慮每個等級傳送後的情況。

$$\tilde{V}_j^i(t) = \frac{W_j(t) + X_i}{\delta_j}, \quad \left(X_i = 0,\ \text{if } j = i;\ X_i = T_i(t),\ \text{if } j \neq i\right) \tag{4}$$

先假設現在要傳送第 i 個等級的第一個封包，則其它等級 j 優先權 $\tilde{V}_j^i(t)$ 分別由(4)算出，其中 $T_i(t)$ 為第 i 個等級第一個封包傳送所需的時間。

$$MP_i(t) = \underset{1 \le j \le N}{Max}\left\{\tilde{V}_j^i(t)\right\}, \tag{5}$$

若是全部有 N 個等級，先考慮傳送第 i 個等級，由(5)可以算出假設等級 i 傳送後，N 個等級的優先權。

$$C(t) = \arg \min_{1 \le i \le N}\left\{MP_i(t)\right\}. \tag{6}$$

接下來每個等級都會重複式子(5)N 次，也就是會考慮 N 個等級先傳送之後的情形，所以會得到 $MP_1, MP_2, \cdots MP_N$，再由(6)式找出 N 個等級中最小的，所以 $C(t)$ 就是所要傳送的等級。

3. Longest Waiting Time First

AWTP 考慮封包傳送時間後雖然可以使效果改善，但是計算方式太複雜。因此本篇提出一個較 AWTP 簡單的而且能達到與 AWTP 類似效能的方法。主要設計的概念為，先針對每個等級假設該等級為最後一個要傳的等級，然後經由計算，選取可能等待時間最久的等級做傳送。因為該等級若是越晚傳可能就會造成更差的影響，也就是說由最長等待時間的等級來做傳送封包的動作。

3.1 演算法

Longest Waiting Time First (LWTF)為考慮封包等待時間最長的等級做傳送。若有 N 個等級，並假設 $W_i(t)$ 為第 i 個等級中第一個封包的實際等待時間，另外 $T_j(t)$ 為第 j 個等級中第一個封包所需處理時間。先計算第 i 個等級 $(1 \le i \le N)$ 若由其他等級的第一個封包先傳送，最後才由第 i 個等級傳送時，則第 i 個等級的第一個封包所需等待的時間，

$$W_i(t) + \sum_{j=1, j \neq i}^{N} T_j(t), \tag{7}$$

然後再由式子(7)做正規化計算，求出第 i 個等級的優先權 $P_i(t)$，

$$P_i(t) = \frac{W_i(t) + \sum_{j=1, j \neq i}^{N} T_j(t)}{\delta_i} \quad (1 \le i \le N), \tag{8}$$

再由(8)將每個等級的優先權 $P_i(t)$ 做比較，

$$j = \arg \max_{1 \le i \le n}\left\{P_i\right\}. \tag{9}$$

由(9)從取出最大值的等級的第一個封包做傳送。

3.2 LWTF 範例

接著舉一個 LWTF 的例子，假設現在有三個等級，其中 $W_1(t) = 0.6$, $W_2(t) = 0.8$, $W_3(t) = 1$ 分別為三個等級中第一個封包的等待時間，$T_1 = 0.6$, $T_2 = 0.05$, $T_3 = 0.5$ 分別為三個等級中第一個封包所需傳送時間，而三個等級的 DDP 分別為 $\delta_1 = 1$、$\delta_2 = 2$、$\delta_3 = 4$。由(8)可以得到 $P_1(t) = 1.15$，$P_2(t) = 0.95$，$P_3(t) = 0.4125$，然後由(9)從三個等級

中取優先權最大的等級出來，因此這個例子是由等級 1 來做傳送封包。

3.3 時間複雜度比較

WTP 考慮每個等級的等待時間，由每個等級的等待時間去計算出優先權，所以其複雜度為 $O(N)$。而 AWTP 是先考慮每個封包傳送後所有等級的優先權的變化，並對所有的等級都做一次，所以其複雜度為 $O(N^2)$。而 LWTF 則是先計算每個等級第一個封包的等待時間和所需處理時間，然後計算出每個等級的優先權，因此 LWTF 的複雜度也是 $O(N)$。由於 LWTF 考慮到在封包有不同大小時，有不同的處理時間，因此可以達到與 AWTP 相似的效果，並且時間複雜度也比 AWTP 好。

