國立交通大學

資訊科學與工程研究所

碩 士 論 文

以行動應用程式為對象的效能導向流量排程演算法

Utility-Based Traffic Scheduling For Mobile Applications

研 究 生：柯承孝

指導教授：邵家健　教授
易志偉　教授

中 華 民 國 一 百 年 九 月

以行動應用程式為對象的效能導向流量排程演算法

Utility-Based Traffic Scheduling For Mobile Applications

研 究 生：柯承孝　　Student：Cheng-Hsiao Keh

指導教授：邵家健　　Advisor：John Kar-kin Zao

易志偉　　Chih-Wei Yi

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

September 2011

Hsinchu, Taiwan, Republic of China

中華民國一百年九月

以行動應用程式為對象的效能導向流量排程演算法

學生：柯承孝　　指導教授：邵家健　易志偉

國立交通大學資訊科學與工程研究所碩士班

摘　　要

在現今的移動通訊網路中，網路服務已經非常普遍，幾乎每位手機用戶都會使用到。但隨著用戶數量迅速的成長，硬體設備漸漸無法滿足大眾的需求，導致網路傳輸速率緩慢、應用程式服務品質下降，因此網路頻寬的不足已經成為一個全世界電信業者共通的課題。若是要增加基地台數目及線路數量，成本將會十分昂貴且緩不濟急，應該從另一個方面著手，試著降低多餘的資料傳輸量，並且尋找有效率的方式應用有限的頻寬資源。

在本篇論文中，將代理伺服器此一元件引進現今的網路架構，設置於手機及網際網路中。手機端的代理伺服器擁有快取的功能，可避免重複的資料傳輸，同時能控管與網路之間的所有連線，並進行流量排程的工作。網路端的代理伺服器會定期向各個應用程式的伺服器抓取新資料，儲存在它的快取中，同時在資料更新時，發送一短訊息告知手機端。所有從手機端送出的資料要求訊息都須經過網路端的代理伺服器，若是代理伺服器中的快取已經過期，才再向應用程式伺服器發送要求。

有了手機端及網路端的代理伺服器的支援後，本文中提出了基於此系統架構下的三個目標: (1)避免重複的資料傳輸造成頻寬浪費。(2)減少網路流量超出負荷及壅塞的情況。(3)針對每個使用者的需求以求將資料效益最大化。第一個目標可由代理伺服器提供的快取功能獲得實現。第二及第三個目標則可由手機端的代理伺服器對資料流量進行適當的排程來達到目的。因此，我們設計出一個以效能函數為基準的排程演算法。此效能函數將以服務品質和個人偏好兩方面做考量來計算其值。演算法會調整每個連線的傳輸順序及各自的流量，以即時的方式做排程，期使在一段時間內所獲得的效能函數值最大化。

實驗部分是以模擬的方式進行，我們基於現在的行動網路架構選擇一個合理的網路流量模型，並且加入了代理伺服器的支援。模擬所得到的數據結果，呈現出在有代理伺服器進行快取和流量排程的情形下，能有效的降低網路流量、減少壅塞的發生以及增加資料效益。未來則可考慮實際將代理伺服器部署於移動通訊網路中，佐以良好的演算法，則能有效減少增加硬體的成本。

關鍵字: 代理伺服器、效能函數、行動網路、線上排程演算法

Utility-Based Traffic Scheduling For Mobile Applications

Student：Cheng-Hsiao Keh

Advisor：Dr. John Kar-kin Zao
Dr. Chih-Wei Yi

Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

In recent years, a variety of services have been available to mobile subscribers. Because the number of subscribers grows too fast, the current infrastructure cannot give services to fulfill everyone's need, which causes reduced data rates and poor service performances.

To solve this problem, we propose *Application Proxies* into current network design. *Mobile Proxy* is set at the cell phone end, which has a local cache and manages all sessions so that traffic scheduling can be achieved. *Cloud Proxy* on the other hand is sited at the external data network namely Internet, which does periodicity crawling to third party application servers. When the content of an application has updated, an short message would be sent to the *Mobile Proxy*.

With the support of Application Proxies, we may attain the following objectives: (1) eliminate redundant data transmission. (2) reduce network congestion and overloaded cases. (3) maximize data values for individual subscribers. First objective can be realized by content caching at *Mobile Proxy*, while second and third objectives will be achieved by sophisticated traffic scheduling mechanism. Thus an utility-based scheduling algorithm, which takes service quality and user preferences into consideration is proposed.

The simulation environment is based on mobile network architecture and corresponding traffic models. The result shows that with suitable caching and traffic scheduling method, our goals can be truly achieved.

Keywords: Application Proxy, Utility Function, Mobile Network, Scheduling Algorithm

誌 謝

本篇論文中的研究進行了一段不算短的時日，能夠獲得今日的成果，絕非我一個人獨力辦到，而是有賴於多位人士的幫助。首先我要感謝的是我的指導教授邵家健老師，從進入研究所以來，他總是能給予我懇切的建議，並協助我一步步的向前邁進。其次要謝謝同組的同學，陳淑華及高偉翔。在研究的過程中時常會遇到難題與挫折，除了與他們彼此討論方法，使得研究內容更加精進之外，同伴間的互相勉勵也讓我的心理更加踏實。最後則要謝謝趙禧綠教授、遠傳電信的黃天立先生及韋殿釗先生，他們給的專業諮詢，指引我朝著正確的研究方向前進。有了這些人士的慷慨協助，才造就了今日本論文的完成。能在這段日子中與他們共事，讓我感到無比的幸運，更是一段珍貴的經驗。

柯承孝

2011 9 月

目 錄

以行動應用程式為對象的效能導向流量排程演算法 i
Utility-Based Traffic Scheduling For Mobile Applications ii
誌 謝 iii
目 錄 iv
表 目 錄 vi
圖 目 錄 vii
符 號 說 明 viii
一、 緒 論 1
1.1 研究背景與動機 1
1.2 研究問題 1
1.3 研究目標 2
1.3.1. 降低無線存取網路(Radio Access Network)中不必要的資料流量 2
1.3.2. 減少網路負荷過載及壅塞的情形 2
1.3.3. 替個別用戶最大化其獲得的資料價值 2
1.4 論文結構與研究流程 2
二、 相 關 文 獻 探 討 4
2.1 效能函數(Utility Function)描述及應用 4
2.2 行動網路中的服務品質(QoS)控管策略 4
三、 研 究 方 法 6
3.1 系統模型(System Model) 6
3.1.1. 行動網路架構 6
3.1.2. 雲端至用戶端訊息傳輸框架(Cloud to Device Messaging, C2DM) 6
3.1.3. 代理伺服器(Application Proxy) 6
3.1.4. 系統架構及運作流程 7
3.1.5. 服務品質參數(QoS Parameters) 8
3.1.6. 流量分類(Traffic Classifications) 9
3.1.7. 用戶檔案(Subscriber Profile) 11
3.1.8. 工作規格(Job's Specifications) 12
3.2 效能導向的排程演算法(Utility-Based Scheduling Algorithm) 13
3.2.1. 概觀(Overview) 13
3.2.2. 假設(Assumptions) 15
3.2.3. 效能函數(Utility Functions) 15
3.2.4. 快取機制(Caching Mechanisms) 20
3.2.5. 應用程式內容預先下載(Application Pre-fetching) 20
3.2.6. 排程策略(Scheduling Strategies) 21
四、 資 料 分 析 與 結 果 29

4.1 模擬設計(Simulation Model) 29
4.1.1. 網路狀況模型(Network Status Model) 29
4.1.2. 流量模型(Traffic Model) 29
4.1.3. 應用程式規範(Application Specifications) 30
4.2 結果分析 31
4.2.1. 整體網路效能評估 31
4.2.2. 應用程式效能評估 32
五、 結 論 35
參 考 資 料 36

表 目 錄

表 1 – LTE服務品質分類表 9
表 2 – 網路品質狀態及其對應的參數值 29
表 3 – 應用程式規格 31

圖 目 錄

圖 1 – 北美洲行動網路的一日平均流量 1
圖 2 – LTE網路架構&代理伺服器 7
圖 3 – 系統運作流程 8
圖 4 – 以服務品質為參數的效能函數圖形(一般性流量) 10
圖 5 – 以服務品質為參數的效能函數圖形(時間重要性流量) 10
圖 6 – 以服務品質為參數的效能函數圖形(品質調整性流量) 11
圖 7 – 工作執行範例 12
圖 8 – 流量控制(a)於RNC上 (b)於手機上 14
圖 9 – 無線資源分配於不同時間尺度下的控制策略 14
圖 10 – 一般性流量的效能函數: (a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率 17
圖 11 – 時間重要性流量的效能函數: (a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率 17
圖 12 – 品質調整性流量的效能函數: (a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率 18
圖 13 – 工作狀態變化示意圖 23
圖 14 – 基於效能及傳輸親合性決定工作執行的順序 24
圖 15 – 以$(r1, r2, \ldots, rN)$為參數的效能概念圖 26
圖 16 – 排程演算法流程圖 28
圖 17 – 網路品質狀態轉移圖 30
圖 18 – 基地台下行流量統計數據 31
圖 19 – 基地台下行流量比較 31
圖 20 – 不同用戶數量下的基地台下行平均傳輸速率 32
圖 21 – 無代理伺服器的手機流量 32
圖 22 – 應用程式平均流量 33
圖 23 – 有代理伺服器的手機流量 33
圖 24 – 應用程式平均回應時間 34
圖 25 – 不同用戶數量下的平均回應時間 34

符 號 說 明

r	資料傳輸速率
r^*	決策者最偏好的傳輸速率
r^o	決策者最不偏好的傳輸速率
r_i	第i個工作的傳輸速率
r_{req}	最小傳輸速率需求
$r_{i,req}$	第i個工作的最小傳輸速率需求
d	延遲時間
d_i	第i個工作的延遲時間
d^*	決策者最偏好的延遲時間
d^o	決策者最不偏好的延遲時間
d_{req}	最大延遲時間需求
$d_{i,req}$	第i個工作的最大延遲時間需求
ε	封包錯誤率
ε^*	決策者最偏好的封包錯誤率
ε^o	決策者最不偏好的封包錯誤率
ε_{req}	最大封包錯誤率需求
$\varepsilon_{i,req}$	第i個工作的最大封包錯誤率需求
U	總體效能函數
U_{bound}	效能邊界值
U_i	第i個工作的總體效能函數
U_T	以傳輸速率為參數的效能函數
U_D	以延遲為參數的效能函數
U_R	以封包錯誤率為參數的效能函數
p_i	使用者對第i個工作的偏好值
M	總資料量
M_i	第i個工作的總資料量
$\tilde{M}$	剩餘資料量
$\tilde{M}_i$	第i個工作的剩餘資料量
job_i	第i個工作
N_{app}	用戶所存取應用程式的數量
W	權重函數
C	工作建立時間
C_i	第i個工作的建立時間
T_c	現在時間
P_{access}	用戶存取應用程式的間隔時間機率分布
P_{update}	應用程式更新的間隔時間機率分布
P_{hit}	預先下載的應用程式被用戶存取的機率
$M_{prefetch}$	預先下載的期望資料量
τ_p	網路參數更新週期
r_{max}	單一用戶最大可用頻寬
B_{max}	無線網路總頻寬

φ_{data}	網路額外負擔時間量
$U_{threshold}$	工作狀態變化的效能門檻值
$\Omega_{utility}$	效能下限比率
$\widetilde{U}_i$	第i個工作的邊際效能

一、 緒 論

1.1 研究背景與動機

自從手機被發明以來，有很長的一段時間被當成單純的通話裝置使用。不過隨著科技的演進，通訊系統的架構已經能整合聲音及資料網路[23]，許多不同種類的應用程式服務也開始被提供到手機上，而能存取這些應用程式服務的手機就被稱為智慧型手機。行動網路與一般廣泛使用的網際網路有若干不同之處，其一是行動網路的末梢需要經由無線通訊的方式和客戶端溝通。其二則是用戶在存取網路時擁有移動的性質，因此網路要確保與用戶之間連線的持續性。其三是由於傳輸介質的因素，無線頻寬的擴增較有線頻寬受到更大的侷限。在這種限制下，業界及學界都不斷地思索改善的方式，行動網路的標準更不斷的演進，現在已經發展到第四世代，也就是 LTE 及 WiMAX 標準。隨著系統架構的改變，新的網路協定和方法也以解決不同的問題為目的不斷被提出。

1.2 研究問題

由於行動網路技術的成熟，手機上能提供的服務種類也大幅成長，這造成了資料流量所佔的比例遠大於傳統的通話流量。而用戶數量的大幅成長，使得網路頻寬不足的問題也漸漸浮上檯面[21,22]。造成此一問題的主要原因是行動網路的移動性，讓使用者可以在任何地點，任何時間存取網路服務。在大部分的情況下，頻寬都可以滿足使用者的需求，但是若是人群大量湧入特定地方，例如市區、百貨公司，而且又同時使用網路，那麼就會造成網路壅塞的情況。這種人群密度較高的場所被稱做熱點(hot spot)。在(圖 1)中可以看到使用者的行為都遵循著類似的模式，大量使用頻寬的行為其實只發生在特定的時段中，這造成了流量在時間層面的極度不均衡。若是要增加更多基地台，或是增加線路在這些地方，事實上是非常不經濟的做法，因為在大部分的時間裡，這些額外的頻寬都是處於閒置的狀態。因此眾多研究者們針對如何改善行動網路壅塞的問題，提出了種種可能的架構及方法[5,24,25,26]。

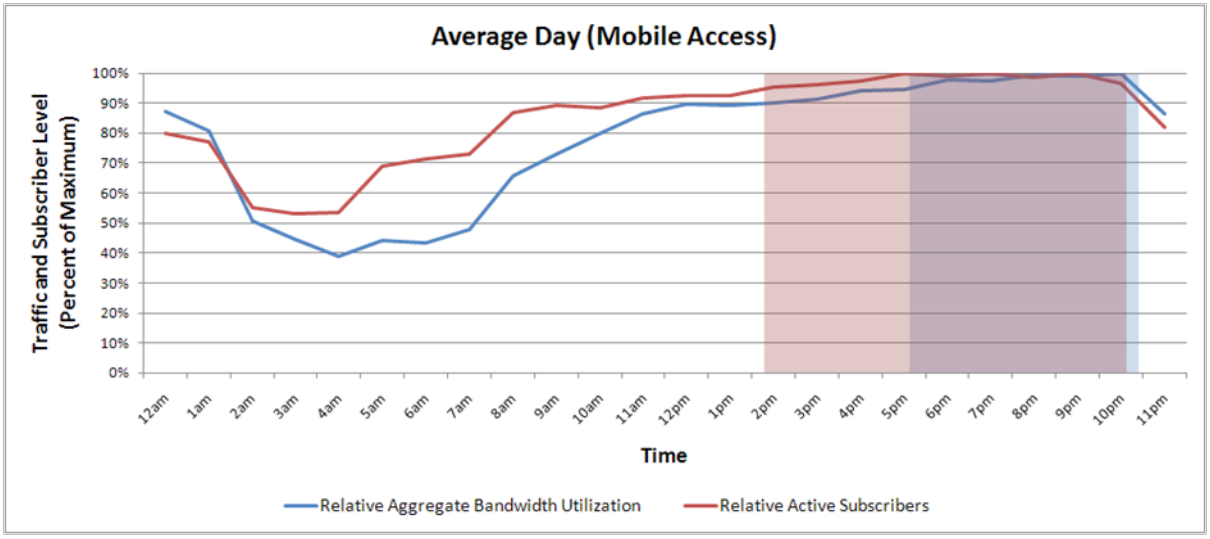

圖 1 – 北美洲行動網路的一日平均流量

有些研究者試著從數學的角度去處理此問題，將其化為適合的最佳化問題型式後解決。另外一種可能的方法則是以適當的演算法，將流量分散至不同路由中，例如基於最大化傳輸速率(max-min)分配，或是按照比例性原則分配，以及基於最小化延遲來分配等方式。但是在實際的網路中，流量變化是無法完全模擬的，因此只有運用良好的流量分散策略還不夠，當流量超出預期時還需要有壅塞防止(congestion avoidance)措施的支援。常見的幾種機制例如將超過負荷的封包踢除，暫停服務新的連線要求，或是隨機早期偵測(Random Early Detection)都是現今的網路所採行的方式。

綜觀過去的研究，都是在網路的樞紐或是路由器上處理壅塞的情況，不過隨著使用者數量逐漸成長，所能達到的效果事實上是很有限的。在本篇論文中，引進了代理伺服器此元件並安裝於手機上，如此一來代理伺服器則可在手機端對流量進行控制，從根本上減少網路的流量。

1.3 研究目標

本篇論文中為了解決上述的問題，提出以下三個目標:

1.3.1. 降低無線存取網路(Radio Access Network)中不必要的資料流量

訊號在無線網路中經由空氣介質傳輸，會有明顯的衰減現象，而且頻寬的擴充也受到了大幅限制。因此如何善用頻寬，避免多餘的資料傳輸變成一個很重要的課題。針對此一問題，可運用快取的機制來解決。在手機上設置一個代理伺服器，擁有快取儲存區可保存各個應用程式的資料，則可避免多次傳輸同樣資訊的情形發生。

1.3.2. 減少網路負荷過載及壅塞的情形

從行動網路的調查報告中，可發現到在不同的時間，網路流量的差異性是很巨大的。其原因在於群眾行為有著類似的模式，傾向於在同一時間同一地點使用網路資源，因此造成了傳輸速率緩慢的情形。但是在大部分的時間內，網路頻寬的使用率卻相對的低。在手機上的代理伺服器，可在網路流量較低時預先下載應用程式的資料，並且根據網路狀況動態調整下載中使用的頻寬，則可減少網路壅塞的情況。

1.3.3. 替個別用戶最大化其獲得的資料價值

每個應用程式都有其傳輸上的需求，也就是服務品質(Quality of Service,QoS)，對用戶而言，每個應用程式也有著不同的重要性。在本論文中，我們提倡一個運用效能函數估算資料價值的方法。效能函數是一種量化資料價值的手段，運算中會把服務品質及用戶的偏好納入考量計算。手機上的代理伺服器則可根據效能函數的輸出值，找出適當的資料傳輸方式及順序，以最大化所獲得的資料價值。如此則可確保各種資料類型的服務品質以及客戶的個人偏好都能獲得滿足。

