國立交通大學

電信工程研究所

碩士論文

下世代分頻多工存取被動光網路
之平行化多目標基因演算法動態頻寬分配設計

Toward Parallel Dynamic Bandwidth Allocation
for Next-Generation OFDMA-PON
using Multi-Objective Genetic Algorithms

研 究 生：徐子凱

指導教授：田伯隆　博士

中 華 民 國 100 年 七 月

下世代分頻多工存取被動光網路之平行化多目標基因演算法動態頻寬分配設計

Toward Parallel Dynamic Bandwidth Allocation for Next-Generation OFDMA-PON using Multi-Objective Genetic Algorithms

研 究 生：徐子凱　　Student：Tzu-Kai Hsu

指導教授：田伯隆　博士　　Advisor：Dr. Po-Lung Tien

國 立 交 通 大 學

電 信 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Department of Communication Engineering

College of Electrical and Computer Engineer

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Communication Engineering

July 2011

Hsinchu, Taiwan, Republic of China

中華民國 100 年七月

下世代分頻多工存取被動光網路

之平行化多目標基因演算法動態頻寬分配設計

學生：徐子凱　　指導教授：田伯隆

國立交通大學電信工程研究所碩士班

摘　　要

在此論文我們提出一個最佳化、快速的動態頻寬分配設計(Dynamic Bandwidth Allocation, DBA)，用於下世代分頻多工存取被動光網路(Next-Generation OFDMA-PON)上，此架構具有高頻寬、低成本、節能的優點，同時可整合無線網路訊號的傳輸，我們以多目標基因演算法達到最佳化，並使用平行化進行快速運算，能即時監控網路流量並調整頻寬分配，讓在該架構上的使用者獲得高 throughput 且公平性(fairness)的服務品質保證(QoS)。我們使用 PQS-PR 形式的媒介存取控制(Medium Access Control, MAC)，其類似但不同於傳統的 token-bucket，同時將複雜的頻寬分配機制轉換為較易處理與理解之最佳化問題，利用非支配型排序基因演算法-II (Non-dominated Sorting Genetic Algorithm-II, NSGA-II)，仿效大自然適者生存的法則以取得最佳解，而訊息傳遞介面(Message Passing Interface, MPI)是現今常被採用的平行運算方案，我們用來加速處理最佳化問題的運算速度。我們期望完美結合以上幾種方法，提供在此新穎的下世代光網路架構的使用者，一套完整且最佳的動態頻寬分配與服務品質方案。

Toward Parallel Dynamic Bandwidth Allocation for Next-Generation OFDMA-PON using Multi-Objective Genetic Algorithms

Student : Tzu-Kai Hsu Advisor : Dr. Po-Lung Tien

Department of Communication Engineering

National Chiao Tung University

ABSTRACT

In this thesis, we propose a optimal, fast dynamic bandwidth allocation (DBA) for Next-Generation OFDMA-PON, which has high bandwidth, low-cost, energy-saving advantages and compatible with wireless signals. Our purpose is to monitor network traffic on-line and adjust bandwidth allocation to achieve high throughput and fairness to ensure quality of service (QoS). Multi-objective genetic algorithm (MOGA) is used to achieve optimization and transform complex bandwidth allocation mechanism into a more easily understanding and handled problem. Message Passing Interface (MPI) is now frequently used and we use it to expedite the processing of parallel computing speed. We expect a perfect combination of several methods to provide users the next generation optical network architecture with optimal bandwidth allocation and quality of service.

誌謝

本篇論文的完成，首要感謝指導教授田伯隆老師，在這兩年中，從無到有訓練我思考問題、解決問題的能力，在遇到瓶頸的時候給予適當的指導，同時對於研究所需的資源也不吝於支持與付出。接著，感謝資工所楊啟瑞老師與施汝霖博士在網路領域的啟蒙與基礎教導為我奠定基礎。此外，實驗室的柯柏宇學長、王櫻瑾同學，學弟妹陳星豪、蕭佑霖以及馬毓晴，平時頻繁的研究討論以及生活交流，讓整個實驗室的研究氣氛非常融洽，使我能隨時保持良好的狀態進行研究。最後也最重要的，感謝父母一路支持鼓勵我攻讀碩士學位，不論是生活的幫助或是心靈方面的鼓勵，讓我能夠無後顧之憂地取得人生中重要的學識資產。