4. 實驗結果與分析

在下列的實驗中，使用 WTP 和 AWTP 來和 LWTF 做比較，我們使用 NS2-2.27(Network Simulator 2)的版本。實驗採用三個等級，而 DDP 分別為 $\delta_1 = 1$、$\delta_2 = 2$、$\delta_3 = 4$，封包抵達速率(packet arrival rate)則是用柏拉圖(Pareto)分布來產生，其中 shape 參數設定為 1.9，封包大小每個等級皆為 40% 的封包是 40 bytes、50%的封包是 550 bytes 及 10% 的封包是 1500 bytes，也就是說平均的封包大小是 441 bytes。另外總頻寬設定為 2646 bytes/sec 且緩衝區(Buffer)的長度是無限制，也就是不會有封包丟棄的情形產生，每個等級的封包抵達速率(Packet Arrival Rate)除了 4.3 節與 4.4 節的設定不同外，在其他的實驗中，每個等級的封包抵達速率都相同，實驗則是跑 30000 實驗秒。

4.1 等待時間的比值

圖 1 是在不同使用率的情況下，計算等級 3 和 等級 1 等待時間的比值 R_{31} 與等級 2 和 等級 1 等待時間的比值 R_{21}，

$$R_{ij} = \frac{\overline{d}_i}{\overline{d}_j}. \tag{10}$$

其中 $\overline{d}_i$ 為第 i 個等級的平均等待時間，由 DDP 可以得到 R_{31} 應該為 $\frac{\delta_3}{\delta_1} = 4$，而 R_{21} 應該為 $\frac{\delta_2}{\delta_1} = 2$。

在使用率達 95%以上時，WTP、AWTP 及 LWTF 三個方法效果是很接近的。在使用率 85% 和 90% 時 LWTF 比 AWTP 差一點，但比 WTP 好。在使用率為 60%~80%時 LWTF 會比 WTP 和 AWTP 還要好，而且 LWTF 傾向於提供優先權較高的等級較優異的服務。

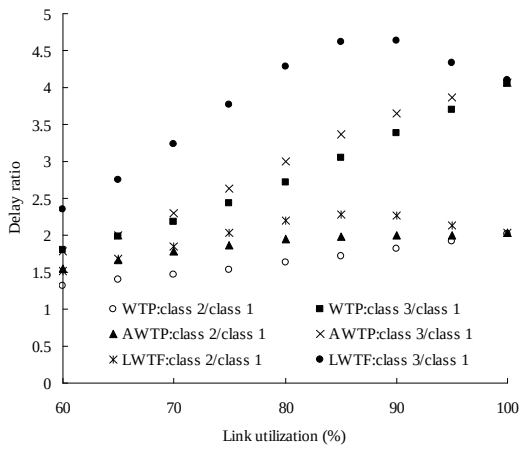

圖 1 WTP、AWTP 及 LWTF 的等待時間比值

4.2 等待時間的改善

接著觀察等待時間的改善程度， LWTF 和 WTP 等待時間改善程度為 $\frac{W-L}{W}$，其中 W 為 WTP 的等待時間，L 為 LWTF 的等待時間，而 LWTF 和 AWTP 等待時間改善程度為 $\frac{A-L}{A}$ 其中 A 為 AWTP 的等待時間，L 為 LWTF 的等待時間。