1.4 論文結構與研究流程

本篇論文的剩餘部份將依序說明以下的內容。在第二章中，我們會先介紹與本研究方法相關的文獻及其差異性。在第三章中將詳細說明研究的系統架構及方法。第四章則

是針對所提出的方法以模擬方式做驗證，並分析其結果。最後在第五章中進行全篇研究的總結，並且評估未來可能的發展方向。

二、 相 關 文 獻 探 討

2.1 效能函數(Utility Function)描述及應用

效能函數原本是經濟學上的一個概念[33]，它指的是某個決策能提供給使用者的滿意程度，當使用者越偏好特定結果，則其效能也越高。那時還沒有人將此說法用於資訊工程的領域中。直到 1995 年 Shenker 提出了較完整的說法[1]，他將效能表示成頻寬的函數，並且針對不同流量類型用不同形狀的效能函數來描述流量的特徵，不過在該篇文獻中並沒有提到如何量化和計算效能函數值的部分。Zimmermann 和 Killat 在[4]中假設效能函數可以對數函數的形式來呈現，並定義了效能最佳化的問題。Dharwadkar[3]則將效能函數分成梯級(step)、線性(linear)、凹形(concave)三種形式，並提出了一個線上排程的原則方法，可以達到效能最大化的目的。Kelly[5]等人參閱了前人的研究結果，將效能函數假設有遞增(increasing)、凹形(concave)、可微分(continuously differentiable)的性質，並證明了對流量分配給各個用戶的最佳化問題，是數學上可解決的。

之後，效能最佳化問題廣泛應用於各種環境，被研究者提出以及驗證。例如 Harks 等人[6]在基於效能公平的假設下，推導出一個適合的方式以評估各個用戶所獲得效能的公平性，並且提出了對應的優先權付費(priority pricing)機制。Chang 等人[7]針對了 HTTP 流量類型建立了模型及效能函數，並解決了在這種假設下的頻寬分配最佳化問題。Chen 等人[8]則考慮於無線感測網路(wireless sensor network)中，把元件的電力消耗作為額外的成本，建立了一個效能-電力的模型，並在數學上解決了此最佳化問題。

2.2 行動網路中的服務品質(QoS)控管策略

在網際網路剛開始發展的年代，封包的傳輸並沒有考慮所謂服務品質的概念，而一概採用最大效能(best-effort)的傳送方式。這種方法使得頻寬資源的使用效率不彰，用戶也無法獲得良好的應用程式體驗。在這種情形之下，基於服務品質的網路架構與方法開始慢慢被提出。能提供服務品質的網路，代表可針對應用程式的需求運用不同的傳輸策略。可能的需求包括頻寬、延遲、排程優先權等等，隨著行動網路的崛起，在行動網路環境下的服務品質調節，也變成了一個重要的研究主題。

Singh[9]在 1996 年時提出了行動網路應該提供兩個額外的服務品質參數:在切換區域時保證無間斷的通訊以及服務的規格降低(degradation)，同時他也提出了能支援服務品質的網路架構及傳輸協定。Campbell 等人[12]也提出了一個可程式化的行動網路架構，能動態的適應網路情況並對應用程式進行傳輸的調節。Mahadevan 和 Sivalingam[11]則針對路由(routing)層面的服務品質調控，提出了對應的行動網路架構及機制。

除了能支援服務品質的網路結構，另外一群人則針對頻寬的分配方式進行研究。在[10]中，Kroner 在考慮服務品質的條件下，對 UMTS 網路中無線資源的分配提出了三種可能的方法:基於傳輸速率分配、基於傳輸功率(transmission power)分配，以及基於

最大化傳輸速率(max/min)分配，並在模擬系統中進行了效能和優劣的分析和比較。在[13]中採用了模糊邏輯(fuzzy logic)的概念，將傳輸資料根據服務品質分成四個類別:交談式(conversational)、影像(streaming)、傳統、背景執行。接著將觀測及計算而得的網路參數做為模糊邏輯運算的輸入，根據設定好的規則列表以決定適當的頻寬分配方式。Yang 等人[14]根據應用程式對頻寬的需求定義了相對的效能值，每當有新的連線進來時，則決定一個新的頻寬分配方式，以最大化所獲取的效能值。另一種可能頻寬分配方法是建立一個決策樹(decision tree)[15]，將服務品質可能的參數以階層式連接，越上層的表示其資訊越為重要並且先抉擇，接著根據適當的分析後找出最適合的路徑。

頻寬分配和流量排程的方法都有一個共通的要求，就是需實現某種條件下的公平。由於兩者能處理的方向不同，演算法的目標也有些差異性。對資源分配來說，如何有效率的使用有限的資源是最重要的課題。相較之下，一般在流量排程演算法重視的是如何降低最大延遲以及優先傳送較重要的資料。例如在無線公平排程方法(Wireless Fair Queuing, WFQ)[16]中，演算法會選擇錯誤率最低的頻道先服務，被服務過的頻道會降低其優先權，如此就可以避免服務分配不公平的情形發生。Wang 等人在[17]中改良此方法，將流量分成即時(real-time)和非即時(non real-time)兩種，並採用不同的策略處理，接著證明了在流量分類的情形下，仍然能夠保證有限的延遲(bounded delay)和公平性。以上的流量排程演算法，雖然考慮了頻道的狀況，但是並未替不同應用程式設定不同的服務品質。當資料類型越來越多時，若是仍然採用統一的原則進行排程，則會發生應用程式效能低落的問題，因此如何滿足服務品質的需求也成了研究者關注的主題之一。Shi 等人[18]提出了一個排程演算法，主要概念是透過動態的將手機切換至睡眠模式，在滿足最低頻寬需求的條件下，同時達到省電的目的。Mendez 等人[19]考慮在以分碼多重存取(Code Division Multiple Access, CDMA)作為資源分配方式的架構中，針對多媒體類型的資料來進行排程。排程的基本原則是根據服務品質和資料量決定其權重，並給予不同程度的頻寬。Choi 等人[20]提出了一個上層的排程演算法，在一個連線進來時，先以整體資料的角度去計算其效能值，以決定工作執行的順序，接著將此工作交由下層的封包排程處理。由於效能跟服務品質需求相關，因此可保證執行的工作能獲得最高的效能。

本論文中所提倡的方法與過去研究最大的不同之處在於排程演算法是在網路中的客戶端運作，而非傳統在路由器或是閘道器上進行。在伺服器上能做的策略是針對已經產生的資料流量而定，因此對網路的傳輸改善十分有限。但是在手機端可以在應用程式的層面直接決定各個連線的控管策略，如此則有可能大幅提升網路的效能。

三、 研 究 方 法

3.1 系統模型(System Model)

3.1.1. 行動網路架構

隨著手機用戶數量急遽增加，研究者和業者都不斷尋找著改良目前網路結構的方法，行動網路的標準和演算法也日新月異。發展到今日為止，被正式公認的新一代標準是3GPP 長期演進技術(3GPP Long Term Evolution, LTE)，也就是俗稱的第四代無線通訊技術[23]。本論文中所提出的額外網路元件及方法，都假設下層硬體設施為 LTE 標準和其網路架構。在 LTE 網路中包括了以下幾個元件:

- 先進基地台(Evolved NodeB, eNB) – 這個元件在 LTE 無線存取網路中扮演了基地台的角色，它除了負責與手機之間的無線傳輸，並且也進行訊號資源的管理和手機區域切換(handover)。

- 行動性管理模組(Mobility Management Entity, MME) – 它負責持續追蹤手機的所在位置，並驗證其合法性，接著將其註冊到正確的閘道器上。

- 服務閘道器(Serving Gateway, SGW) – 此閘道器負責決定封包的路由並進行轉送。用戶的個人資訊也在此進行註冊，例如現在所在位置、可使用的服務、以及路由資訊等等。

- 資料網路閘道器(PDN Gateway, PGW) – 此閘道器提供了手機與公開資料網路(Public Data Network, PDN)之間連線的橋樑，並針對各個用戶執行不同的策略，例如封包過濾，網路服務費用計算等功能。

3.1.2. 雲端至用戶端訊息傳輸框架(Cloud to Device Messaging, C2DM)

行動通訊系統結合雲端運算，在智慧型手機興起後，被認為是未來十分有潛力的應用方式。為了輔助這個新概念， Google 提供了 C2DM 的服務，它提供了簡單的方式讓應用程式伺服器通知手機端，有新資料可供下載。手機端收到通知後，可選擇不做任何回應，也可以跟伺服器要求連線取得資料。從開發者的角度來說，可以將大部分的運算在伺服器端就完成，而利用 C2DM 告知應用程式來取得結果，因而降低手機的處理負擔。本論文中使用 C2DM 技術作為手機端和資料網路間的溝通橋樑，實際應用方式將於下一節說明。

3.1.3. 代理伺服器(Application Proxy)

為了實現本論文中提出的方法，需要將代理伺服器此元件加入現有的網路架構中。根據代理伺服器其負責的功能不同，可分為行動端代理伺服器(MS Proxy)及雲端代理伺服器(Cloud Proxy)兩種，部署於網路中不同的位置:

- 雲端代理伺服器 – 這個元件部署於公開資料網路也就是網際網路中，並作為第三

方應用程式伺服器的逆向代理(reverse proxy)來運作。所有從手機端使用的應用服務都會經過此代理伺服器，如果快取中沒有該應用程式最新的資料，才將此資料要求轉送至對應的應用程式伺服器。除此之外，它還會定期去查看應用程式是否有更新，以分析各個應用程式的更新模式，這樣即可確定快取是否過期。每當應用程式伺服器有新資訊產生時，雲端代理伺服器會利用 C2DM 服務傳送一個短訊通知手機。各個應用程式的更新通知和更新週期，皆會傳送到行動端代理伺服器上，成為排程策略的參考資訊。

- 行動端代理伺服器 – 這個元件設置於手機上，手機與外界進行連線都需經由此代理伺服器。他擁有獨立的快取空間，負責儲存應用程式最新的內容副本。它不只會監控網路流量的狀態，更可進一步對各個連線進行程式化的操作，例如改變連線開始的時間、暫停一個連線、或是傳輸速率的調整等等。另一方面，行動端與雲端的代理伺服器可經由C2DM服務進行點對點的加密傳輸，以取得在雲端所獲得的資訊。藉由以上敘述的功能，行動端代理伺服器則能根據演算法，實現適當的工作排程。

代理伺服器詳細的硬體規格以及功能實做方法，並非本論文的重點，因此不再贅述。接下來章節中所提出的演算法，皆假定代理伺服器已部署於網路中，且能提供所需的功能支援和網絡資訊。

3.1.4. 系統架構及運作流程

將3.1.3所提出的代理伺服器加入現在的網路後，所呈現的架構就如同(圖 2)所示。行動端代理伺服器位於行動裝置上，作為手機的系統程式控管所有對外的連線。雲端代理伺服器則位於網際網路上，負責從各個應用程式伺服器下載資料。須注意在圖中並未表現出 LTE 網路中所有的元件，而只表示資料會經過的部分。黑色實線表示應用資料傳輸的路徑，紅色虛線則是雲端代理伺服器和手機端代理伺服器傳送訊息的路徑。使用 C2DM 服務做為資訊的傳輸方式有著若干優點，其一是 C2DM 是專門設計給第三應用程式伺服器傳送一些短訊給手機端的服務，不論是架構或是方法上都較為完善，讓開發者不需自行設計額外的溝通方式。其二則是短訊的傳輸是由 C2DM 伺服器負責控制，除了過

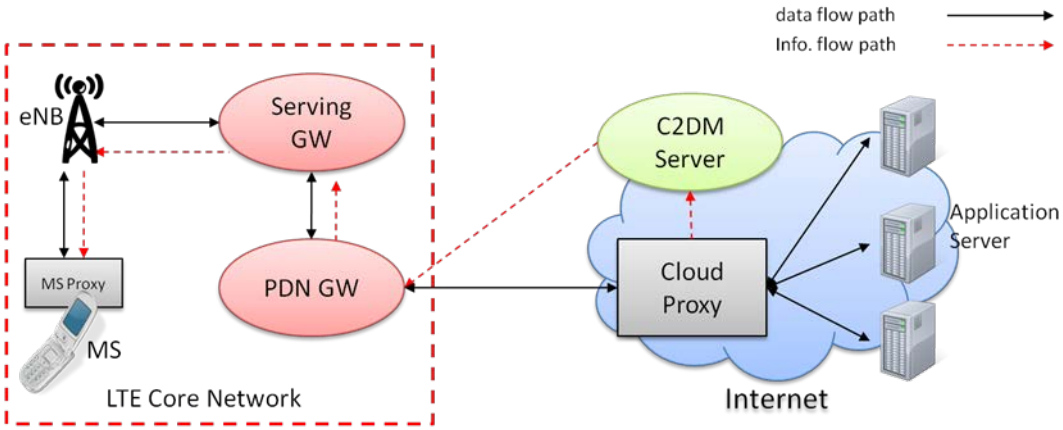

圖 2 – LTE 網路架構&代理伺服器

濾重複的短訊之外，它也會知道行動裝置是否在線，當行動裝置運作時才將短訊送至目標裝置，因此可以減少大量的網路額外負擔。

(圖 3)將代理伺服器中的各個模組以及其功能以方塊圖的方式描繪了出來，圖中紅色的方塊代表手機用戶，黑色的方塊代表手機端代理伺服器，綠色的方塊代表雲端代理伺服器，而藍色的方塊則代表第三方應用程式伺服器。當用戶要求某個應用程式的資料時，手機端代理伺服器會先搜尋快取中有沒有最新的內容，如果沒有找到則將此要求轉

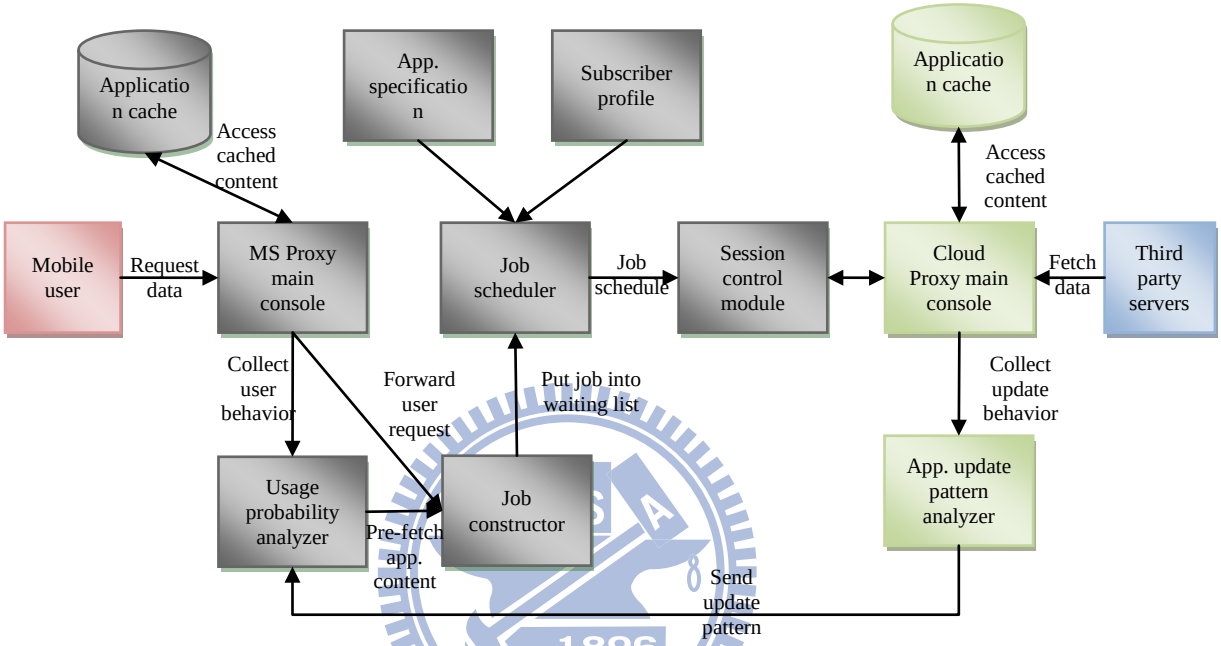

圖 3 - 系統運作流程

換成演算法可處理的工作(job)格式，作為排程策略的輸入。除了使用者的要求之外，代理伺服器還會分析使用者的行為，預先下載(pre-fetch)高機率被使用的應用程式。除了進行排程的目標工作之外，應用程式規範(application specification)及用戶檔案(subscriber‘s profile)也是排程演算法必要的輸入。應用程式規範指的是服務商對應用程式定義的服務品質(Quality of Service, QoS)需求，而用戶檔案則包括了使用者對應用程式的個人偏好及用戶歷史行為。有了適當的輸入資訊後，演算法就能據此找出一個可能的工作排程方式。決定好的排程將交由代理伺服器的連線控制模組(session control module)執行，開始進行應用程式資料的下載。

雲端代理伺服器接收到從手機端來的連線，會先搜尋快取，如果快取中沒有對應內容才向應用程式伺服器下載。除了轉送手機端來的連線要求，雲端代理伺服器還會定期自動查看應用程式伺服器的情況，以分析應用程式的更新週期。雲端代理伺服器會將更新週期的資訊利用 C2DM 的機制送給手機端的代理伺服器，與用戶行為模式一同作為決定是否預先下載的參考。

3.1.5. 服務品質參數(QoS Parameters)

QCI	Resource type	Priority	Packet delay budget	Packet error loss rate	Example services
1	GBR	2	100 ms	10^{-2}	Conversational voice
2		4	150 ms	10^{-3}	Conversational video (live streaming)
3		3	50 ms	10^{-3}	Real time gaming
4		5	300 ms	10^{-6}	Non-conversational video (buffered streaming)
5	Non-GBR	1	100 ms	10^{-3}	IMS signaling
6		6	300 ms	10^{-6}	Video (buffered streaming) TCP-based (e.g., www, e-mail, chat, ftp, p2p file sharing, progressive video, etc.)
7		7	100 ms	10^{-6}	Voice, Video (live streaming), Interactive gaming
8		8	300ms	10^{-3}	Video (buffered streaming) TCP-based (e.g., www, e-mail, chat, ftp, p2p file sharing, progressive video, etc.)
9		9		10^{-6}	

表 1 – LTE 服務品質分類表

在 LTE 行動網路中所定義的服務品質類型，基本上是 UMTS 標準中的延伸和擴充，在(表 1)中可以看到有九種可能的服務類型，以傳輸速率(bit rate)、封包延遲(packet delay)，及封包錯誤或遺失率(packet error or loss)這三種傳輸參數來描述服務品質層級。為了增加演算法與網路架構之間的親和度並提升效能，本篇論文中也將採用同樣概念的參數，茲說明如下:

- 傳輸速率(transfer rate) – 指的是每單位時間中所收到的資料量，是最普遍的效能評估指標。
- 延遲時間(total delay) – 與 LTE 標準的定義方式不同，這邊指的是整份資料下載完成加上等待所花的時間，而非傳送單一封包的延遲。
- 封包錯誤率(packet error rate) – 在無線環境中用以評估頻道品質的中一個指標，我們將封包遺失率也算進封包錯誤率中。

這三種服務品質的需求會定義在應用程式的規範中，並且當作演算法的其中一項輸入。為了簡單起見，我們假設所有應用程式對傳輸質量的要求皆只包含以上三種參數。在下一小節中將會描述不同應用程式的特性，並根據服務品質參數和效能的關係進行分類。

3.1.6. 流量分類(Traffic Classifications)

隨著智慧型手機的興起，應用程式種類也越來越多。為了提供使用者良好的服務品質，每個應用程式對傳輸參數皆有不同程度的要求，例如封包遺失率、頻寬、延遲時間等等。基於過去的研究[1,3,4,5]，我們描繪出這些參數跟用戶體驗之間可能的關係後進行分類，並採用用戶效能作為量化和測量用戶滿意度的方式。根據效能函數形狀，可將流量類型區分為以下三種:一般性(Ordinary Traffic)、時間重要性(Time-Critical

Traffic)、品質調整性(Quality-Scalable Traffic)。

- 一般性 – 這類應用程式對服務品質的要求不高，甚至沒有明確的需求，而且對延遲的容忍度也較高。這意味著當傳輸速率較低時，增加頻寬所能獲得的邊際效益(marginal benefits)是很高的。但是當傳輸速率越來越高時，增加頻寬所能提升的用戶滿意度卻越來越不明顯。有著此種特徵的應用程式像是網頁瀏覽、檔案下載，和收發郵件等等。從(圖 4)中可以觀察到此類應用程式的特性，一開始增加傳輸速率時，效能有著明顯的增長，但是隨著速率越來越高，額外效益也隨之變差。而傳輸時間越長，用戶體驗也就是效能會呈現不斷下降的趨勢，不過由於此類資訊沒有

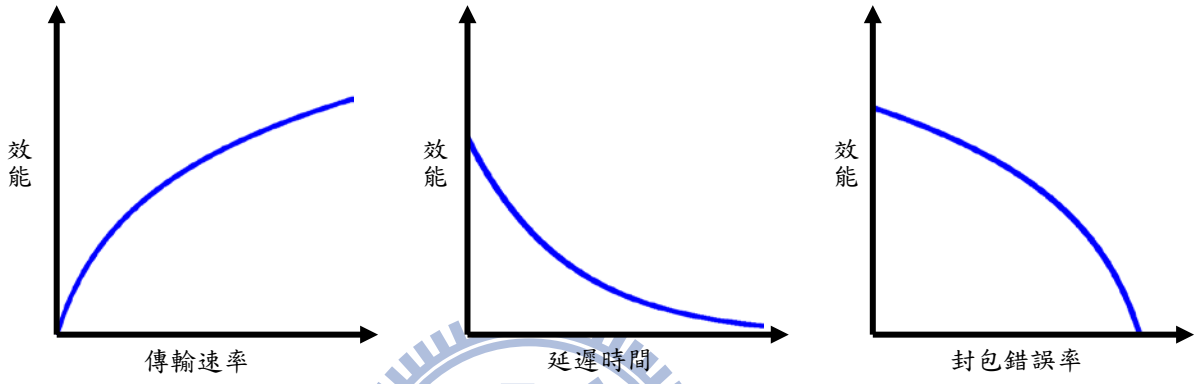

圖 4 – 以服務品質為參數的效能函數圖形(一般性流量)

明確的時間限制，因此不會導致效能變成零的情況。封包錯誤率與效能之間的關係也是成反比的，與傳輸時間不同之處在於，一般應用程式有其能接受的封包錯誤率上限，當超過此限制時收到的資料就不再擁有可辨識的資訊。

- 時間重要性 – 有些應用程式的資料更新十分迅速，而且講求時效性，若是超過限制的時間才取得，對使用者來說就失去了其原有的價值，因此對服務品質的要求也

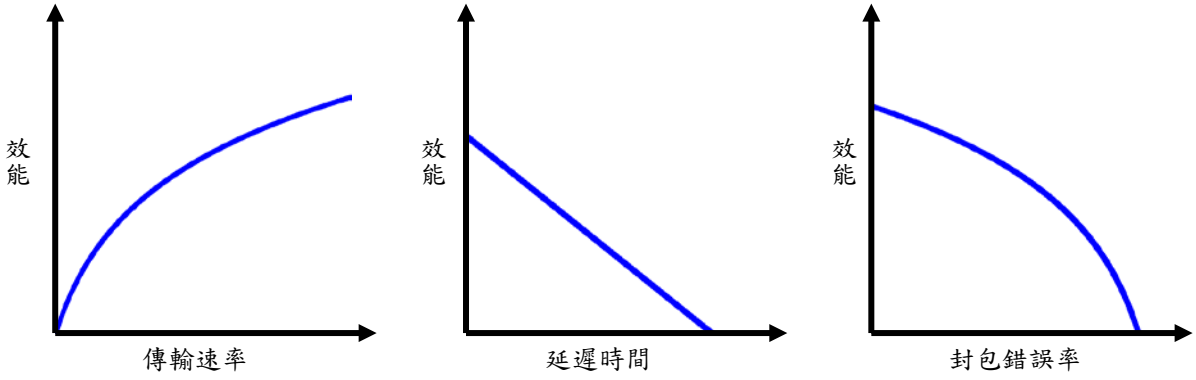

圖 5 – 以服務品質為參數的效能函數圖形(時間重要性流量)

較為嚴苛。我們將股票內容跟新聞定義於此類別中。(圖 5)顯示了可能的效能圖形，與一般性流量的差別只在於傳輸潛伏時間與效能之間的關係，當時間超過應用程式設定的上限，收到的資料就沒有任何效能可言。

- 品質調整性 – 對影片跟圖像來說，不一定需要完整收到全部封包才能辨識裡頭的

內容，因此可以對每個服務品質參數定義多個可能的需求等級，並根據現在的網路資源調選擇可接受的品質層級。Google Earth 的地面圖像和 YouTube 的影片皆屬於此類型。可能的效能函數圖形如(圖 6)，可以看到在傳輸速率的層面，函數形狀就像階梯一般有較明顯的間隔。舉例來說，在 YouTube 播放影片時，有多種解析度可選擇，當用戶頻寬能滿足某個需求時，就能獲得相應的效能。但是頻寬介於高解析度和低解析度需求之間時，仍然只能播放低解析度的影片，當頻寬符合高解析度的

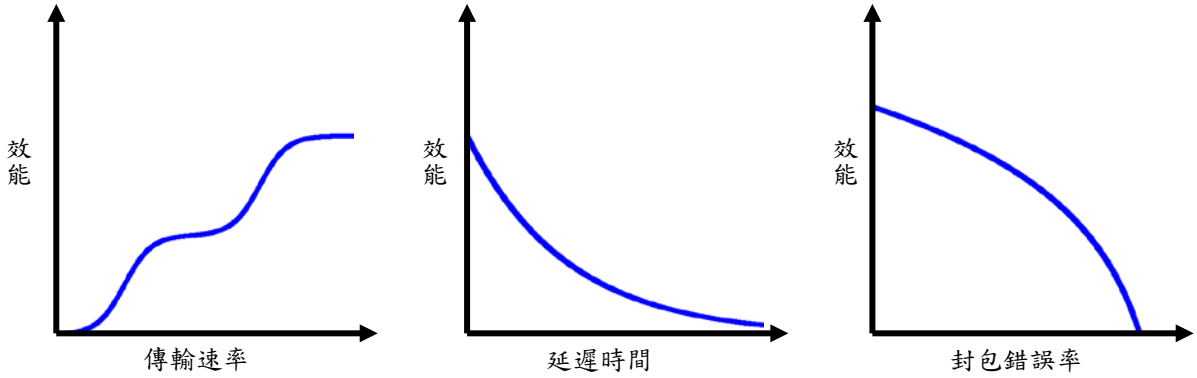

圖 6 – 以服務品質為參數的效能函數圖形(品質調整性流量)

需求，用戶才能獲得較高的效能。因此品質調整性的流量就如同將多個一般性流量組成群集，它們對於傳輸速率有不同的品質要求，但可接受同等的傳輸潛伏時間及封包錯誤率，因為它們都屬於同一種應用程式的內容。

3.1.7. 用戶檔案(Subscriber Profile)

我們的研究目標包含了替使用者取得最大化的資料價值，在效能函數及排程演算法中都需要考慮特定的個人資訊。這些資料有些是由用戶設定，有的則是由代理伺服器主動收集，儲存在用戶檔案中，裡頭包含了以下內容:

- 應用程式偏好 – 用戶需要決定各個應用資料的重要性，也就是此工作在傳輸時的優先權。此資訊以編號方式進行設定，數字 1 表示優先權最高，不同應用程式也可以設定為同一優先權。進行流量排程時會將用戶偏好作為演算法的輸入之一，若是一個工作的優先順序較高，那麼執行它會帶來額外的用戶效能補正。

- 應用程式預先下載 – 當用戶認為某種資訊對他十分重要，希望即時取得最新的內容，那他可以在用戶檔案中設定預先下載，則代理伺服器會在空閒的時候自動更新該應用程式的內容。這種做法對用戶來說有若干個好處，其一是使反應時間最小化，其二是避免錯過可能的新資訊。不過需要注意，若選擇要預先下載的資料太多，超過頻寬資源可負荷的程度，可能使得此機制無法完全發揮其作用。

- 用戶歷史行為 – 手機端的代理伺服器會記錄用戶存取應用程式的次數、時間，以及使用的頻寬。根據這些資訊，可以幫助演算法在未來的排程中調整工作的傳輸速率及時間，將資料流量分散到不同時間處理以減少網路負荷和壅塞的情況發生。同時可用來分析應用程式存取週期，與更新週期一同作為考慮是否預先下載的因素。

3.1.8. 工作規格(Job's Specifications)

每當用戶使用應用程式，或是代理伺服器決定自動下載資料時，就有一個對應於此要求(request)的工作(job)產生。排程演算法的目標，就是決定適合的方式完成這些工作，以求最大化使用者效能。一個工作的規格可以用以下資訊進行表示。

- 應用程式規格(Application Specifications) – 應用程式規格中包含了服務品質參數和流量類型，在 3.1.5 和 3.1.6 小節已經對這兩個名詞進行了完整的定義，每個應用程式都會有其固定的服務品質需求和所屬的流量類型。根據以上規格我們能建立對應的效能模型，用以計算完成此工作可獲得的效能值。

- 優先權(Priority) – 這個參數指的是用戶認為此資料對他而言的重要性，在用戶檔案中進行設定。這個數值並非永遠保持不變，演算法可決定於排程中動態調整優先權。

- 資料量 – 說明這個工作總共大小是多少位元組，與傳輸速率配合就可以估計傳輸時間，以作為效能函數可能的輸入之一。

- 建立時間 – 使用者要求應用程式資料或是代理伺服器自動下載發生的時間點，也

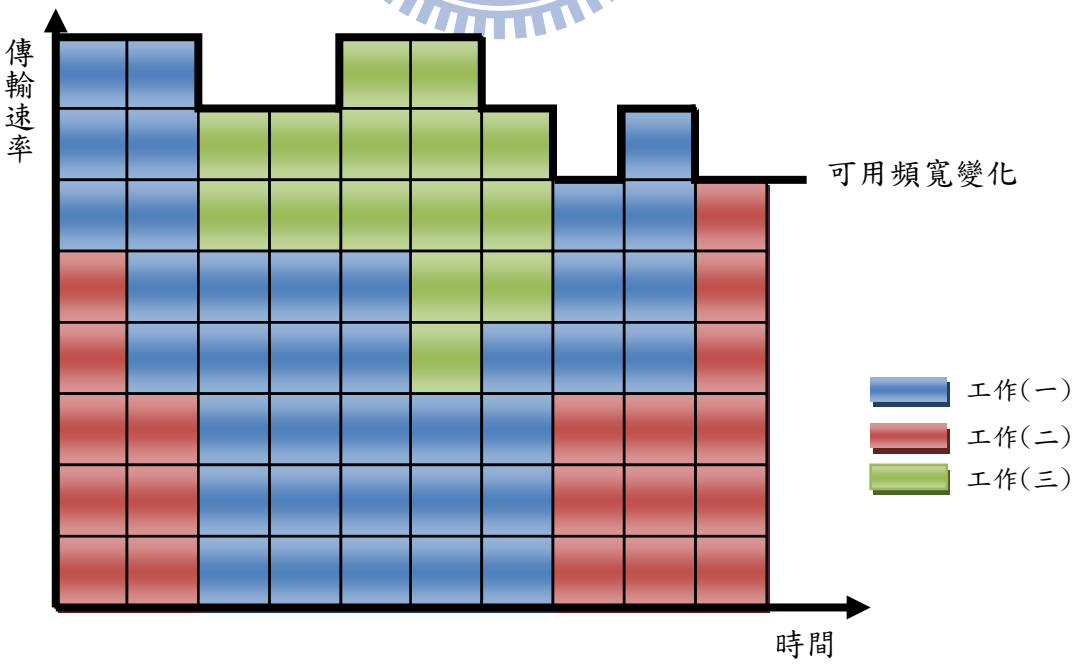

圖 7 – 工作執行範例

就是工作的最早可能執行時間(earliest starting time)。

- 延遲時間(delay) – 這裡指的是系統現在時間減去工作建立時間。為了保證各個工作都能獲得服務的機會，當延遲時間越來越接近工作的完成期限時，執行它所能獲得的額外效能補正也隨之成長。

在我們定義的系統模型中所有工作皆先由排程演算法決定適合的傳輸時間和傳輸策略，然後交給通訊端控制模組(session control module)負責執行。由於代理伺服器能控管所有對外連線，因此可以直接對通訊端進行程式化的操作。例如決定下載開始時間、暫停/恢復傳輸、調節傳輸速率。注意在排程中將時間及頻寬資源皆視為離散，即有最小單位存在。(圖 7)描繪了一個可能的情境，說明工作如何排程以及執行。一開始有兩個待執行的工作，工作(一)跟(二)。代理伺服器基於效能因素來分配給他們適當的傳輸頻寬，隨著時間推移，可用頻寬以及訊號品質等網路參數也會改變，代理伺服器也會根據網路狀況動態調整各個工作的速率。時間到了第三格時，有一個新的工作(三)開始執行，代理伺服器判斷此工作比(二)重要，因此就暫停(二)的傳輸，將頻寬分配給(三)。而當(三)完成時，就繼續(二)的傳輸。

3.2 效能導向的排程演算法(Utility-Based Scheduling Algorithm)

3.2.1. 概觀(Overview)

行動網路和傳統有線網路在特性上有若干不同之處，其一是資料需經由無線訊號傳遞，因此無線環境中的頻寬跟有線網路比起來受到較大的限制。另一方面由於訊號在空氣中傳輸，背景雜訊所佔的比例相對較高，造成通訊品質時常有大幅的波動。考慮以上這些因素，行動網路上對不同流量類型進行頻寬資源分配(bandwidth resource allocation)以及流量排程(traffic scheduling)的研究著重於如何有效率的使用頻寬資源，並在網路狀況不佳時滿足不同應用程式服務品質的需求。

在本論文中所提倡的方法也是針對行動網路的特性而設計的，但是跟過去研究關注的層面則有所差異。傳統上進行頻寬分配和流量排程的元件是無線電網路控制器(Radio Network Controller, RNC)，因此相關的演算法也是運作於此位置。在我們提出的系統中有代理伺服器的存在，可以在手機端決定適合的工作排程和傳輸速率，從流量源頭直接進行可能的控制。從(圖 8)中可以觀察到兩種方式的差異，在一個對話(session)中，無線電網路控制器扮演了中繼點的角色，它能對到達的各個流量根據某種公平的原則分配頻寬並轉送封包至對應的目的地，但是不主動建立連線和結束連線。若是單位時間內到達的封包數量太多，只能選擇將封包丟棄，是一種被動(passive)的運作模式。當手機上有代理伺服器的存在，則可以直接對各個連線做控制，包括分配頻寬、暫停/恢復連線、以及選擇下載時間。當網路狀況不佳時，可以動態調整頻寬，將資源用於執行最重要的工作，屬於(proactive)的控制方式。

本研究跟傳統方法在不同的層次中處理資料流量，由於基地台並非對話中的端點，因此無法觀測到應用層(Application Layer)的資訊，需以封包為對象進行資源分配與