目錄

圖目錄

1 次世代分頻多工存取被動式光網路

1.1 背景介紹

光纖技術因為其高頻寬、低雜訊等特點使其成為用來取代傳統低頻寬 ADSL 並能夠提供 triple-play(數據、影、音三合一)的解決方案，而其中又以具有高頻寬，低成本的被動式光纖網路(Passive Optical Network, PON)被寄予高度期望。PON 的優勢在於可以利用光纖提供高品質與高頻寬的資料傳輸，同時由於 PON 架構僅在局端及用戶端設備使用主動元件，在兩端中間的傳輸線路中則完全使用被動式光元件，因此避免了主動元件暴露在環境中可能衍生的許多維修以及電源供應的問題。

近年來由於數位訊號處理(Digital Signal Process, DSP)晶片的快速發展及成本大幅下降，[1][2][3]率先提出將發展多年並大量應用在無線通訊的正交分頻多工(Orthogonal Frequency Division Multiplexing ,OFDM)技術應用在 PON 上。由於 OFDM 同時在 frequency domain 及 time domain 將頻寬切割成多個正交子載波(orthogonal sub-carriers)，而且其頻寬單位甚小(4bits~16bits)因此 bandwidth granularity 相較於 TDM-PON 要來的細緻很多，因此每個 ONU(Optical Network Unit)可以很彈性的用固定或是動態方式分配到一個或多個子載波。除此，OFDM 目前可以利用至少 16-QAM 以上的 modulation 技術將訊號傳輸率大幅提高，相對於傳統 TDM-PON 上使用的 on-off keying (OOK) format，具有極高的 spectral efficiency[4]，因此可以使用較低速(例如 2.5G)的光電零組件卻能達到很快的傳輸速度(例如 10G)，因而降低成本及硬體需求門檻。另外，利用 OFDM 的特性可以用來解決光纖中光色散現象（fiber chromatic dispersion），因此訊號傳遞可以達更遠距離，極適合用來達成 long reach PON[5]的要求。總結 OFDMA-PON 相對於 TDM-PON(EPON、GPON)及 WDM-PON 的優勢在於其既可以提供比 TDM-PON 更大的傳輸頻寬、更細緻的頻寬分配，以及可以多個 ONU 同時上傳的優點，但

卻避免了 WDM-PON 高成本、不易達成 colorless 需求、以及波長不易共享導致頻寬浪費的缺點。更重要的是其系統建置成本極低，因此可以預見 OFDMA-PON 將成為下一世代極具潛力的 PON 架構。

此外，無線網路建置容易、能夠提供移動通訊、使用方便而成為現今網路的發展重點，但其因訊號干擾及建築物遮蔽效應所產生的衰退(fading)現象而造成訊號品質不良的問題一直是多年來研究者積極想解決的課題。其中 MIMO(Multiple-Input and Multiple-Output)系統利用多組收發天線搭配加解碼技術，被認為是改善無線訊號品質的最佳方案之一，近來建構在 MIMO 系統基礎上的 Network MIMO 技術進一步延伸基本 MIMO 系統的操作範圍，其透過多個基地台的快速同步(multi-cell coordination)共同解調出最佳的信號，使得在 cell edge 訊號不佳的問題可望獲得解決，達到整體系統最佳的 throughput。

值得一提的是光載波技術(Radio over Fiber, ROF)[6]，RoF 的基本運作方式是在基地台直接調變光強度，將 RF 訊號的振幅直接調變雷射的電流以製造出相對應的光訊號，然後再將調變後的光信號利用光纖從基地台傳送到使用者所在的天線端，最後只需一個接收器和光放大器，即可將 RF 訊號送出。由於光傳輸的特性，此種作法可以完全保留 RF 信號的形式與格式，任意頻帶的 RF 訊號皆可直接載在光纖網路上。除此之外，因為將所有複雜的處理都交由基地台負責，所以使用者端所需要的硬體組件少、複雜度低、消耗的功率極低，可符合綠能的需求，且透過光纖網路傳輸距離遠、低功率衰減的特色將 RF 信號的覆蓋範圍擴增。

因此，如何利用光纖網路(PON)傳輸距離長、成本低、頻寬高的特點並支援高彈性的無線網路，亦即提供所謂的 Quadra-play 服務[7]，開始成為最近重要的研究課題。探究近年來 PON 與無線網路的混合架構，作者[8]提出關於 PON 與 WiMAX 的兩種整合方式如圖 1.1。第一種整合模式是 EPON 和 WiMAX 系統各自獨立操作(independent architecture)，WiMAX base station 透過線路連結到 ONU，在此這個架構下 ONU 和 BS 各自擁有獨立的系統，不但硬體設備無法共用，傳