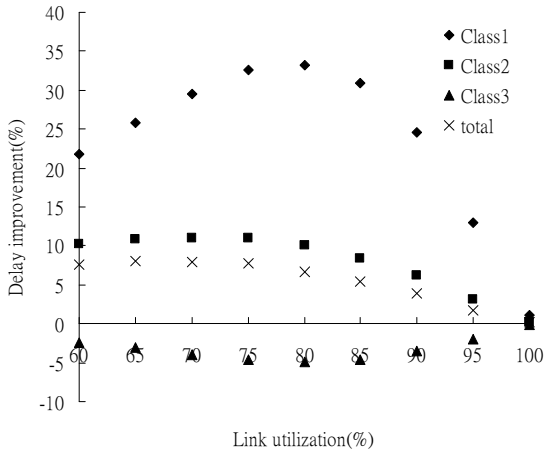

圖 2 WTP 和 LWTF 在等待時間的改善

圖 2 是由 LWTF 和 WTP 二個方法分別對三個等級的等待時間改善程度的比較，以及由三個等級等待時間的總和來看等待時間改善程度的比較。圖 2 中可看出 LWTF 在總共的等待時間會比 WTP 好，而在使用率越高的部份，二個方法也會越接近，從圖 2 中可看出 LWTF 在等級 3 的等待時間是較長的，且等級 1 和等級 2 的等待時間就比較短，因為 LWTF 的 R_{31} 較接近 4，當等級 3 的等待時間增加，R_{31} 也會增加。

圖 3 是由 LWTF 和 AWTP 二個方法分別對三個等級的等待時間改善程度的比較，以及由三個等級等待時間的總和來看等待時間改善程度的比較。而在圖 3 中可以看出，當使用率較低時，LWTF 在等級 1 與等級 2 的表現都優於 AWTP，AWTP 在等級三的表現則優於 LWTF，從這裡又再一次驗證，LWTF 比 AWTP 較傾向於提供較高的優先權給高等級的封包。而在使用率越高時以上 LWTF 和 AWTP 是差不多的。

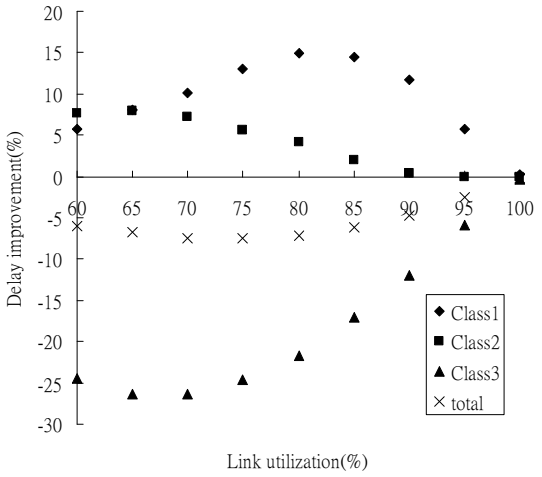

圖 3 AWTP 和 LWTF 在等待時間的改善

4.3 相異等級間不同封包抵達速率對等待時間比值的影響

本節觀察，當每個等級的封包抵達速率(Packet Arrival Rate)改變時，對等待時間的比值所造成的影響，各實驗環境是使用率為 70%。

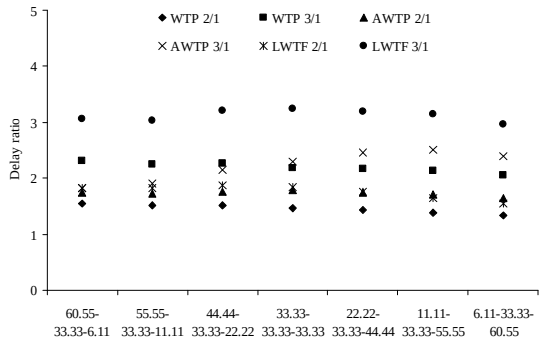

圖 4 在不同的封包抵達速率分布下等待時間的比值

圖 4 可以看得出來，LWTF 在等待時間的比值 R_{21} 和 R_{31} 和 WTP 與 AWTP 比會比較接近 2 和 4。另外 LWTF 的方法 R_{21} 介於 1.55 到 1.87 之間，R_{31} 介於 2.96 到 3.23 之間。而 AWTP 的方法 R_{21} 介於 1.65 到 1.78 之間，R_{31} 介於 1.82 到 2.51 之間。而 WTP 的方法 R_{21} 介於 1.34 到 1.54 之間，R_{31} 介於 2.06 到 2.31 之間。可以看出 LWTF 和 WTP 及 AWTP 相比是比較不會受到不同等級間封包抵達速率的影響。

4.4 相異等級間不同封包抵達速率與等待時間的改善

本節觀察當每個等級的封包抵達速率(Packet Arrival Rate)改變時，對等待時間的改善所造成的影響。各實驗的環境都是使用率為 70%，且針對三個等級分別做統計，另外也有做三個等級的總和做統計。