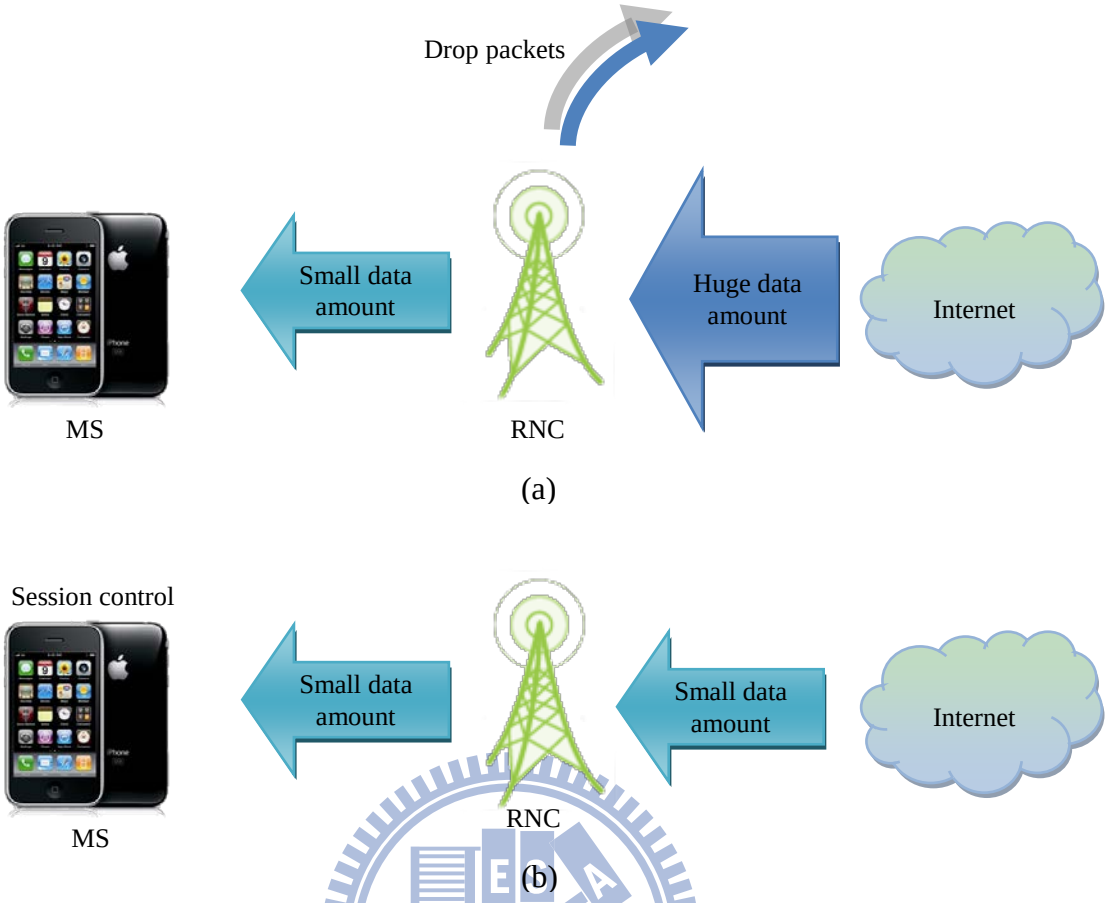

圖 8 – 流量控制(a)於 RNC 上 (b)於手機上

排程的演算法。不過在手機上則可從宏觀的角度處理，如此一來排程方法的設計較為直觀，且應用程式的服務品質調整也更明確。(圖 9)描述了在不同層次中資源分配的概念，在硬體層中負責改變傳輸功率，此事件發生的時間間隔大約是一毫秒左右。而在傳輸層則進行封包的排程，一般來說封包傳輸速率的變化週期大約以十毫秒為最小單位。在最上層中進行的是通訊控制，例如建立與接受連線等操作。

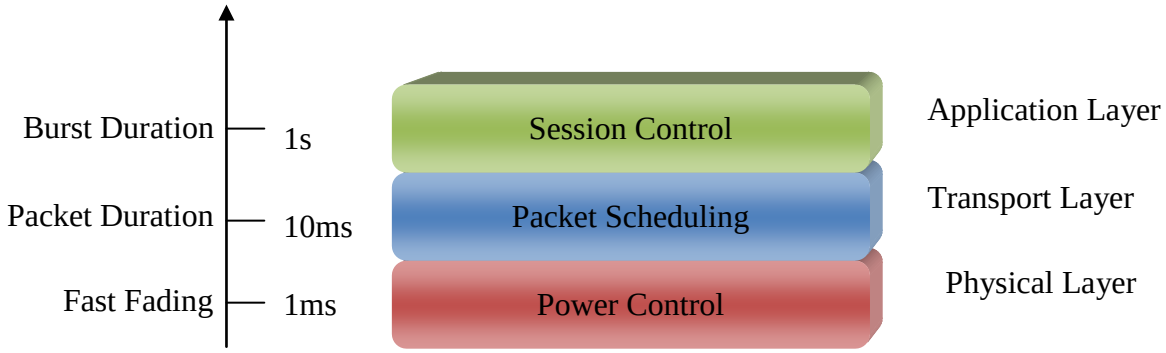

圖 9 – 無線資源分配於不同時間尺度下的控制策略

基於上述的理由，我們將專注於連線層面的排程，而不考慮下層如何傳送單一封包。因為隨時可能有新的要求發生，因此演算法需以線上即時的方式進行。另一方面，我們令各個手機的排程演算法獨立運作，也就是不考慮基地台範圍內其他手機的排程情況。不考慮其他手機流量的原因在於，若是採取合作的模式，不只會造成頻寬的額外通訊負

擔，而且由於網路狀況改變迅速，需要時常改變現有排程，無法達到預想的效能。為了實現 1.3 提出的目標，我們設計了通用的效能函數以及執行效能區域性最佳化(local optimization)的排程演算法。

3.2.2. 假設(Assumptions)

在 3.1 提出的系統模型中在網路架構中加入了手機端代理伺服器和雲端伺服器兩種元件，我們可以此為基礎做一些合理的假設，令其可提供以下功能以及資訊。

- 手機端代理伺服器知道它使用的應用程式列表及其規範，意即代理伺服器知道應用服務對三種服務品質參數的需求，且使用者存取的應用程式總數不會變更。
- 手機端代理伺服器可對基地台發出詢問(query)，取得頻道的頻寬和接收訊號強度，同時它可以運用一些程式化函數觀測與計算獲得現在的最大傳輸速率、封包延遲、封包錯誤率等網路狀況資訊。
- 手機端代理伺服器從應用層控管所有網路連線，包括各個對話的開始時間、傳輸速率、暫停或繼續傳輸等功能，皆可經由程式化函數實現，但是它不接觸下層的封包排程方式和傳輸功率調節函數。
- 手機端代理伺服器會記錄使用者存取應用程式的時間，並能透過統計分析得到存取間隔時間的機率分布。
- 雲端代理伺服器會定期去應用程式伺服器抓取資料，記錄更新時間後分析可能的更新間隔時間機率分布。每當它發現新的應用資料內容時，都會發送一個短通知給手機端代理伺服器。

3.2.3. 效能函數(Utility Functions)

效能函數原本是經濟學中的一個概念，用於量測消費者對商業服務的滿意程度，後來被擴展調整型態後應用於網路資料傳送中[1,2,5,7]，此時它定義的對象變成了用戶對網路服務的滿意度。隨著它用於不同種類的系統模型中，效能函數的內容也需要進行調整。過去量測有線網路效能時[8,27,28]，一般只考慮以頻寬作為函數的輸入，原因在於有線的連線品質較為穩定，其他因素對效能的影響相對較小。但是在無線環境中，可用頻寬相對較小，通訊品質的波動較大，因此應用程式的服務品質無法僅以傳輸速率表示，而需考慮其他網路條件。在我們所提倡的排程演算法中，也是以效能函數最大化為目的，因此如何合理設計效能函數的型式，令其能符合現實中使用者對服務品質的期望，成為一個十分重要的課題。

在 3.1.6 小節中對流量類型進行了分類，並概略描述了對應的效能圖形。這邊開始將基於過去的研究及合理的推導，建構出對應於不同參數的效能模型，接著根據相關的定理將其合併。我們定義服務品質的三個參數是傳輸速率、傳輸潛伏時間、封包錯誤率，基於這三個屬性的效能函數分別以U_T、U_D及U_R表示。

首先我們先定義何謂效能函數，令進行一個決定(decision)所產生的結果可能是$x_1, x_2, \ldots, x_N$，這些x_i可能是純量(scalar)或向量(vector)，且決策者對這些結果有明確的偏好順序，$x_1 \prec x_2 \prec \cdots \prec x_N$，$x_1 \prec x_2$表示對$x_2$的偏好大於$x_1$。這個偏好是相對而非絕對的，當我們希望建立量化的單位時，需先定義其中兩個結果的基準值，再根據基準值去設定其他結果的相對數值。在不失一般性的條件下，我們可建立以下的比例尺。

$$u(x^*) = 1 \quad and \quad u(x^o) = 0$$

x^o表示決策者最不偏好的結果，而x^*表示決策者最偏好的結果，$u(x)$表示定義的偏好。此時對其他可能的結果x來說，可寫成

$$u(x) = \pi \cdot u(x^*) + (1 - \pi)u(x^o)$$

以上式子的意義是說，獲得x的結果等同於有π的機率獲得x^*加上$1-\pi$的機率獲得x^o，在實際情形中，$u(x)$可能很複雜，此時不針對個別x做定義，而找出其擬似曲線(fitting curve)是較好的選擇。

根據過去的研究[1-3, 6, 28]，可以知道對一般性流量來說，增加傳輸速率的邊際效能(marginal utility)會不斷遞減，意義是每單位的資源所能給予的效能將越來越低。在數學上此函數會有以下的性質，單調遞增(monotonic increasing)，和嚴格凸形(strictly concave)。擁有這些性質的函數圖形皆可作為適當的U_T，我們則採用指數圖形來表示:

$$U_T(r) = 1 - e^{-c(r - r_{req})} \text{。} \tag{1}$$

r_{req}是應用程式的傳輸速率需求，c是形狀參數。

對傳輸潛伏時間來說，效能的變化也受到邊際效應的作用[20, 29]，起初延遲增加時，用戶的感受十分明顯，但是過了很長一段時間還沒有收到資料時，那麼效能幾乎不再變動。因此函數性質為單調遞減(monotonic decreasing)，及嚴格凹形(strictly convex)，因此可以用以下函數表達:

$$U_D(d) = e^{\frac{ln(U_{bound})}{d_{req}}d} \text{。} \tag{2}$$

d_{req}是應用程式的延遲需求，U_{bound}表示在延遲是d_{req}時的效能邊界值。

封包錯誤率與用戶效能間的關係又不太一樣，使用者並不會直接因為錯誤率的改變影響其偏好，而是封包錯誤影響了傳輸速率跟傳輸時間，而間接造成滿意度的變化。在這種情況下，封包錯誤率較低時對邊際效能的影響較小，但是當封包錯誤率接近應用程式能忍受的上限時，效能即開始急遽下降。因此函數特性包括了單調遞減(monotonic decreasing)，及嚴格凸形(strictly concave)，可能的形式之一為:

$$U_R(\varepsilon) = 2 - e^{\frac{ln2}{\varepsilon_{req}}\varepsilon} \text{。} \tag{3}$$

ε_{req}是應用程式的封包錯誤率需求。

(圖 10)顯示了一般性流量對應不同參數方面的效能函數，可以觀察到此類型應用程式對於傳輸速率和封包錯誤率有嚴格要求，網路條件無法滿足其需求時，就無法獲得任何效能。但是它對延遲並沒有硬性規範，只說明延遲太高時效能會極端低落。

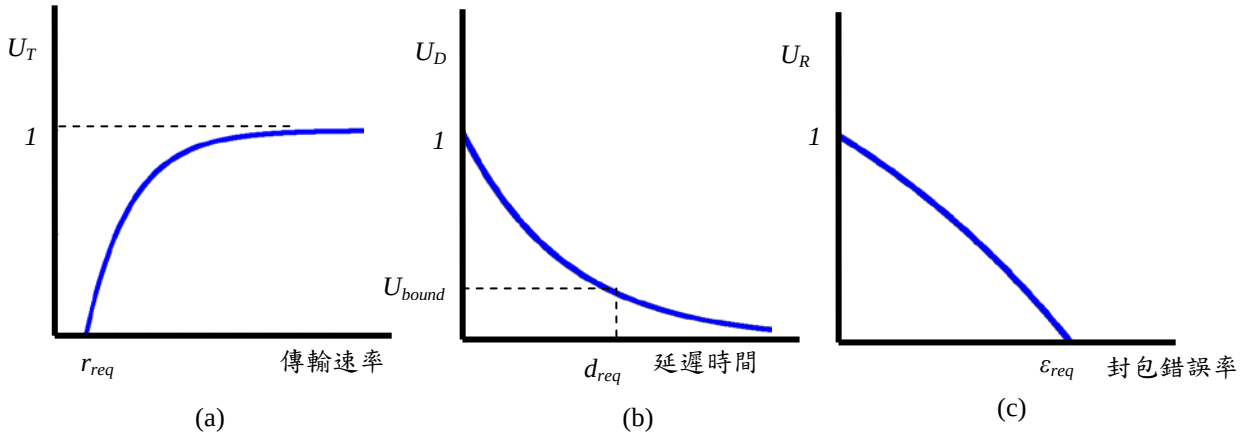

圖 10 – 一般性流量的效能函數:(a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率

接下來我們考慮時間重要性的流量類型，過去的研究[20, 29, 30]說明了此類型流量的特性。對於頻寬並沒有特殊要求，但是當延遲超過其規範的訴求，服務表現就會變的很差。根據以上條件，我們可以選擇傳輸速率與封包錯誤率跟一般性流量相同的函數形狀，而針對延遲層面的函數形狀，我們希望有以下數學性質，單調遞減(monotonic decreasing)，並與 x 軸相交。基於這些特性，我們提出了線性的效能模型:

$U_D(d) = 1 - \frac{d}{d_{req}}$。 (4)

(圖 11)顯示了時間重要性流量可能的函數圖形，相較於一般性流量，差異只在延遲造成的效能變化。隨著時間推遲越久，兩種流量的效能都會隨之降低，但是對一般應用程式來說只會變的很低而不會變成零，它對延遲時間的需求事實上屬於軟性規範(soft constraint)。對注重時效性的應用程式如股票來說，不能及時取得資料對用戶造成的影響是相當大的，因此需要明確的硬性規定(hard constraint)告知伺服器其需

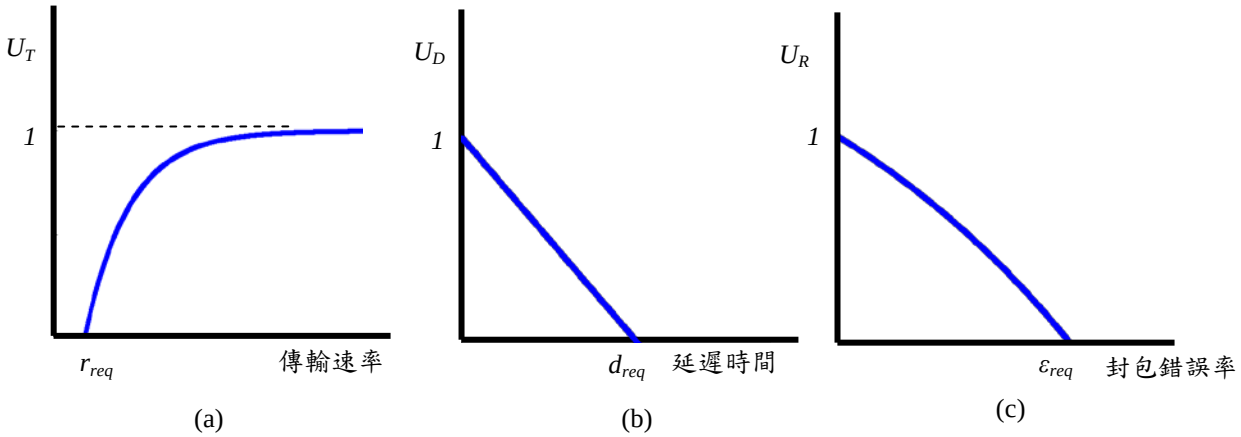

圖 11 – 時間重要性流量的效能函數:(a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率

求。

在我們分類中的最後一種流量是品質調整性，顧名思義這種類型的應用程式可以根據網路狀況的改變而選擇不同的服務品質，像是 YouTube 影片跟 Google Earth 的圖像皆有此種功能，讓使用者根據可用頻寬來決定品質。因此應用程式有多個可能的速率需求，但是對延遲跟封包錯誤率的需求只有一種，原因在於資料類型並沒有改變，所以仍有相同的延遲跟封包錯誤率需求。概念上來說，品質調整性流量可以視做多個一般性流量的集合，因此傳輸速率的效能函數也可以疊合方式來計算，此時它的數學性質變為: 單調遞增(monotonic increasing)和部分凸形(quasi concave)，我們以多邏輯函數(multi-logistic function)作為模型:

$$U_T(r) = \frac{1}{n}\sum_n 1/(1+e^{-r+r_{reqn}}) \ , r_{req1} < r_{req2} < \cdots < r_{reqn} 。 \quad (5)$$

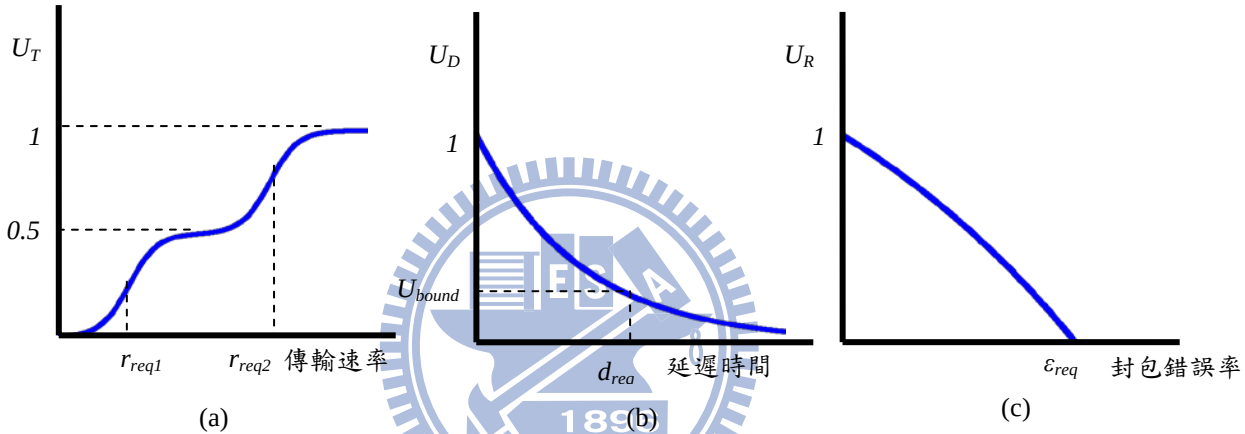

圖 12 – 品質調整性流量的效能函數: (a) 傳輸速率 (b) 延遲時間 (c) 封包錯誤率

當傳輸速率介於兩個需求之間時，使用者獲得的效能只能根據較低的需求計算，因此函數中有較明顯的階層差，在速率能滿足較高的需求時，效能才會有顯著的提升，如同(圖 12)所示。

到這邊為止我們已經將所有流量類型的U_T、U_D及U_R都定義完成，但是這是考慮單一參數的情況下所得出的效能值。實際上用戶效能會同時受到這三個品質參數的影響，因此我們必須找出它們之間的關聯性並建立一個整合的效能函數表示形式，如同 $U(r,d,\varepsilon) = f(U_T(r), U_D(d), U_R(\varepsilon))$。
f是將各個單一參數的效能函數進行合成的方式，例如以加法(additive)或是乘積(multiplicative)的形式進行，接下來我們將根據多屬性效能定理(Multi-Attribute Utility Theorem)中提倡的標準方式，以數學推導出一個合理的效能描述方法。