輸格式亦不相同，因此需要作訊號的轉換，且因各自為獨立系統，整體的頻寬的分配無法達到最佳化，失去整合的意義。第二種整合模式則是將 ONU 與 WiMAX base station 整合在單一的系統(hybrid architecture)。如此可以較容易使兩個系統在硬體和軟體方面作整合以降低成本，並改善頻寬分配的效率。但在此架構下，因 EPON 為 TDM 的光信號傳輸模式，而 WiMAX 則是 OFDM 的電訊號傳輸模式，因此在 BS 接收無線訊號後必須先解調為電訊號，再轉換成 PON 的光資料格式，在處理過程中過於複雜且成本高。

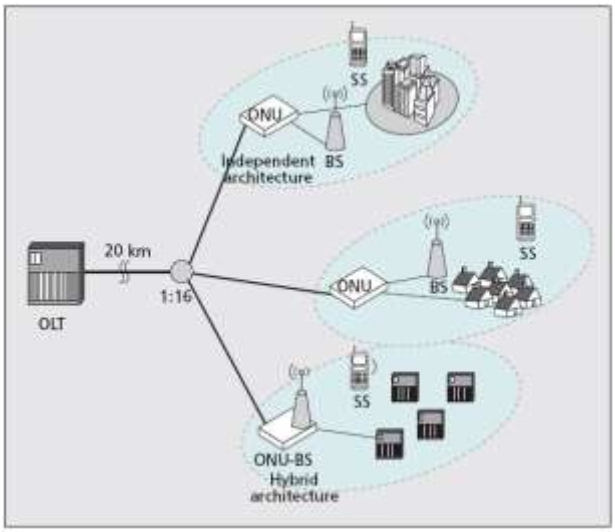

圖 1.1 EPON 和 WiMAX 系統整合模式

鑑於上述的缺點，有學者提出圖 1.2 的架構[8]以改善這些缺失。此架構利用 PON 做為骨幹，在 ONU 端利用 WiMAX remote antenna 取代價格昂貴的 base station，而 WiMAX base station 則是和 OLT 整合在一起。這樣的好處在於成本大幅下降，卻仍可以使無線接取網路覆蓋的範圍擴大。在系統實作上可以利用 Radio-Over-Fiber (ROF)技術將無線訊號直接調變成光訊號而降低硬體複雜度及成本。但隨之即而來的問題為 RA 隨接即送不暫存訊號的特性，而使得多個同頻率的無線訊號會同時載在光纖網路，而在 splitter 到 OLT 之間造成訊號碰撞的現象，亦即所謂的 Optical Beat Interference (OBI)的問題。另外，由於 WiMAX 與 PON 的訊號格式仍舊不同，如何達到兩個系統 seamless 的結合仍是很大的挑戰。

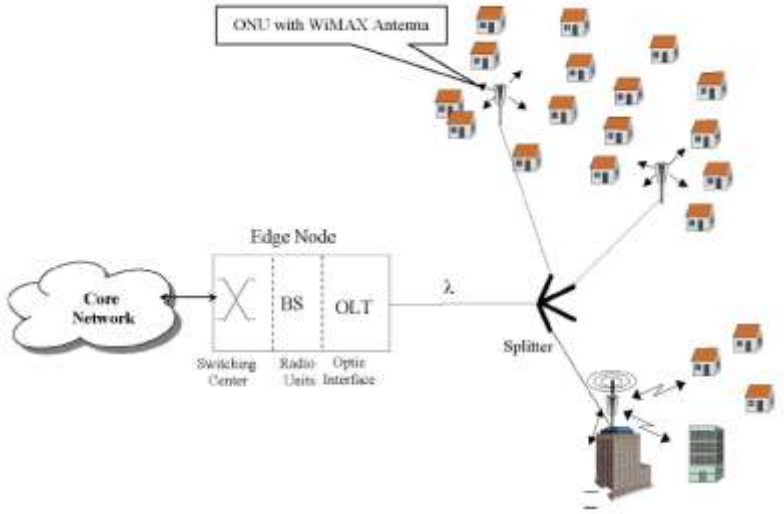

圖 1.2 ONU with WiMAX Antenna

NEC America [9][10]率先提出結合 WDM-PON 與 OFDM 技術而能夠支援 ROF 的系統如圖 1.3 所示。此架構在 WDM-PON 上採用與 WiMAX 相同的 OFDM 格式，解決了訊號格式不同的問題。另外，因此用 WDM 架構使得每個 ONU 皆使用單獨的波長而避免了 OBI 的問題。但是 WDM-PON 伴隨而來的缺點即是多個波長無法共享而造成頻寬浪費而且光元件成本過高。