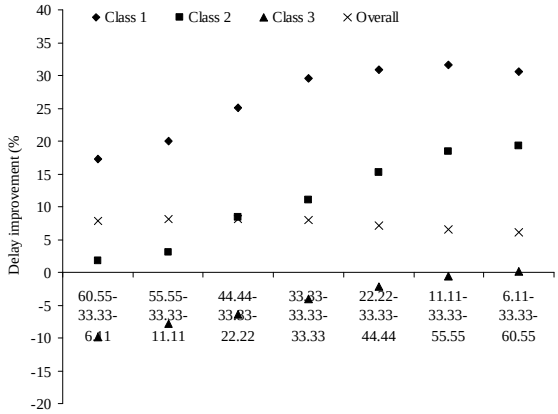

圖 5 在不同的封包抵達速率下 WTP 和 LWTF 等待時間的改善

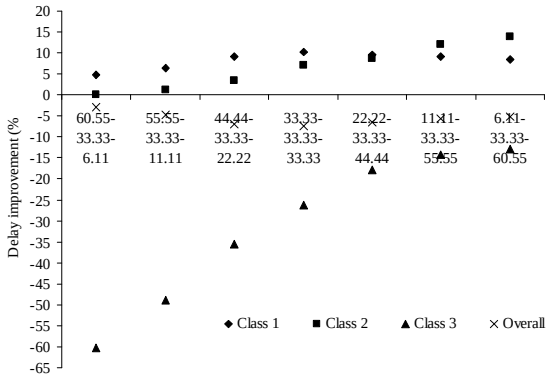

圖 6 在不同的封包抵達速率分布下 AWTP 和 LWTF 等待時間的改善

計算方式是和 4.2 節相同。從圖 5 中可以看出來在等級 1 和等級 2 的等待時間都有明顯的改善，而等級 3 的等待時間則差一點，不過在總合來看有不錯的改善。圖 6 所觀察到的結果與 4.2 節的結果是類似的。

由圖 5 和圖 6 可以知道 LWTF 在等待時間上是介於 WTP 和 AWTP 之間，但是在等級 1 和等級 2 的等待時間比 AWTP 來的好，而等級 3 比 AWTP

差，但是也由於等級 3 的等待時間變長，才會使得等待時間的比值會比 AWTP 好，在每個等級有不同封包抵達速率所得到的結果也是和 4.2 節一致。

4.5 測量間距與平均等待時間的比值

本節觀察，測量的時間間距，對各方法的影響。實驗分成每送十個封包統計一次平均的等待時間比值，以及每送一百個封包統計一次平均的等待時間比值和每送一千個封包統計一次平均的等待時間比值。模擬環境為三個等級，使用率 90%，而 DDP 分別為 $\delta_1 = 1$ 、$\delta_2 = 2$ 、$\delta_3 = 4$

而這個實驗中平均等待時間的比值的計算方式為 $\frac{R_{32} + R_{21}}{2}$ ，由 DDP $\delta_1 = 1$ 、$\delta_2 = 2$ 、$\delta_3 = 4$ 可以算出 $\left(\frac{\delta_3}{\delta_2} + \frac{\delta_2}{\delta_1}\right) \times \frac{1}{2} = \left(\frac{4}{2} + \frac{2}{1}\right) \times \frac{1}{2} = 2$ ，因此平均等待時間的比值越接近 2，則是有越好的效果。然後將每隔一段間距所得到的數個平均等待時間的比值加以統計分析，將各方法分別統計出平均等待時間的比值的第一四分位數、第二四分位數和第三四分位數。

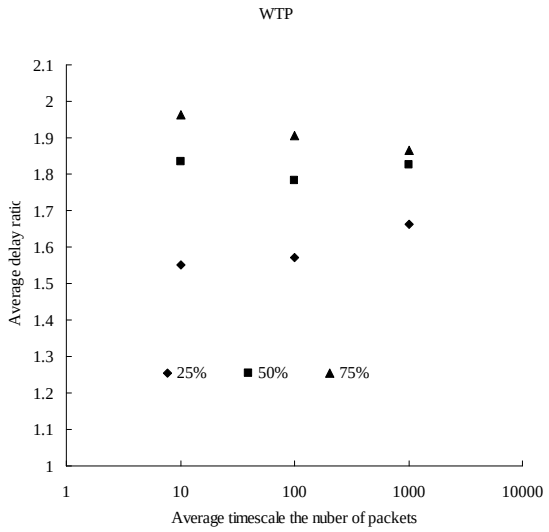

圖 7 WTP 測量間距與平均等待時間的比值

由圖 7 可以看出 WTP 在測量間距越長時會有越穩定的效果，而在每傳送一千個封包測量的實驗中平均等待時間的比值介於 1.66 到 1.87 之間，另外由圖 7 也可以看出來測量的間距越長會有收斂的趨勢。