多屬性效能定理要成立的基本的前提是效能獨立(utility independent)，意思是說決策者對於一個參數X的偏好，不會受到另一個參數Y值的影響。其數學定義如下。

定義 1. 若決策者對X在給定Y值下的條件偏好(conditional preference)，不受到Y的

影響，則X效能獨立於Y。

$\pi \cdot u(x_1, y) + (1-\pi) \cdot u(x_2, y) = u(x_3, y)$

$\Rightarrow \pi \cdot u(x_1, y') + (1-\pi) \cdot u(x_2, y') = u(x_3, y')$

實際情況中傳輸速率、延遲時間、封包錯誤率這三個參數的效能關聯性可能無法完全滿足效能獨立的概念，但是根據多屬性效能定理，我們知道在多數情形中可以直接假設效能獨立滿足於兩兩參數之間。理由在於只要選擇適當的函數型式，就足以良好的模擬原始效能函數，並達到同樣的效果。進一步可應用許多可能的數學方法，對問題的解決有很大的助益。

定義 2. 基於效能獨立的假設下， $u(x,y,z) = f(u_X(x), u_Y(y), u_Z(z))$且有以下性質

i. $u(x^o, y^o, z^o) = 0$且$u(x^*, y^*, z^*) = 1$

ii. $u_X(x), u_Y(y), u_Z(z)$的值皆落在[0,1]的區間

iii. 可寫成$u(x,y,z) = k_x u_X(x) + k_y u_Y(y) + k_z u_Z(z) + k_{xy} u_X(x) u_Y(y) + k_{xz} u_X(x) u_Z(z) + k_{yz} u_Y(y) u_Z(z) + k_{xyz} u_X(x) u_Y(y) u_Z(z)$

從定義 2. 中可以看出$u(x,y,z)$中的項次可能有一次項還有乘積項，一次項如$k_x u_X(x)$表示它對總效能有獨立的影響力，而乘積項如$k_{xy} u_X(x) u_Y(y)$則表示$u_X(x)$對$u(x,y,z)$的影響，與$u_Y(y)$的值有關。為了確定其適當的函數型式，需要引進另一個概念。

定義 3. 若X與Y效能獨立，且滿足$u(x,y) + u(x',y') = u(x',y) + u(x,y')$，則稱參數$X$和$Y$為偏好獨立(preferentially independent)。

當兩個參數X和Y為偏好獨立時，意味著總效能與(x,y)的組合情形無關，而只受到x和y個別的值所影響。

定義 4. 在多參數效能函數中，若兩兩參數皆為偏好獨立，則函數型式可用完全加性(additive)表示。若兩兩參數皆非偏好獨立，則可用完全積性函數(multiplicative)表示。

我們針對服務品質需求的定義，指的是對每個應用程式來說，皆有傳輸速率、傳輸潛伏時間、封包錯誤率的最低需求。當傳送應用資料時，若任一要求無法被滿足，用戶即無法獲得對應的品質。現在可以針對定義 2. 的標準效能函數形式求得各項的係數，令$U_T(r^*) = 1, U_D(d^o) = 0, U_R(\varepsilon^o) = 0$，則

$U(r^*, d^o, \varepsilon^o) = k_x$

$U(r^*, d^o, \varepsilon^o)$表示傳輸速率是用戶最偏好的狀態，延遲和封包錯誤率則為最不偏好的狀態。根據我們對服務品質的定義，這種狀況下應用程式無法給予用戶可行的品質，即獲得效能為零，因此$k_x = 0$。以同樣的方式可知兩兩參數皆非偏好獨立，則根據定義 4. ，合併後的效能函數可表示成

$$U(r,d,\varepsilon) = U_T(r) \cdot U_D(d) \cdot U_R(\varepsilon) \quad (6)$$

當應用程式的傳輸協定不同時，同樣的網路品質狀況可實現的用戶效能也會不一樣。例如應用程式採用 UDP 通訊協定時，封包有錯誤時不需重傳，因此傳輸速率即為資料的有效傳輸速率(effective throughput)，則效能函數可直接表示成

$$U(r,d,\varepsilon) = U_T(r) \cdot U_D(d) \cdot U_R(\varepsilon)$$

但是若以 TCP 協定進行傳輸，則部分頻寬會用於重傳機制，因此有效傳輸速率會低於其分配到的頻寬，此時效能函數的型式應為

$$U(r,d,\varepsilon) = U_T(\hat{r}) \cdot U_D(\hat{d}) \cdot U_R(\varepsilon)$$

$$\hat{r} = r(1-\varepsilon-\varepsilon^2-\cdots) \cong \mathrm{r}(1-\frac{\varepsilon}{1-\varepsilon})$$

$$\hat{d} = (T_c - C) + \frac{\tilde{M}}{\hat{r}}$$

T_c表示現在時間，C表示工作建立時間，$\tilde{M}$則是剩餘資料大小。

3.2.4. 快取機制(Caching Mechanisms)

當一個應用程式的新資料被下載至手機上時，代理伺服器會將其副本放到至本地的快取儲存區中，快取的可用期限則與應用程式的更新週期相關聯。位於雲端的代理伺服器會定期的查看應用程式伺服器的狀態，統計分析資料的變化情形以了解應用程式的更新頻率。雲端代理伺服器會將此資訊傳給有使用此應用程式的手機端代理伺服器，如此則可以對快取設定其可用的期限。每當用戶使用某個應用程式時，如果快取中有此資料且尚未過期，則可直接回傳給使用者而避免耗費頻寬傳送重複的內容。如果應用資料的延遲時間過長，超過其可接受的期限(deadline)，那麼代理伺服器就會判斷將快取中較舊的資料給使用者。但是當應用程式屬於時間重要性的流量類型，例如股票跟新聞時，代理伺服器並不會儲存其快取資料，而是在用戶發送資料要求時才向伺服器下載。採用這種方式的原因在於股票與新聞本身的特性，過期的內容對使用者來說完全沒有價值可言，只有即時下載才能保證資料的新鮮度。

3.2.5. 應用程式內容預先下載(Application Pre-fetching)

行動網路中的流量在不同時間的差異十分大，大多數人有類似的作息因此傾向於在差不多的時間存取服務。這種現象會造成壅塞的發生，網路傳輸速率緩慢，服務品質下降等情形。在基地台或是路由器上遇到此種狀況時，只能選擇將過多的封包丟棄。如果手機可以在空閒的時候先下載所需的應用資料至本機端儲存，則可有效改善壅塞的問題。因此代理伺服器除了可以替用戶的要求做排程之外，還可以自動產生工作執行，如此則能分散各個連線傳輸的時間，使整體效能獲得提升。對使用者來說，則有最小化回應時間的好處。需要注意的是預先下載的資料沒有被使用者存取的話，反而變成網路的額外負擔。因此我們運用一個簡單的策略來決定是否該進行預先下載的操作，考慮此決定的時間點只發生在雲端代理伺服器告知應用程式內容更新時。令使用者存取一個應用程式

的間隔時間機率分佈為$P_{access}(t)$，而此應用程式更新的間隔時間機率分佈為$P_{update}(t)$，這兩種分佈資訊皆可由代理伺服器分析歷史紀錄而取得。預先下載機制能發揮作用的情況，是使用者在應用程式再度更新前就存取其內容。假設用戶上次存取此應用程式的時間是x，現在的時間點即應用程式更新時間是u，應用程式下次更新的時間點為u'，則在u'前存取的機率是

$$P_{access}(u-x \le t < u'-x) = \int_{u-x}^{u'-x} P_{access}(t)dt \quad 。 \tag{7}$$

考慮所有可能的u':

$$P_{hit} = \int_{u}^{\infty} P_{access}(u-x \le t < u'-x) \cdot P_{update}(u'-u)du'$$

$$= \int_{u}^{\infty} \int_{u-x}^{u'-x} P_{access}(t) \cdot P_{update}(u'-u)dtdu' 。 \tag{8}$$

P_{hit}為預先下載的資料被存取的機率，則對網路來說額外資料量的期望值為

$$M_{prefetch} = M_i(1-P_{hit}) 。 \tag{9}$$

當$M_{prefetch}/B_{max} < \varphi_{data}$時，表示對網路的額外負擔在可接受的程度內，就可以建立預先下載的工作放入待執行工作佇列中。

在 3.1.8 中針對排程演算法中的工作規格進行了詳細的說明，預先下載的工作同樣也有這些參數，但是設定方式跟一般工作有很大的差異。原本優先權為p_i的應用程式，其預先下載的工作優先權將被設成$p_i + N_{app}$，這保證了一般工作會先被執行，而在所有預先下載的工作之間，仍維持著使用者定義的偏好順序。另一方面，預先下載的工作並非直接傳送給用戶即時使用，因此對服務品質不需有嚴格的要求。我們將其傳輸速率的需求設為零，傳輸延遲的需求設為應用程式再度更新的時間點，封包錯誤率的需求設為一。

3.2.6. 排程策略(Scheduling Strategies)

行動網路中的狀況例如可用頻寬、訊號強度、封包錯誤率會隨著時間而連續不斷地改變。但是實際進行演算法時，無法完全針對所有可能的變化去調整現在的排程，因為如此會對代理伺服器帶來大量的額外負荷，而所獲得的結果也不一定能明顯改善。因此我們每隔τ_p的時間觀測及計算網路參數的變化，並假設在τ_p這段時間內排程演算法需考慮的網路參數皆不會發生改變，因此τ_p即為排程的週期。排程演算法的核心概念，是希望找出適當的工作傳輸方式以最大化使用者所獲得的效能函數值，當等待執行的工作有N個時，可以表示成以下的最佳化問題:

最大化$\sum_{i=1}^{N} W(p_i) \cdot U_i(r_i, d_i, \varepsilon)$，

限制條件為

$0 \le \sum_{i=1}^{N} r_i \le r_{max}$ ，

$d_i = (T_c - C_i) + \widetilde{M_i}/r_i \quad for\ \forall i$。

$W(p_i) = 1/(p_i \cdot (1 - \frac{T_c - C_i}{d_{i,req}}))$

$U_i(r_i, d_i, \varepsilon)$是第$i$個工作的效能函數，注意效能函數會因流量類型以及服務品質需求的差別而有不同形狀。加權函數$W(p_i)$會隨著p_i值越小，即優先順序越高時而增加，並隨著

時間越來越接近工作的期限而增加。T_c表示現在時間，C_i是工作建立時間，$\widetilde{M_i}$是資料剩餘大小。p_i是用戶偏好而ε是封包錯誤率，ε的值與工作互相獨立只與現在網路狀況有關，這兩者皆非排程時可以改變的因素。d_i會受到r_i的影響而變化，因此我們在排程方法中考慮的問題簡化成找出一組適當的$(r_1, r_2, \dots, r_N)$。後面提到效能函數時，若無特別說明，則$U_i(r_i, d_i, \varepsilon)$和$U_i(r_i)$表同一概念。此問題可用拉格朗日乘數法(lagrangian multiplier)來表示，意即對以下方程式組求解。

$$\sum_{i=1}^{N} r_i - r_{max} = 0 \tag{10}$$

$$\frac{d}{dr_i} \cdot \sum_{i=1}^{N} W(p_i) \cdot U_i(r_i, d_i, \varepsilon) = \frac{d}{dr_i} \lambda \cdot \left(\sum_{i=1}^{N} r_i - r_{max}\right) \quad , \forall i \tag{11}$$

等式(11)可改寫成下列的式子。

$$W(p_i) \cdot U_i'(r_i) = \lambda \quad , \forall i \tag{12}$$

我們可找到其解為

$$r_i = (U_i')^{-1}\left(\frac{\lambda}{W(p_i)}\right) \quad , \forall i \tag{13}$$

有$N+1$個式子，理論上可以解出$r_1, r_2, \dots, r_N, \lambda$，但是由於效能函數由三個部分組成，要計算微分及反導數所耗費的複雜度太高，無法在實務中運用，因此我們採用了試探性(heuristic)原則來做區域性的最佳化，以嘗試接近於最佳解。

引理 1. *當工作列表不變動時，在τ_p時間內的效能最大化方式唯一，意即存在唯一一組的$(r_1, r_2, \dots, r_N)$，使$\sum_{i=1}^{N} W(p_i) \cdot U_i(r_i, d_i, \varepsilon)$最大。*

說明. 基於我們前面的假設，在τ_p時間內網路參數皆不變化，會改變的只有剩餘資料大小。我們可以明顯看出在$(U_i')^{-1}\left(\frac{\lambda}{W(p_i)}\right)$這個值中，並不受到資料大小的影響，因此總效能也不會變化。

除了每次到達排程週期τ_p的時間需要進行重新計算外，每當有工作完成或是新工作加入時，也需要重新考慮資源的利用及排程，以得出新的效能最大方式。我們將演算法的執行分成兩個部分，工作選擇階段(job selection phase)和頻寬分配階段(bandwidth allocation phase)。在工作選擇階段，會依據用戶偏好跟可獲取的效能來決定工作應執行的順序。在頻寬分配階段，則考慮如何分配頻寬給各個工作可獲得最大利益，同時試著將流量分散至不同時間，以減少網路負擔。

我們根據系統中工作執行的狀態不同，將所有工作分成三個集合(set)。一個新工作剛加入時會進入等待集合(waiting set)，正在進行傳輸的工作屬於活躍集合(active set)，已經傳輸部份資料而現在暫停的工作屬於不活躍集合(inactive set)。在(圖 13)中可以看到工作如何在狀態間移動，藍色的狀態表示此工作還在系統中待處理，紅色則表示已離開系統，可能是完成或是被丟棄。

在工作選擇階段時，我們對三個集合中的工作皆計算其潛在效能值(potential utility value)。潛在效能值指的是在給定工作最大可用頻寬r_{max}時，使用者所能得到

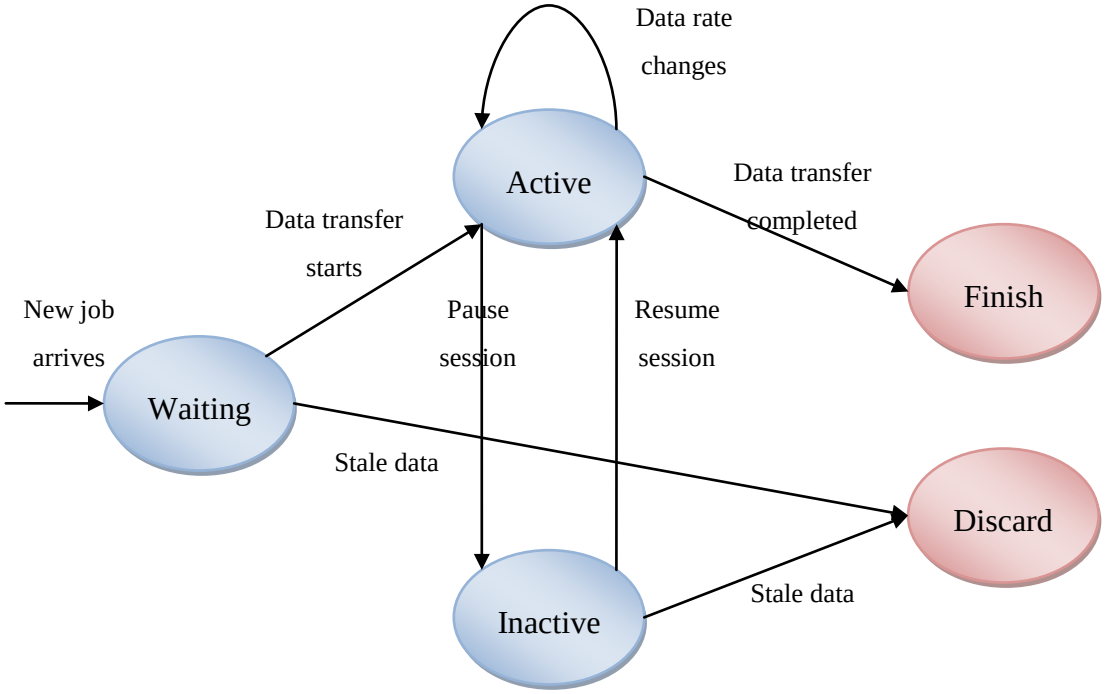

圖 13 – 工作狀態變化示意圖

的效能，也就是根據$W(p_i)\cdot U_i(r_{max})$做排序。有了工作的潛在效能次序後，並不直接根據此順序來執行工作，而是進一步考慮其傳輸親和性(transmission affinity)。傳輸親和性指的是正在傳送資料的連線，會傾向於保持在活躍的狀態，意即對使用者來說，連續且平穩的資料下載能提高用戶體驗。因此我們提出了狀態變化的門檻值(threshold)，等待和不活躍集合中的工作效能比活躍集合的工作效能高過門檻，才能搶先活躍工作執行。基於傳輸親和性調整順序的演算法如下。

Let $job_1,\ldots,job_N$ *be the job list in descending order of potential utility value*
$For(i=1\sim N)$
{ *If(* job_i *belongs to waiting or inactive set)*
$For(j=i+1\sim N)$
If(job_j *belongs to active set)*
$If\left(W(p_i)\cdot U_i(r_i,d_i,\varepsilon)-W(p_j)\cdot U_j\left(r_j,d_j,\varepsilon\right)<U_{threshold}\right)$
Switch the positions of job_i *and* job_j
}

我們以(圖 14)做為例子來說明以上演算法的實際運作情形，左邊是根據潛在效能值排序完的工作順序，右邊是考慮傳輸親和性進行調整後的結果。這種方式有兩個優點，其一是將資源集中於活躍的工作，避免許多工作都只傳輸了部分資料的狀況，其二是減少工作時常在狀態中切換(switch)，造成傳輸品質不佳的問題。

知道了工作執行順序後，接著我們選擇可滿足

$$\sum_{i=1}^{N} W(p_i) \cdot U_i\left(\frac{R_{max}}{N}\right) / \sum_{i=1}^{N} W(p_i) > \Omega_{utility} \quad (14)$$