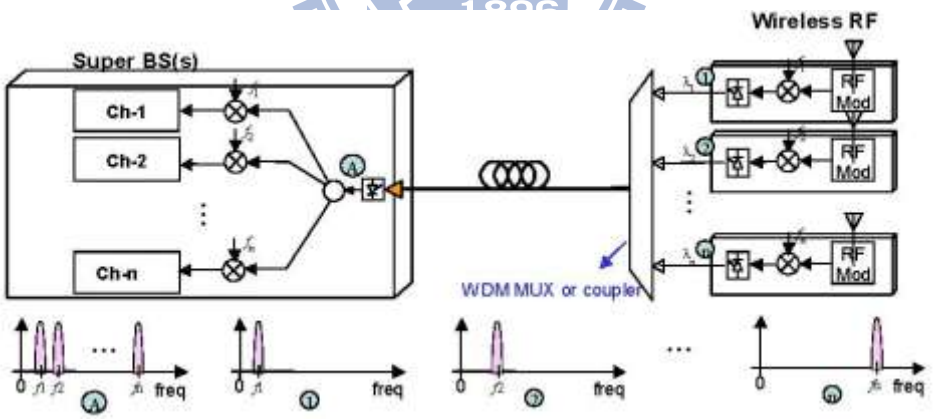

圖 1.3 NEC America 之結合 WDM-PON 與 OFDM 技術

現存 PON 網路架構除了存在許多缺點，更難以避免為支援 RoF 而產生 OBI 的問題，本篇論文採用的 OFDMA-PON 可支援 RoF，避免 OBI 問題以達成 quadra-play 的要求，且避免採用 WDM 以降低成本。