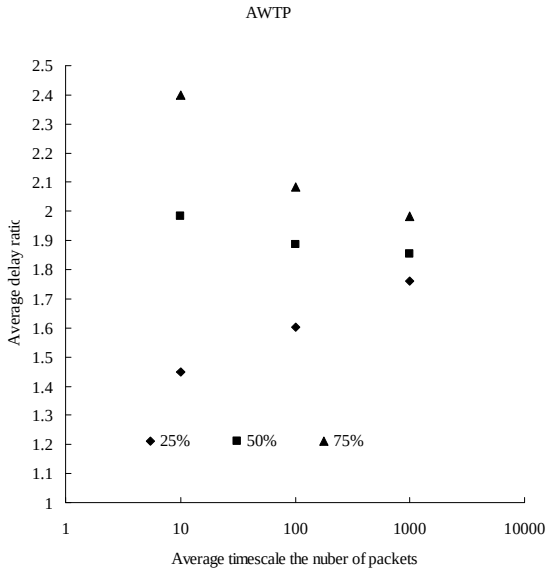

圖 8 AWTP 測量間距與平均等待時間的比值

由圖 8 可以看出 AWTP 在測量間距越長時會有越穩定的效果，而在每傳送一千個封包測量的實驗中平均等待時間的比值介於 1.75 到 1.98 之間，另外由圖 7 和圖 8 也可以看出來 AWTP 收斂於一定值的趨勢較 WTP 快。

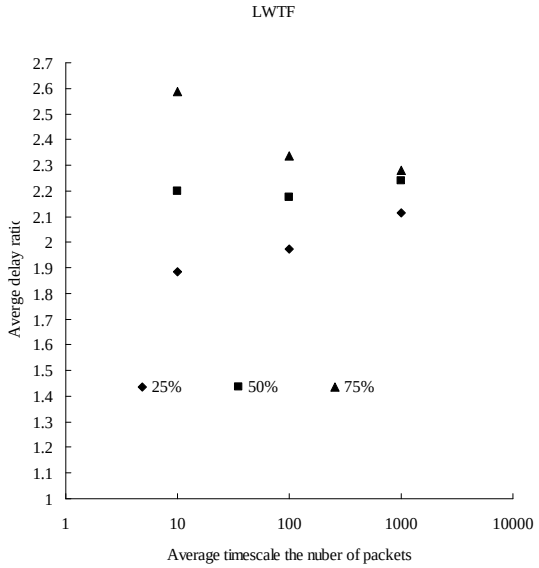

圖 9 LWTF 測量間距與平均等待時間的比值

由圖 9 可以看出 LWTF 在測量間距越長時會有越穩定的效果，而在每傳送一千個封包測量的實驗中平均等待時間的比值介於 2.11 到 2.28 之間，另外由圖 11 和圖 12 也可以看出來 LWTF 收斂的趨勢較 AWTP 快，當然也比 WTP 快。

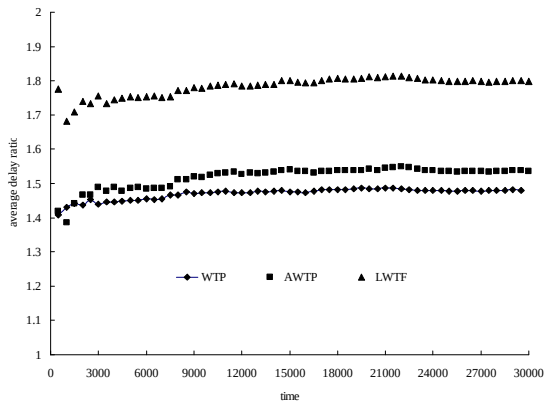

圖 10 WTP、AWTP、LWTF 平均等待時間的比值在不同的時間

圖 10 為每 500 秒測量一次，使用率為 70%，由圖可以看出來 WTP 雖是較快到達穩定的值，但效果比 AWTP 和 LWTF 差，而 LWTF 和 AWTP 達到穩定的值是很接近的，但是效果比 WTP 好。