此式子的最大N值，以大略估計可以同時服務的工作數量。$\sum_{i=1}^{N} W(p_i)$是同時執行N個工

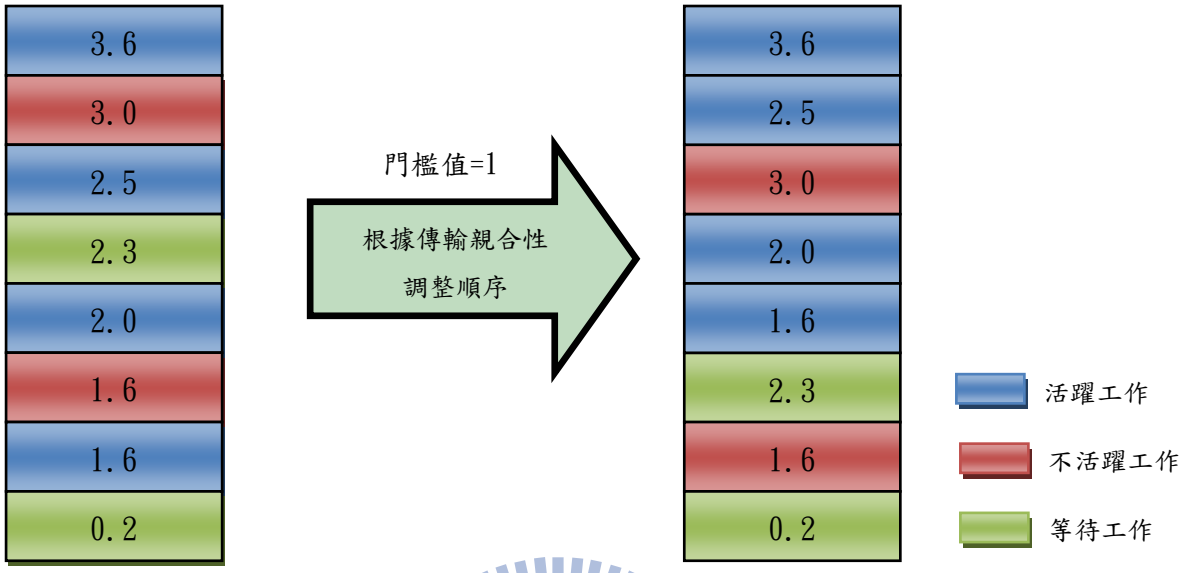

圖 14 – 基於效能及傳輸親合性決定工作執行的順序

作時的效能上限，$\Omega_{utility}$是獲得效能比率。當每個工作分到$\frac{R_{max}}{N}$的頻寬時，所獲得的效能$\sum_{i=1}^{N} W(p_i) \cdot U_i\left(\frac{R_{max}}{N}\right)$高於效能下限$\Omega_{utility}$時，我們選擇擁有最大效能的$job_1, \ldots job_N$執行，到此為止演算法的第一階段即結束。

引理 2. *當$r_{max} \gg \sum_{i=1}^{N} r_{i,req}$時，選擇盡量多的工作執行可獲得較高的用戶效能。*

說明. 每個工作的效能值在考慮其加權幅度時，範圍皆落在$[0, W(p_i)]$之間，當有N個工作時，可能的效能範圍則是$[0, \sum_{i=1}^{N} W(p_i)]$，因此在$r_{max}$足夠大時，同時傳輸越多工作所獲得的效能越顯著。

接著進入到頻寬分配階段，開始考慮如何將可用頻寬分配給$job_1, \ldots, job_N$，使得總效能可達到最大值。前面提過此問題可寫成拉格朗日乘數的方程式組，但是在實際應用時不可行。因此我們根據邊際效能進行區域性搜尋(local search)，找出可能的解。邊際效能的定義是對工作增加每單位頻寬所能獲得的效能，可用下面的式子表示

$$\tilde{U}_i(r_i) = \frac{W(p_i) \cdot U_i(r_i+1) - W(p_i) \cdot U_i(r_i)}{r_i + 1 - r_i} = W(p_i) \cdot [U_i(r_i + 1) - U_i(r_i)] \quad (15)$$

首先令每個工作至少都擁有要求的速率$r_{i,req}$，以此為起點計算$\tilde{U}_i(r_{i,req}) = W(p_i) \cdot [U_i(r_{i,req} + 1) - U_i(r_{i,req})]$。選擇擁有最大$\tilde{U}_i$的工作，將其傳輸速率提升至$r_{i,req} + 1$。反復進行以上計算，直到各個工作的速率總合已經達到可分配上限r_{max}。此演算法可彙整如下。

for each job_i

$r_i = r_{i,req}$

Do

{ $for\ each\ job_i$

$\widetilde{U}_i(r_{i,req}) = \frac{W(p_i)\cdot U_i(r_i+1) - W(p_i)\cdot U_i(r_i)}{r_i+1-r_i} = W(p_i)\cdot[U_i(r_i+1) - U_i(r_i)]$

$Choose\ job_j\ with\ largest\ \widetilde{U}_j$

$r_j = r_j + 1$

}

$Until(\sum_{i=1}^{N} r_i = r_{max})$

引理 3. *根據邊際效能進行的區域性搜尋，不會卡在區域最大值(local maximum)上。*

證明. 觀察$W(p_i)\cdot U_i(r_i, d_i, \varepsilon)$的性質，在改變傳輸速率時，此式子可化簡成

$$W(p_i)\cdot U_i(r_i, d_i, \varepsilon) = W(p_i)\cdot U_T(r_i)\cdot U_D(d_i)\cdot U_R(\varepsilon) = c_1\cdot U_T(r_i)\cdot U_D(d_i) \quad (16)$$

$$= c_1\cdot U_T(r_i)\cdot U_D'(r_i) \quad (17)$$

因$W(p_i)$和$U_R(\varepsilon)$不會變動，可用常數c_1表示。

根據 3.2.3 小節中的定義，$U_T(r_i)$皆為單調遞增，而$U_D(d_i)$皆為單調遞減圖形。延遲與傳輸速率有成反比的關聯性，因此可將$U_D(d_i)$改寫成$U_D'(r_i)$，此時$U_T(r_i)\cdot U_D'(r_i)$有單調遞增的性質。則以$(r_1, r_2, \dots, r_N)$為參數的多維效能圖形，不存在區域最大值。

將概念性的圖形於空間中表示時，此函數大略如(圖 15)所示。縱軸表示$\sum_{i=1}^{N} W(p_i)\cdot U_i(r_i)$，水平各軸表示$(r_1, r_2, \dots, r_N)$，黑色虛線框框表示各個工作的最低傳輸速率需求$(r_{1,req}, r_{2,req}, \dots, r_{N,req})$。我們所採用的區域性搜尋是以最低傳輸速率需求為起點，根據邊際效能逐步增加各個維度的r_i值，直到紅色虛線框框的範圍，也就是邊界條件$\sum_{i=1}^{N} r_i = r_{max}$為止。

引理 4. *根據邊際效能進行的區域性搜尋，在所有工作的$U_i(r_i)$皆有單調遞增(monotonic increasing)和嚴格凹形(strictly concave)的性質時，可找出其全域最大值(global maximum)。*

證明. 令基於邊際效應進行的區域性搜尋找出的解為$(r_1, r_2, \dots, r_N)$，假設此非最佳解，意即有另解$(r_1', r_2', \dots, r_N')$，使得$\sum_{i=1}^{N} W(p_i)\cdot U_i(r_i') > \sum_{i=1}^{N} W(p_i)\cdot_i (r_i)$。不失一般性，我們令$r_1' = r_1 + k$，$r_2' = r_2 - k$，而在$i \geq 3$時$r_i' = r_i$。則

$$\sum_{i=1}^{N} W(p_i)\cdot U_i(r_i') > \sum_{i=1}^{N} W(p_i)\cdot U_i(r_i)$$

$$\Rightarrow W(p_1)\cdot U_1(r_1') + W(p_1)\cdot U_2(r_2') > W(p_1)\cdot U_1(r_1) + W(p_2)\cdot U_2(r_2)$$

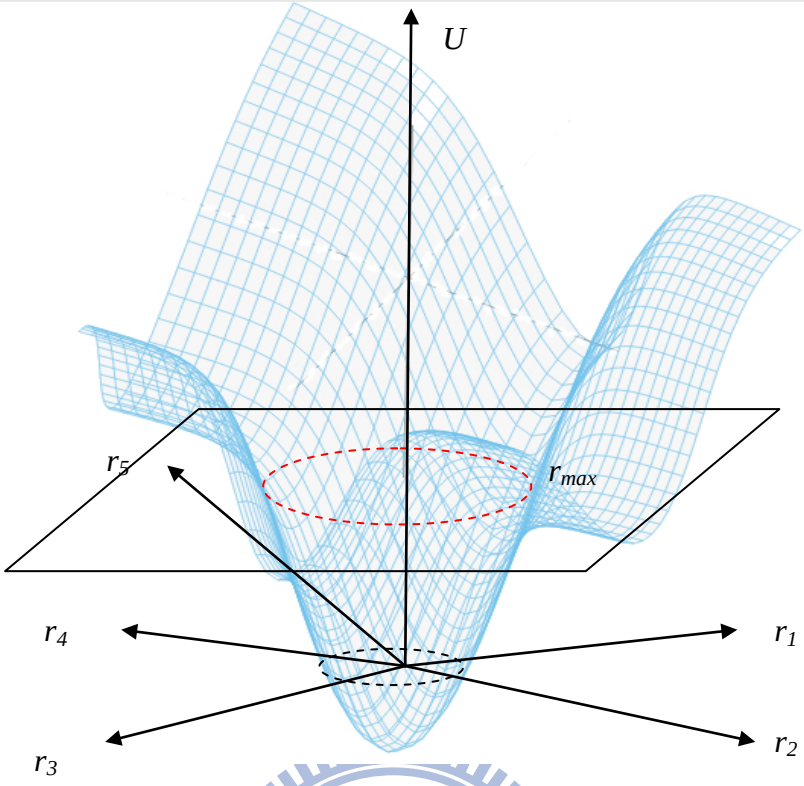

圖 15 – 以$(r_1, r_2, \dots, r_N)$為參數的效能概念圖

$$\Rightarrow W(p_1)\cdot(U_1(r_1')-U_1(r_1)+U_1(r_1))+W(p_2)\cdot U_2(r_2')$$
$$> W(p_1)\cdot U_1(r_1)+W(p_2)\cdot(U_2(r_2)-U_2(r_2')+U_2(r_2'))$$

$$\Rightarrow W(p_1)\cdot(U_1(r_1')-U_1(r_1)) > W(p_2)\cdot(U_2(r_2)-U_2(r_2')) \quad (18)$$

則在$(r_1, r_2', \dots.)$此位置時，由於已知區域性搜尋的解為$(r_1, r_2, \dots, r_N)$，且$r_2 > r_2'$。因此$r_2', r_2'+1, \dots, r_2'+k-1$各點的邊際效能皆會高於$r_1$此位置的邊際效能，即

$$W(p_2)\cdot[U_2(r_2'+m)-U_2(r_2'+m-1)] > W(p_1)\cdot(U_1(r_1+1)-U_1(r_1)), 1\le m\le k$$

$$\Rightarrow \sum_{m=1}^{k} W(p_2)\cdot[U_2(r_2'+m)-U_2(r_2'+m-1)] > k\cdot\{W(p_1)\cdot(U_1(r_1+1)-U_1(r_1))\} \quad (19)$$

由於$U_i(r_i)$是嚴格凹形，因此我們可知

$$W(p_i)\cdot[U_i(r_i+1)-U_i(r_i)] > W(p_i)\cdot[U_i(r_i+k)-U_i(r_i+k-1)],\ \forall i,k\ge 1 \quad (20)$$

根據(14)的不等式，可將(13)改寫成

$$\sum_{m=1}^{k} W(p_2)\cdot[U_2(r_2'+m)-U_2(r_2'+m-1)]$$
$$> \sum_{m=1}^{k} W(p_1)\cdot[U_1(r_1+m)-U_1(r_1+m-1)]$$

$$\Rightarrow W(p_2)\cdot\big(U_2(r_2'+k)-U_2(r_2')\big) > W(p_1)\cdot\big(U_1(r_1+k)-U_1(r_1)\big)$$

$$\Rightarrow W(p_2)\cdot\left(U_2(r_2)-U_2(r_2')\right)>W(p_1)\cdot\left(U_1(r_1')-U_1(r_1)\right) \quad (21)$$

(15)與(12)矛盾，因此$(r_1,r_2,\ldots,r_N)$為全域最大值。

引理 5. *根據邊際效能進行的區域性搜尋，在給定執行的工作數量N個，且當中不包括品質調整性流量時，可找出全域最大值。*

證明. 根據(16)，效能函數在進行區域性搜尋時可寫成以下形式

$$W(p_i)\cdot U_i(r_i,d_i,\varepsilon)=W(p_i)\cdot U_T(r_i)\cdot U_D(d_i)\cdot U_R(\varepsilon)=c_1\cdot U_T(r_i)\cdot U_D(d_i)$$

在不喪失其形狀跟數學性質的條件下，可令$d_i=1/r_i$

$=c_1\cdot U_T(r_i)\cdot U_D(1/r_i)$

對不同流量類型來說$U_T(r_i)$和$U_D(1/r_i)$的型式為

$$U_T(r_i)=\begin{cases}1-e^{-c(r_i-r_{i,req})} & for\ ordinary\ traffic\\ 1-e^{-c(r_i-r_{i,req})} & for\ time-critical\ traffic\\ \frac{1}{n}\sum_n 1/(1+e^{-r_i+r_{i,reqn}}) & for\ quality-scalable\ traffic\end{cases},$$

$$U_D(1/r_i)=\begin{cases}e^{\frac{ln\,(U_{bound})}{r_i\cdot d_{req}}} & for\ ordinary\ traffic\\ 1-\frac{1}{r_i\cdot d_{req}} & for\ time-critical\ traffic\\ e^{\frac{ln\,(U_{bound})}{r_i\cdot d_{req}}} & for\ quality-scalable\ traffic\end{cases}$$

一般性流量和時間重要性流量的$U_T(r_i)$和$U_D(1/r_i)$以r_i作為參數時，皆有單調遞增且嚴格凹型的函數特性，因此合併後$W(p_i)\cdot U_i(r_i,d_i,\varepsilon)$仍保持此特性。但是品質調整性流量的$U_T(r_i)$非嚴格凹型，因此合併後只有單調遞增的性質留存。則根據**引理 4**，我們可知只有在執行的工作類型不包含品質調整性流量時，基於邊際效能的區域性搜尋才可找出全域最大值。

完整排程演算法的流程敘述如(圖 16)所示:

i. 在排程週期開始時，代理伺服器讀取當下的網路狀況參數r_{max}及ε，r_{max}是最大可用頻寬，ε是封包錯誤率，以上資訊在排程週期內皆假設不變化。

ii. 進入工作選擇階段，根據潛在效能值跟傳輸親合性對等待、不活躍、活躍三個集合中的工作進行排序，並選擇可同時執行的工作$job_1,\ldots,job_N$以求達到最大效能。

iii. 將$job_1,\ldots,job$ 放入活躍集合中，已經傳輸部分資料的其他工作放入不活躍集合中，剩下的工作則放入等待集合中。

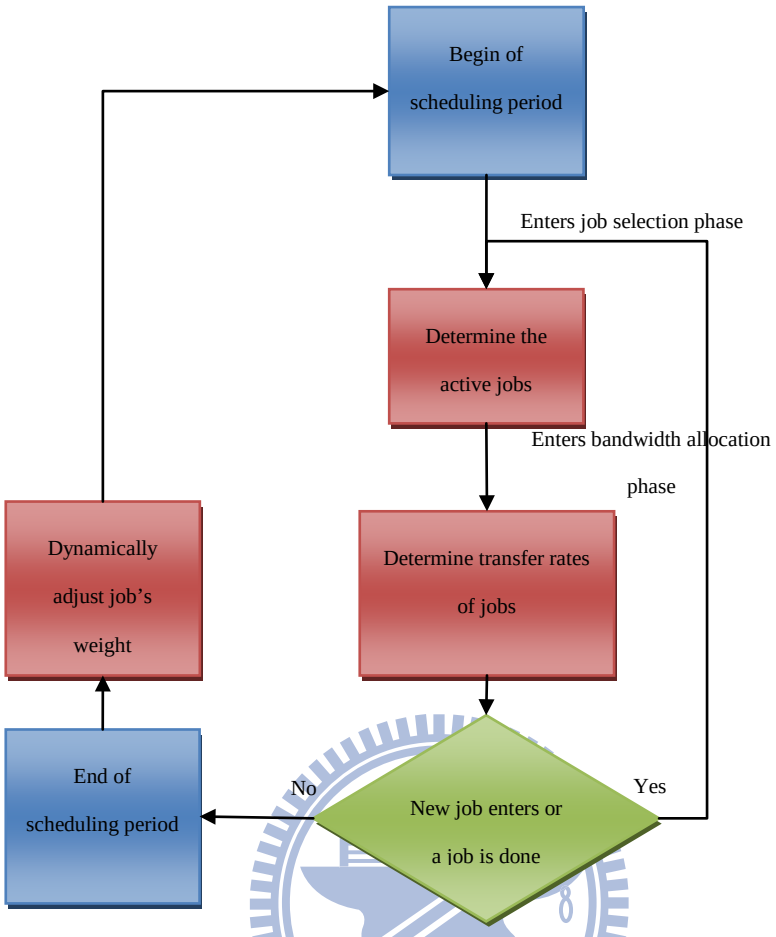

圖 16 – 排程演算法流程圖

iv. 進入頻寬分配階段，以基於邊際效能的區域性搜尋方法，找出N個工作的傳輸速率$(r_1, r_2, \ldots, r_N)$，盡量使總效能$\sum_{i=1}^{N} W(p_i) \cdot U_i(r_i)$達到最大值。

v. 代理伺服器根據前兩個步驟決定的工作和傳輸速率開始傳輸資料。

vi. 當有新工作進入等待列表，或是有活躍工作完成時，回到步驟 ii. 重新決定應執行的工作及其傳輸速率。

vii. 排程週期結束時，隨著越來越接近工作的完成期限而提高其權重，接著回到步驟 i.。

四、 資 料 分 析 與 結 果

4.1 模擬設計(Simulation Model)

4.1.1. 網路狀況模型(Network Status Model)

本篇論文所提倡的方法將透過模擬的方式評估其效果，並對網路狀況進行以下假設。由於無線網路相對於後端的有線網路，在頻寬上受到較大的限制，且連線品質變化幅度也較明顯，因此僅需模擬無線網路中的變化即可決定整體網路的表現。我們不考慮基地台在下層使用的技術，如封包排程方式，頻寬分配策略，因為在設計中代理伺服器是從應用層的角度觀察網路，只能取得根據測量及計算獲得的數值。在手機端上觀察到的網路品質變化情形，我們以馬可夫模型(Markov Model)來表示，並且令與基地台間的連線速率和封包錯誤率作為網路參數，定義了四種可能的狀態(表 2)，其狀態轉移圖(state transition diagram)，則如同(圖 17)所示。