基於以上的技術背景，Next Generation OFDM-PON 應運而生，下一小節將介紹其如何結合以上技術，建構下一世代的被動式光網路。

1.2 Next Generation OFDMA-PON 架構介紹

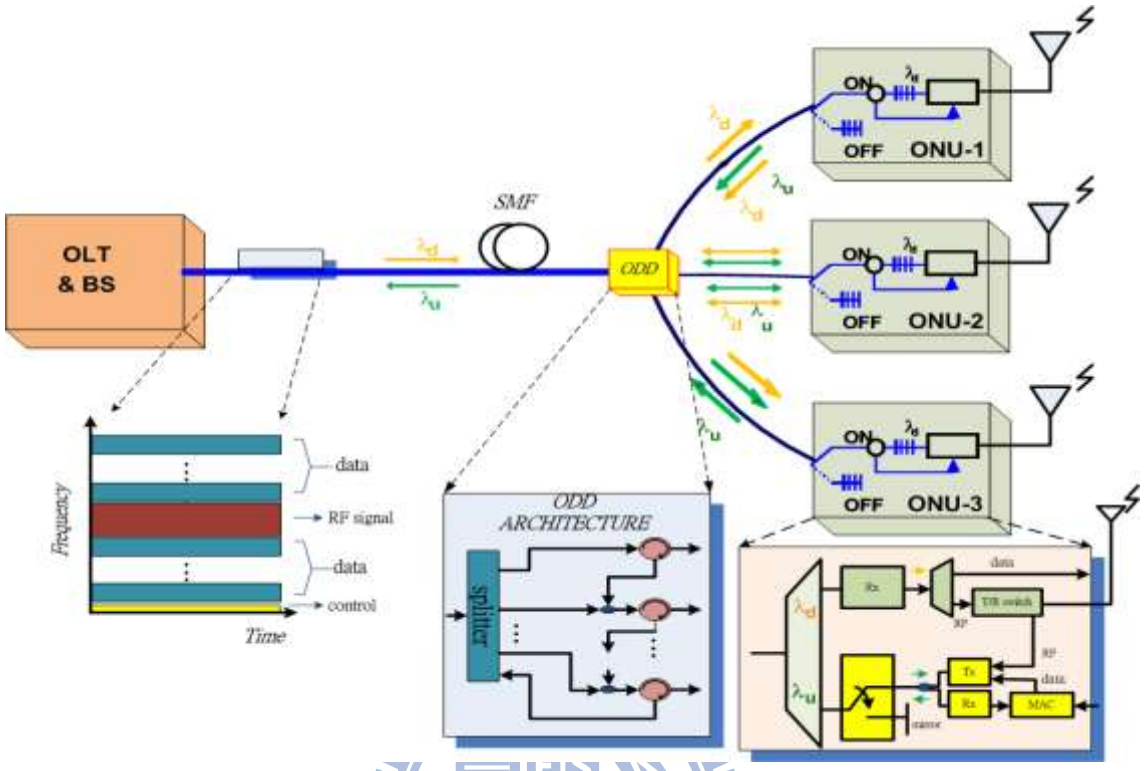

圖 1.4 Next Generation OFDMA-PON 架構

Next Generation OFDMA-PON 架構如圖 1.4，OLT(Optical Line Terminal)經由 feeder fiber 連接到 ODD(Optical Distribution node)的 splitter，ODD 再經由 optical circulators 連接到 ONUs(Optical Network Unit)，其中 OLT 可視為基地台，ODD 為轉傳分配的中繼站，ONU 則為使用者端。

此架構的上行傳輸(upstream)和下行傳輸(downstream)分別使用波長λ_u和λ_d，在下行傳輸中，OLT 傳送的資料經由 ODD 的 splitter 和 circulator，直接傳送給對應的 ONU，類似於 broadcast 的形式；在上行傳輸中，ONU-1 將欲上傳的資料和控制訊號透過 circulator 傳送給 ONU-2，控制通道則為 OFDMA 預先分配好一個子通道，接著，ONU-2 接收 ONU-1 的上行資料和控制訊號，ONU-2 藉由控制訊號得知分配到的子通道，於是將 ONU-1 的訊號解調後加上本身的訊號，重新產生一組新的 OFDMA 訊號。餘此類推，最後一個 ONU-N 透過 ODD，於上行傳輸將資料回傳給 OLT，此資料包含所有 ONUs 的資料。並且，在每個 ONU 中

皆設置一個光切換器(optical switch)，若有 ONU 發生故障或是未開機，上行傳輸可直接略過該 ONU，繼續完成傳輸的工作。

上述的 OFDMA-PON 可結合 RoF 來提供無線訊號服務，做法是在無線訊號的頻帶位置處，將 OFDMA 的子載波訊號接設為零，亦即在頻譜圖中，為了避免無線訊號和寬頻有線訊號發生頻帶重疊，挖一個特定位置的洞來支援無線訊號的頻帶。假設在上行傳輸中 ONU-1 透過 Remote Antenna 接受到一無線訊號，利用一個 mixer、oscillator、中頻濾波器 BPF1，將無線訊號的使用頻帶移動到洞口內位置，使其和有線訊號不會發生頻帶重疊，也使有線訊號和無線訊號之間的干擾可以被控制，然後將移頻過後的 RF 訊號和有線訊號透過 circulator 同時傳輸給 ONU-2，而 ONU-2 接收到 ONU-1 的訊號後，先將訊號經過 splitter 複製成兩份，第一份，將 OFDMA PON 的訊號透過 FFT 解調，因特定子通道為控制通道，根據控制訊號得知可使用的子載波，將於上傳的訊號加入後，重新產生一 OFDMA 訊號，另外一部分，將訊號經過一個中頻濾波器 BPF2，將 OFDMA 的訊號濾掉，接著把 ONU-2 天線收到的訊號，移頻到 RF1 的旁邊，但依然還是在洞口內，接著把 RF1 和 RF2 訊號加起來。最後把這些無線訊號和重新產生過的 OFDMA 訊號同時傳送出去。以此類推，若有 N 個 ONUs，最後一級 ONU 會把一組 OFDMA 訊號和 N 組 RF 訊號用波長為λ_u的雷射傳送回 OLT。如此一來，便可以成為一種新的 OFDMA 架構且支援無線訊號。