由本節的實驗可以看出測量的間距對平均等待時間的比值是有影響的，當測量時間越長三種方法都會趨於一定值，效果也越好。另外由本節最後的實驗可以看出來，當實驗到 30000 秒的時候 WTP、AWTP 和 LWTF 都會達到穩定的情形。

5. 結論

本篇提出一個架構在 PDD 模組下的封包排程演算法 LWTF，而演算法的時間複雜度為 $O(N)$，相對於 AWTP 的時間複雜度 $O(N^2)$，因此從實做的角度看，LWTF 是比 AWTP 簡單的方法。

LWTF 採用最長等待時間的觀念，而 AWTP 是用最短工作時間優先的觀念，都有預先考慮目前等待在佇列上封包的傳輸時間，因此在平均等待時間的比值的效果上 LWTF 與 AWTP 相較，有類似的效果，而且 LWTF 比 AWTP 較傾向於提供較高的優先權給高等級的封包。另外，當每個等級的封包抵達速率改變時，相較於其他兩種方法，LWTF 較不易受到影響。

參考文獻

[1] Abhay K. Parekh, Robert G. Gallager, “A Generalized processor Sharing Approach to Flow Control in Intergated Services Networks: The Single Node Case”, IEEE/ACM Transactions on Networking, vol. 1, no.3, 1993.

[2] Charny, “Delay Bounds in a Network with Aggregate Scheduling”, Tech. Rep. Draft version 3, Cisco Systems, 2000.

[3] D. Clark and W. fang, “Explicit Allocation of Best Effort Packet Delivery Service”, IEEE/ACM Transactions on Networking, vol. 6, pp. 362-373, 1998.

[4] Dovrolis, and P. Ramanathan, “A Case for Relative Differentiated Services and the Proportional Differentiation Model”, IEEE Network, 1999.

[5] Dovrolis, D. Stiliadis, and P. Ramanathan, “Proportional Differentiated Services: Delay Differentiation and Packet Scheduling”, ACM SIGCOMM, 1999.

[6] J. L. Boudec, M. Hamdi, L. Blaxevic, and P. Thiran, “Asymmetric Best Effort Service for Packet Networks”, in Proceedings Global Internet Symposium, 1999.

[7] J. S. Sahu, D. Towsley, “A Quantitative Study of Differentiated Services for the Internet”, in Proceedings Global Internet Symposium, 1999.

[8] J. Wroclawski, “the Use of RSVP with IETF integrated Services”, RFC 2210, 1997.

[9] K. Nichols, S. Blake, F. Baker, and D. L. Black, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers”, IETF RFC 2474, 1998.

[10] K. Nichols, V. Jacobson, and L. Zhang, “A Two-bit Differentiated Services Architecture for the Internet”, IETF RFC 2628, 1999.

[11] M. May, J. C. Bolot, C. Diot, and A. Jean-Marie, “Simple Performance Models of Differentiated Services Schemes for the Internet”, in Proceedings IEEE INFOCOM, 1999.

[12] R. Braden, D. Clark, S. Shenker, “Integrated Services in the Internet Architecture: an Overview”, RFC 1633, 1994.

[13] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An Architecture for Differentiated Services”, RFC 2475, 1998.

[14] Stoika, and H. Zhang, “LIRA: An Approach for Service Differentiated Services Network”, Tech. Rep., Texas A&M University, 2000.

[15] Stoika, and H. Zhang, “Providing Guaranteed Services without Per Flow Management”, in Proceedings ACM SIGCOMM, 1999.

[16] Stoika, S. Shenker, and H. Zhang, “Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks”, in Proceedings ACM SIGCOMM, 1998.

[17] Yuan-Cheng Lai, Wei-His Li, “A Novel Scheduler for Proportional Delay Differentiation by Considering Packet Transmission Time”, IEEE Communications Letters, vol. 7, no. 4, pp. 189-191, 2003.