我們假設最大傳輸速率是常態分布變數，而封包錯誤率是對數常態分布變數。當最大傳輸速率較低且封包錯誤率較高時，則表示現在的連線品質屬於很差的狀態。相反地當最大傳輸速率較高同時有著較低的封包錯誤率時，代表連線品質極佳。另外可能的情況是最大傳輸速率跟封包錯誤率兩者皆高，或是最大傳輸速率跟封包錯誤率兩者皆低，這兩種情形則歸類成普通的連線品質。

Link quality / Network parameters	Bad	Normal High	Normal Low	Good
Maximum data rate(μ,σ^2)	Normal $(\mu_{low},\sigma^2_{high})$	Normal $(\mu_{high},\sigma^2_{low})$	Normal $(\mu_{low},\sigma^2_{high})$	Normal $(\mu_{high},\sigma^2_{low})$
packet error rate(μ,σ^2)	Lognormal $(\mu_{high},\sigma^2_{high})$	lognormal $(\mu_{high},\sigma^2_{high})$	lognormal $(\mu_{low},\sigma^2_{low})$	lognormal $(\mu_{low},\sigma^2_{low})$

表 2 - 網路品質狀態及其對應的參數值

在狀態轉移圖中，$P_{I,J}$表示從狀態I轉移至狀態J的機率，每次排程週期開始時，會根據上次週期所處的狀態及轉移機率決定現在的連線品質。狀態轉移只會發生於相鄰的狀態之間，避免連線品質的變化幅度波動太大。

4.1.2. 流量模型(Traffic Model)

本篇論文提出的排程方式是一種個人化的應用層排程演算法，因此只需模擬應用程式的存取間隔時間分布，而不需考慮下層的封包到達時間。根據過去所提出的流量模型[31,32]，我們進行以下假設。

- 用戶對應用程式要求的間隔時間以卜瓦松分布(poisson distribution)來模擬，隨機產生其平均值及標準差，且隨著應用程式類型不同，隨機數值範圍也有差異。

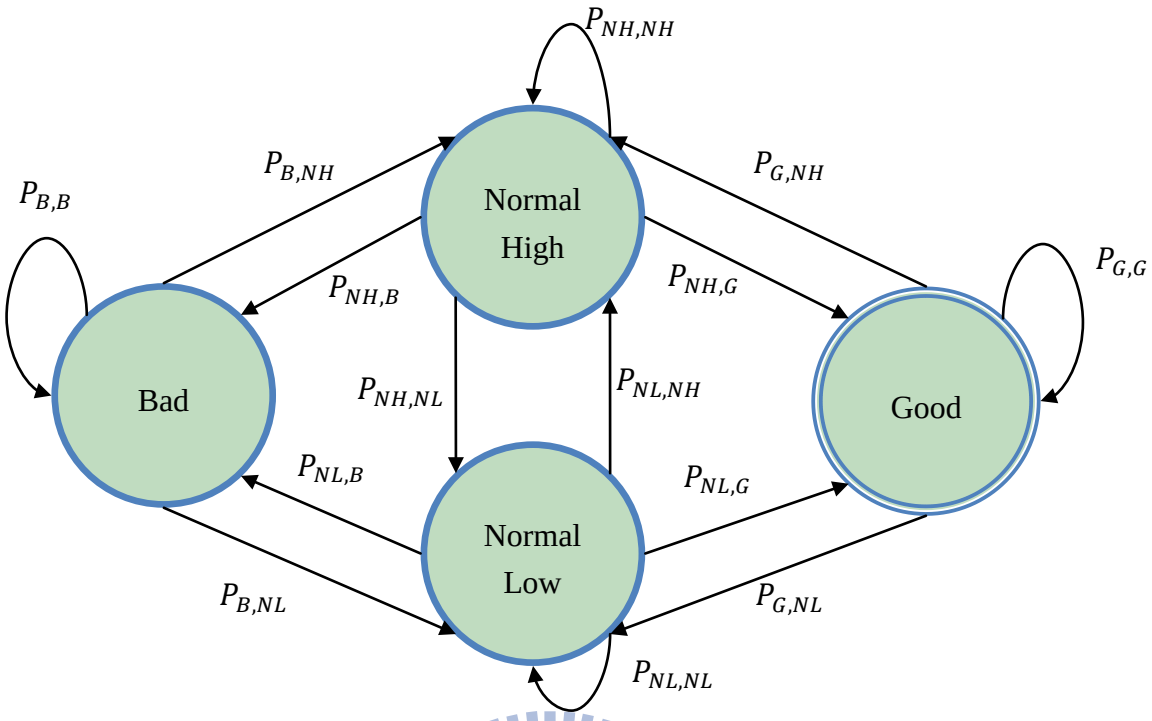

圖 17 – 網路品質狀態轉移圖

- 每種應用程式的資料大小呈現常態分布(normal distribution)，其平均值及標準差隨著應用程式類型而變化，但是與用戶無關。
- 在模擬過程中用戶人數不會改變，用戶使用的應用程式最多七種，以均勻(uniform)方式隨機產生，但在模擬過程中維持不變。
- 用戶對應用程式的偏好亦為均勻方式隨機產生，不同應用程式可能有同樣的偏好值。

4.1.3. 應用程式規範(Application Specifications)

模擬中採用的應用程式列表及其性質如(表 3)所示，每個應用程式的服務品質需求和傳輸協定皆為定值，資料大小為常態分布，更新間隔時間為卜瓦松分布。

規格 / 應用程式	流量類型	傳輸速率需求 (Kb/s)	延遲需求 (s)	封包錯誤率需求	傳輸層協定	資料大小 (μ, σ^2)(Kb)	更新間隔時間 (μ, σ^2)(s)
網頁	一般性	20	30	1%	TCP	(250, 100)	(3600, 80)
電子郵件	一般性	20	30	1%	TCP	(100, 50)	(600, 50)
氣象	一般性	40	30	1%	TCP	(250, 100)	(1200, 50)
新聞	時間重要性	15	20	1%	TCP	(200, 100)	(300, 10)
股票	時間重要性	15	20	1%	TCP	(80, 50)	(60, 10)

Google 地圖	品質調整性	50	40	5%	UDP	(2500,200)	(7200,150)
即時影片	品質調整性	60	40	5%	UDP	(5000,200)	(7200,150)

表 3 – 應用程式規格

4.2 結果分析

4.2.1. 整體網路效能評估

為了評估本研究方法對網路的影響，我們考慮在第一個基地台的涵蓋範圍中，皆為一般用戶，亦即未裝置代理伺服器於手機上。第二個基地台的涵蓋範圍中，皆為擁有代理伺服器的手機用戶。當無線最大傳輸速率為 1000Kb/s，且人數為 20 人時，則這兩個基地台的下載流量如(圖 19)所示，結果顯示了在手機上執行快取及排程機制，可以有效的減少無線網路的總頻寬消耗。在(圖 18)中呈現了流量的統計數據，可以看出最大

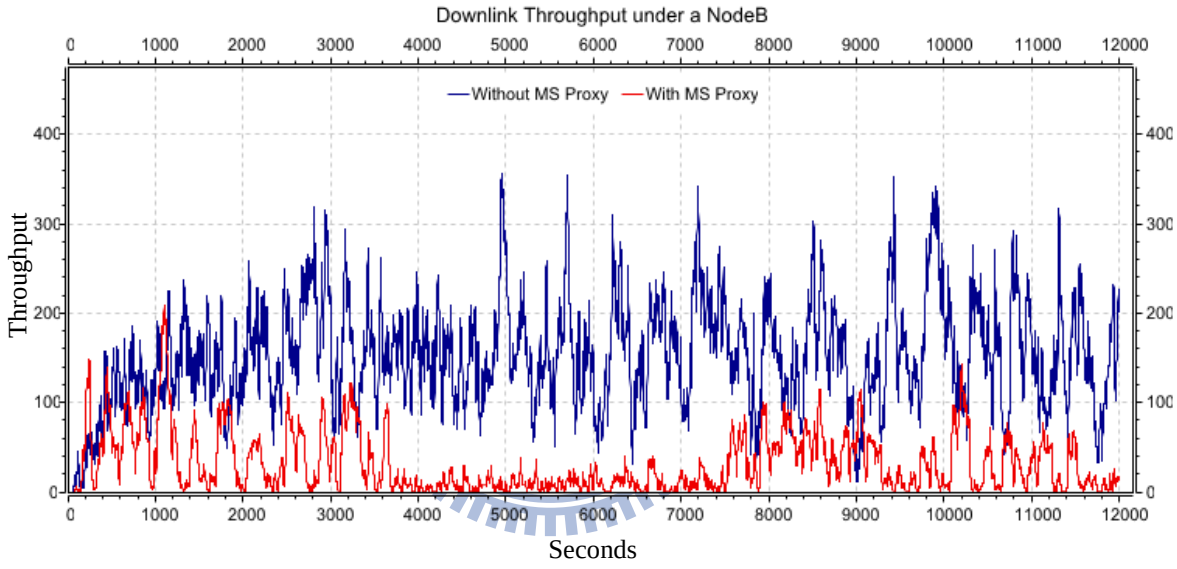

圖 19 – 基地台下行流量比較

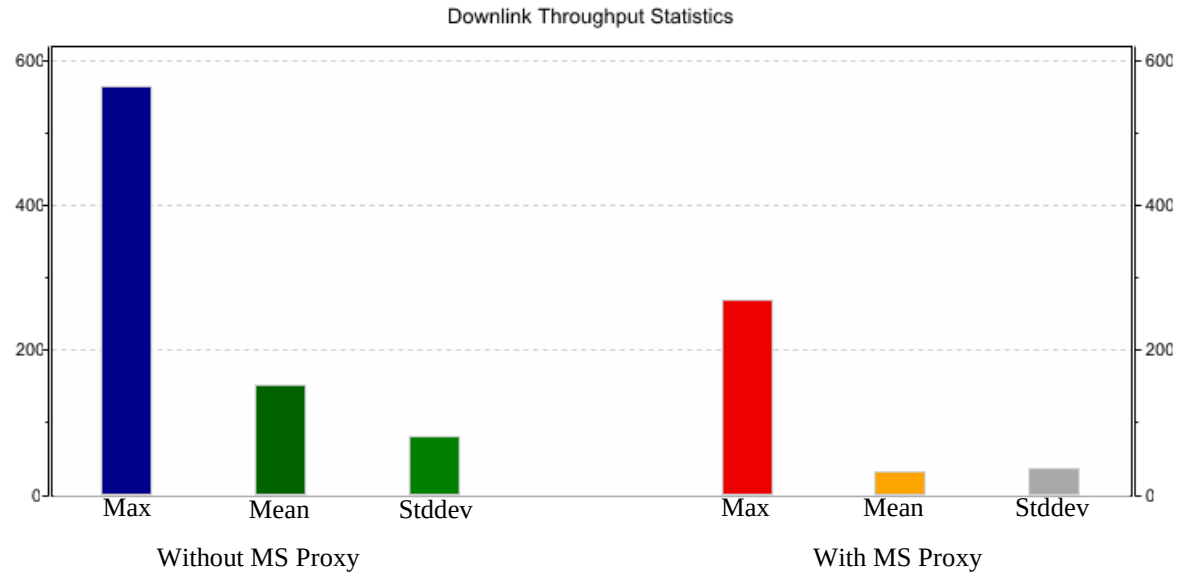

圖 18 – 基地台下行流量統計數據

瞬間流量、平均流量及流量標準差皆有明顯的下降。

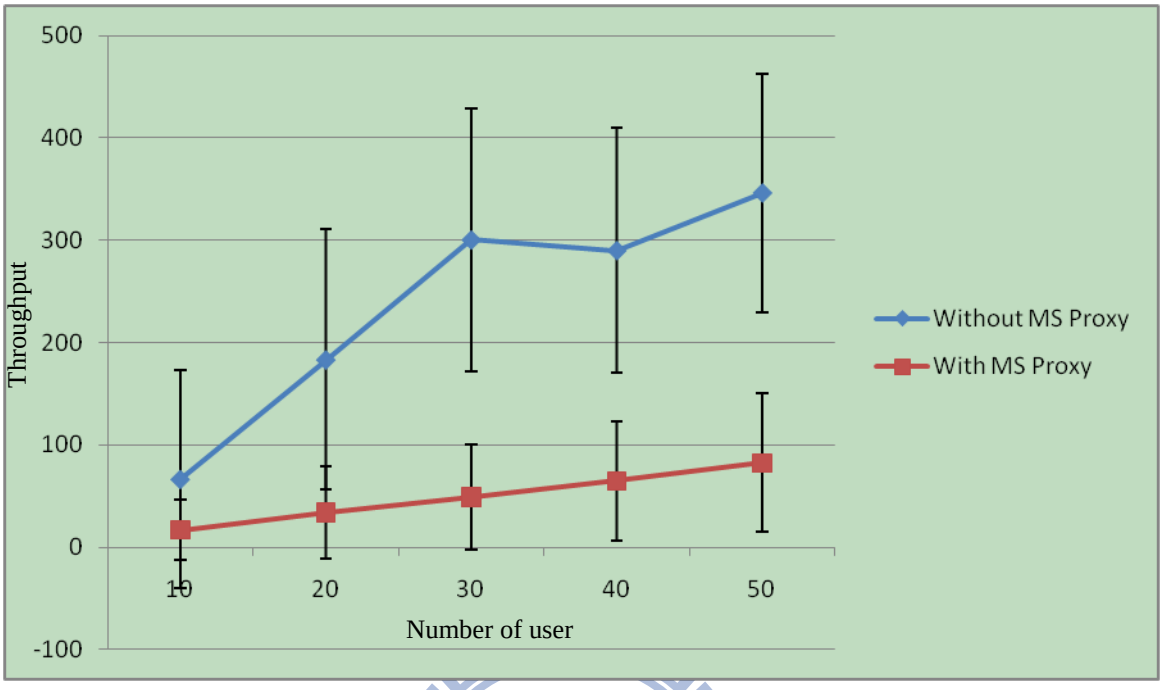

圖 20 – 不同用戶數量下的基地台下行平均傳輸速率

另一方面，由於受到無線頻寬的侷限，可服務的人數也有其限制。藉由快取跟預先下載的機制，可以更有效的利用頻寬資源並且降低尖峰，就如同(圖 19)中所見，代理伺服器可以有效減少基地台的平均下行傳輸速率，使得可服務人數大幅增加。

4.2.2. 應用程式效能評估

當我們考慮減少無線網路的流量負載時，也需思考此方法會如何影響手機用戶的使用體驗。我們以卜瓦松分布來模擬用戶使用應用程式的間隔時間，則有無代理伺服器兩種情形的手機資料流量分別如(圖 23)和(圖 21)中所示。由於快取的緣故可以看出擁有

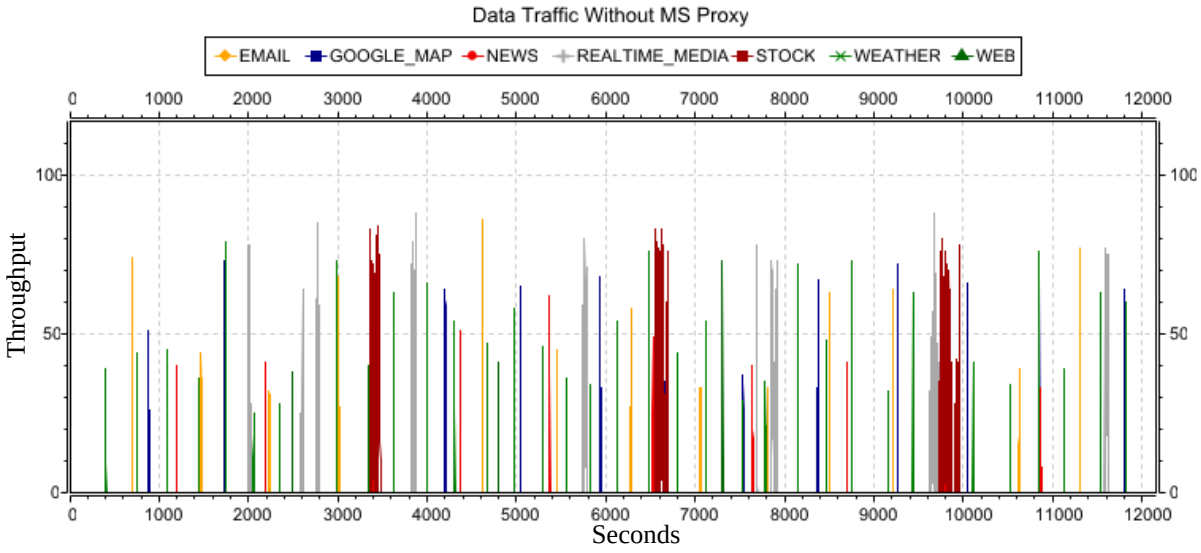

圖 21 – 無代理伺服器的手機流量

代理伺服器資料存取次數明顯的減少，另一方面代理伺服器根據效能函數以決定適當的

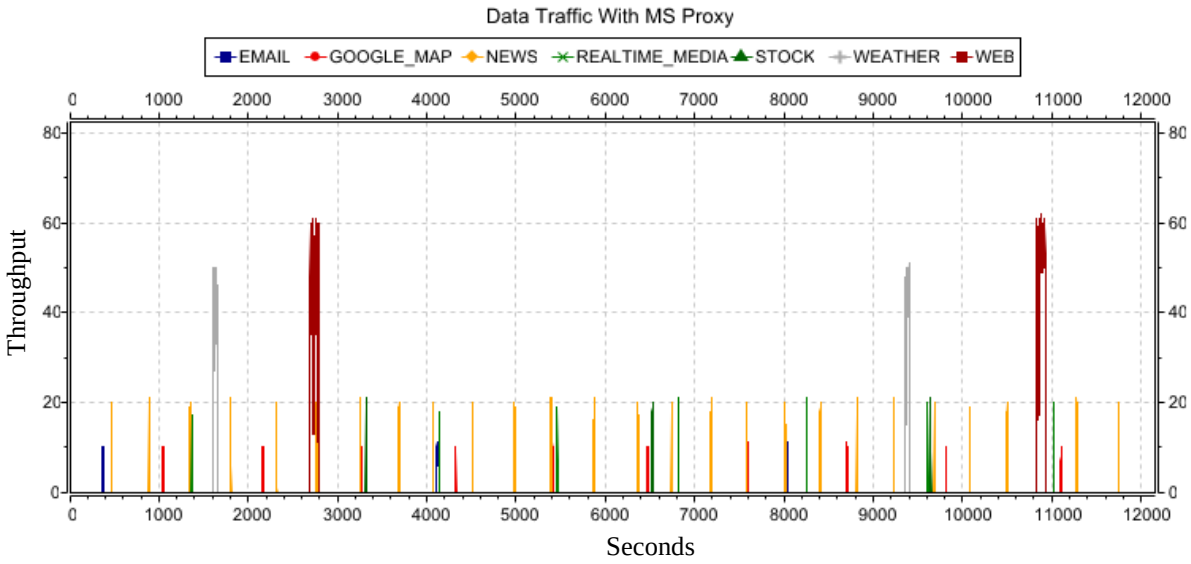

圖 23 – 有代理伺服器的手機流量

資料傳輸速率，在不影響用戶效能的同時，使得頻寬資源能獲得有效的利用。

接著觀察應用程式的使用頻寬(圖 22)，我們可以發現在沒有代理伺服器的狀況下，各個應用程式的平均流量皆差異不大，因為在手機端並無流量控管的機制。但有代理伺服器存在時，則會依據每個應用程式的速率需求不同分配資源，使得頻寬的使用更有效

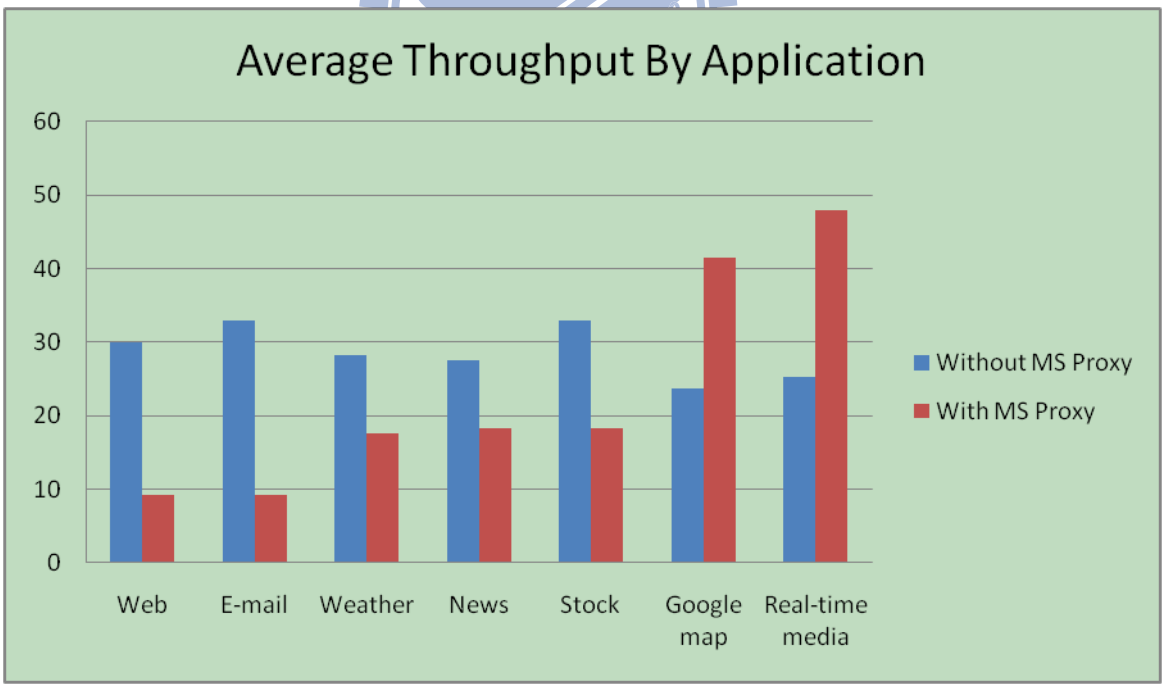

圖 22 – 應用程式平均流量

率。

在無代理伺服器的情形中，各個應用程式的反應時間與資料大小的關係呈現正相關，亦即傳輸資料越多則延遲時間越久。有代理伺服器時，會考慮工作本身的執行期限，隨著時間越來越接近期限，工作的權重也會隨之提高。此時將會優先傳輸此工作並給予較

多的資源，避免了反應時間過長的情形發生。在(圖 24)可以看到，對於資料量越大的應用內容，基於效能決定的排程策略更能發揮其功效。

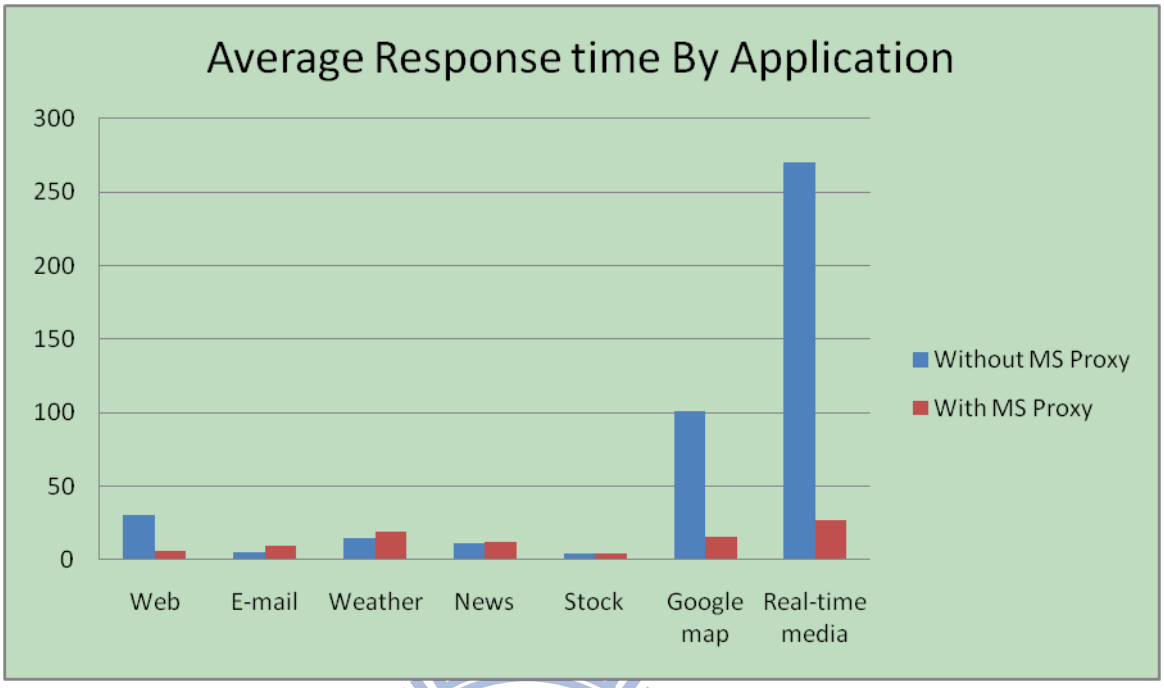

圖 24 – 應用程式平均回應時間

當基地台中用戶數量越來越多時，網路服務的品質也隨之逐漸下降，甚至可能造成壅塞的現象。當這種情況發生時，傳統上只能以丟棄封包的機制消極處理，但是透過快取和預先下載的方法，可以使網路壅塞對用戶造成的影響降至一定程度之下。當基地台下的用戶數量增加時(圖 25)，反應時間仍能保持於可接受的範圍內。

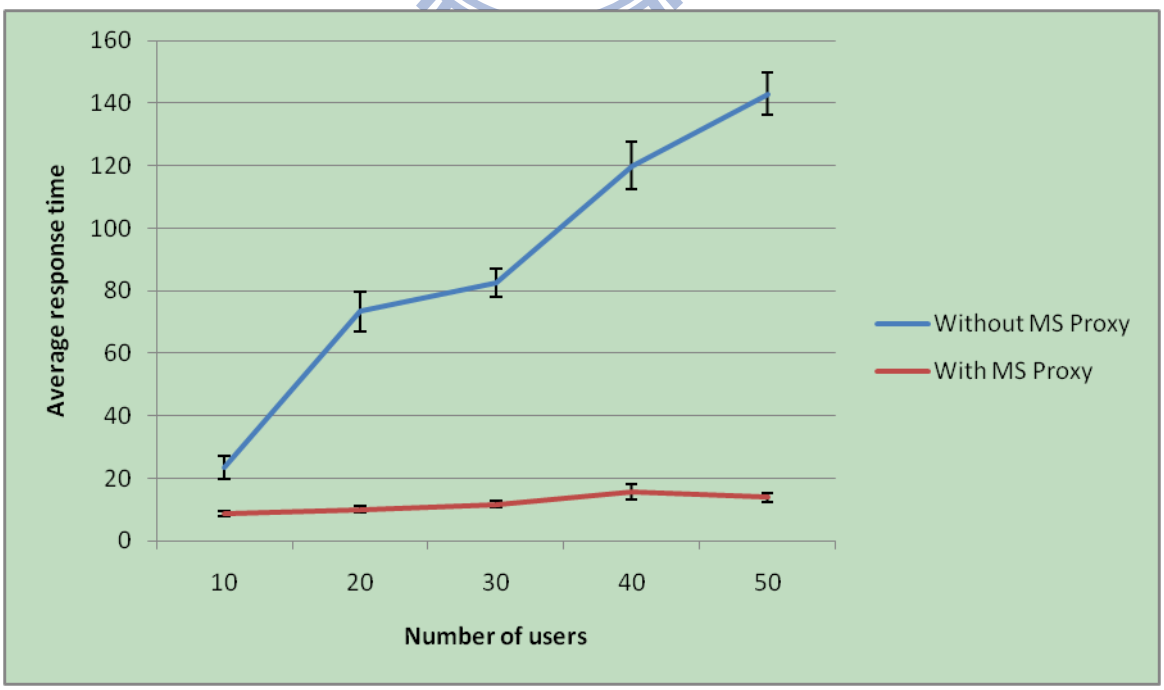

圖 25 – 不同用戶數量下的平均回應時間

五、 結 論

在本篇論文中描述了現今行動網路的架構，並且指出在無線存取網路中，流量分布極度不均衡造成的網路壅塞問題。針對此問題，我們提出三個希望達成的目標:(1)降低無線網路中多餘的資料流量。(2)減少網路流量超出負荷及壅塞的情形發生。(3)替個人最大化下載的資料價值。

為了達成這三個目的，我們引進代理伺服器此元件並設置於手機及網際網路上。位於網際網路上的代理伺服器負責收集應用程式的更新資訊，而手機上的代理伺服器則擁有快取和流量控管的能力。憑藉這兩種設備所提供的功能，得以實現我們的研究目標。

代理伺服器進行流量控管的方式是根據效能函數來決定。效能函數是一種評估網路效能、服務品質，或是用戶滿意度的計算方式。經由效能函數可將抽象的概念量化後，進行運算及比較。我們以過去的研究為基礎提出了適當的效能函數形式，設定了頻寬、封包遺失率、延遲三個影響服務品質的參數與效能的關聯性，以量化服務品質的效能，得出對應的服務品質效能函數。基於多屬性效能定理，可將這三個單一參數的效能以合理的方式整合成為總體效能函數，用來評估在不同網路條件下的用戶滿意程度。

代理伺服器從應用層觀測對外連線情況，因此可直接操作各個連線，不需考慮下層的封包排程方式。我們設計了一個排程演算法，於網路狀況變化、新工作加入或是現有工作結束時，自動選擇應執行的工作列表及其對應的傳輸速率。工作排程的順序基於最大潛在效能，而傳輸速率的控制則以邊際效能決定。我們並證明了此種區域搜尋方式可於工作皆只有一種速率需求的情況下獲得最大效能。

在本論文的實驗分析和結果中，呈現出在有代理伺服器以及適當的演算法的情況下，對於降低網路的負擔有明顯的助益。除了整體網路流量降低之外，網路壅塞的情形也顯著減少。從用戶的角度來看，則可感受到應用程式的反應時間明顯變短，以及服務品質提升等優點。進一步透過個人化的設定，使得有限的頻寬可以運用於重要的資料傳輸，大幅提升用戶獲得的資料價值。

在經過更嚴謹的模擬驗證後，下一步即可考慮將代理伺服器實際部署至行動網路上。手機代理伺服器可加入手機內建的程式，或是放進應用程式商店裡供用戶免費下載。雲端代理伺服器則應架設於網際網路之上，並且需有足夠的數量及頻寬以確保所有手機都能獲得支援。對網路服務商來說，建立以上系統將帶來若干好處。最重要的是無線網路壅塞的問題將能有效的解決，不需添購大量的額外線路，使成本可大幅降低。利用代理伺服器的快取和排程機制，使得現有的網路資源能被妥善的運用，達到網路使用率最佳化的目標。由於服務品質的提升，讓舊有的使用者願意繼續選擇此服務商，同時吸引更多的客戶群加入，形成用戶及服務商雙贏的局面。

參 考 資 料

[1] Shenker, S. (1995). "Fundamental design issues for the future Internet." IEEE Journal on Selected Areas in Communications **13**(7): 1176-1188.

[2] Shi, L., C. Liu, et al. (2008). "Network utility maximization for triple-play services." Computer Communications **31**(10): 2257-2269.

[3] Dharwadkar, P., H. J. Siegel, et al. (2001). "A Heuristic for dynamic bandwidth allocation with preemption and degradation for prioritized requests." ICDCS '01: Proceedings of the The 21st International Conference on Distributed Computing Systems**:** 547-556.

[4] Zimmermann, S. and U.Killat (2002). "Resource Marking and Fair Rate Allocation." 2002 IEEE International Conference on Communications**:** 1310-1314.

[5] Kelly, F. P., A. K. Maulloo, et al. (1998). "Rate control for communication networks shadow prices, proportional fairness and stability." Journal of the Operational Research Society **49**: 237-252.

[6] Harks, T. and T. Poschwatta (2005). "Priority Pricing in Utility Fair Networks. " ICNP '05 Proceedings of the 13TH IEEE International Conference on Network Protocols:311-320.

[7] Chang, C.-S. and Z. Liu (2004). "A Bandwidth Sharing Theory for a Large Number of HTTP-Like Connections." IEEE/ACM Transactions on Networking **12**(5): 952-962.

[8] Chen, J., S. He, et al. (2009). "Optimal flow control for utility-lifetime tradeoff in wireless sensor networks." Computer Networks **53**(18): 3031-3041.

[9] Singh, S. (1996). "Quality of service guarantees in mobile computing." Computer Communications **19**(4): 359-371.

[10] Kroner, H. (2001). "Radio Resource Allocation for Data Services in UMTS Networks." AEU - International Journal of Electronics and Communications **55**(1): 55-62.

[11] Mahadevan, I. (2000). "A Hierarchical Architecture for QoS Guarantees and Routing in Wireless/Mobile Networks." Journal of Parallel and Distributed Computing **60**(4): 510-520.

[12] Campbell, A. T., M. E. Kounavis, et al. (1999). "Programmable mobile networks." Computer Networks **31**: 741-765.

[13] Chandramathi, S., S. P. P. Raghuram, et al. (2008). "Dynamic bandwidth allocation for 3G wireless systems—A fuzzy approach." Applied Soft Computing **8**(1): 274-284.

[14] Yang, X., J. Bigham, et al. (2005). "Resource Management for Service Providers

in Heterogeneous Wireless Networks." IEEE Wireless Communications and Networking Conference. **3:** 1305-1310.

[15] Song, Q. and A. Jamalipour (2005). "A Network Selection Mechanism for Next Generation Networks." IEEE International Conference on Communications. **2:** 1418-1422.

[16] Lu, S., V. Bharghavan, et al. (1999). "Fair Scheduling in Wireless Packet Networks." IEEE/ACM Transactions on Networking **7**(4): 473-489.

[17] Wang, Y., S. Ye, et al. (2005). "A fair scheduling algorithm with traffic classification for wireless networks." Computer Communications **28**(10): 1225-1239.

[18] Shi, J., G. Fang, et al. (2006). "Improving Mobile Station Energy Efficiency in IEEE 802.16e WMAN by Burst Scheduling." IEEE international conference on global telecommunications conference**:** 1-5.

[19] Mendez, A., R. Aquino, et al. (2010). "Performance evaluation of the RQMA-CDMA scheme for multimedia traffic with QoS guarantees." AEU - International Journal of Electronics and Communications **64**(10): 916-923.

[20] Choi, Y., J. Choi, et al. (2007). "Upper-level scheduling supporting multimedia traffic in cellular data networks." Computer Networks **51**(3): 621-631.

[21] (2010). "Fall 2010 Global Internet Phenomena Report", Sandvine.

[22] (2011). "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2010–2015", Cisco.

[23] "LTE Encyclopedia." Retrieved Aug, 2011, from http://sites.google.com/site/lteencyclopedia/home.

[24] Kelly, F. (1997). "Charging and rate control for elastic traffic." European Transactions on Telecommunications **8**: 33-37.

[25] Jiang, Y., C.-K. Tham, et al. (2001). "A Probabilistic Priority Scheduling Discipline for High Speed Networks." 2001 IEEE Workshop on High Performance Switching and Routing**:** 1-5.

[26] Massoulié, L. and J. Roberts (2002). "Bandwidth Sharing: Objectives and Algorithms." IEEE/ACMTransactions on Networking **10**(3): 320-328.

[27] Kim, J.-H. and Y. Y. Kim (2003). "Analysis of User Utility Function in the Combined Cellular/WLAN Environments." Proceedings of the 2nd international conference on Human.society@internet**:** 725-730.

[28] Derbel, H., N. Agoulmine, et al. (2007). "Service Utility Optimization Model Based on User Preferences in Multiservice IP Networks." Globecom Workshops, 2007 IEEE**:** 1-5.

[29] Choi, Y.-J. and S. Bahk (2006). "Delay-Sensitive Packet Scheduling for a Wireless Access Link." IEEE Transactions on Mobile Computing **5**(10):

1374-1383.

[30] Li, Y., A. Papachristodoulou, et al. (2011). "Congestion control and its stability in networks with delay sensitive traffic." Computer Networks **55**(1): 20-32.

[31] Klemm, A., C. Lindemann, et al. (2001). Traffic modeling and characterization for UMTS networks. Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE. **3:** 1741-1746 vol.1743.

[32] Tran-Gia, P., D. Staehle, et al. (2001). "Source Traffic Modeling of Wireless Applications." AEU - International Journal of Electronics and Communications **55**(1): 27-36.

[33] Keeney, R. L. and H. Raiffa (1993). "Decisions with Multiple Objectives: Preferences and Value Tradeoffs", Cambridge University Press.