在該架構下，上行傳輸利用類似環光網路(ring)的方式將訊號串連，因此只需一個單一波長為λ_u的雷射，便可以達到上行傳輸的需求，相較於以往的 TDM-PON 架構，可解決多個無線訊號在光網路中彼此碰撞而產生的 Optical Beat Interference(OBI)，也解決 WDM-PON 多個波長無法共享而造成的頻寬浪費和光元件成本過高的問題。OFDMA 則提供多個彼此正交且互相不受干擾的子載波(subcarrier)用以傳送和接收訊息，達到頻寬使用效率高和分配方便的目的。最後配合 RoF 技術與無線接取網路結合，骨幹網路端使用光纖網路，靠近用戶端提

供無線接取網路，可避免建築物中光纖布建困難的問題，對於複雜地形擁有絕佳的適應力，提供使用者高頻寬且低功率的無線網路環境。

下一章節將介紹在 Next Generation OFDMA-PON 上動態頻寬分配的運作，如何提供使用者良好公平的服務品質。

2 Next Generation OFDMA-PON 上之動態頻寬分配

2.1 背景介紹

目前以 TDM 為基本傳輸模式的 E(Ethernet)-PON，G(Gigabit)-PON，以及 GE(Gigabit Ethernet)-PON 都已標準化並即將被鋪設使用。然而要提供多媒體傳輸的應用服務，除了要滿足高頻寬需求之外，還須面臨使用者端流量訊務之即時變化，以及非常嚴格之傳送延遲之要求。因此如何設計高效率的動態頻寬分配(Dynamic Bandwidth Allocation, DBA)機制，以同時滿足各用戶端的公平性(fairness)以及服務品質保證(QoS)便成為 PON 上極為重要的問題。過去已有眾多的文獻提出在 TDM-PON 上各種 DBA 演算法，約可劃分成 centralized control 及 decentralized control[11]。

- Centralized control

Kramer G.等人在 2002 年針對 EPON 架構，提出 IPACT[12]動態頻寬分配演算法。此方法中 OLT 內部會有一 polling table，紀錄著每個 ONU 請求上傳的資料大小以及各個 ONU 的 Round Trip Time。OLT 首先對 ONU-1 發出 Grant，ONU-1 收到 Grant 後，開始上傳資料並在資料的末端加上下次要上傳資料量的請求(Request)。同時，OLT 提前將 Grant 發給 ONU-2，待 OLT 一接收完 ONU-1 傳送來的資料，便繼續接著收取 ONU-2 傳來的資料。OLT 使用如此交錯式 polling 的方式，依序讓每個 ONU 得以上傳資料，這樣的 Grant-Request 的方式，後來被採用在 Multi-Point Control Protocol (MPCP)通訊協定上。IPACT 的缺點在於當某個

ONU 上傳的資料量太大，會導致其它 ONU 等待的時間過長。這種 bursty traffic 的特性在靠近使用者端的網路時常會發生，不僅對其它 ONU 不公平且不利於 QoS 的保證，難以提供 triple-play 的服務。

J. Zheng 基於 MPCP 協定，提出新的演算法[13]以改進 IPACT 的缺點。此演算法是在上一個 cycle 結束時，OLT 已得知每個 ONU 的 Request，而在新的 cycle 開始時按照每個 ONU 預設的 weight 比例，先計算分配部分頻寬給每個 ONU。由於在每次的 cycle 中，某些 lightly loaded ONUs 需求的頻寬可能會比這次計算出來的分配頻寬小，而 heavily loaded ONUs 的需求可能比分配頻寬大，因此在 OLT 分配完後，這些多出來的頻寬再分配給 heavily loaded 的 ONUs 利用。此外，此演算法利用 early allocation 的方式，提前分配頻寬給低頻寬需求的 ONUs，因此在低或中等的 traffic load 環境下可提升頻寬利用度。這樣的動態頻寬分配方式，重於充分利用整體的頻寬以減少頻寬浪費，但如果每次 cycle 中，某個 ONU 一直處於 heavily loaded 狀態，長時間下來它勢必比其它 ONU 拿到更多的頻寬，而無法提供 fairness 的要求。

Hassan Naser[14]等人則更細部的依 Class-of-service 等級，去分配頻寬。假設每個 ONU 皆包含 N 個佇列，每個佇列服務一種 Class-of-service(CoS)。例如將 voice、video 和 data traffic 分成不同的等級(CoS)。在前一次的 cycle 結束後，OLT 收到所有 ONU 的每個佇列中所有訊息，然後利用 Weighted Random Early Discard(WRED) buffer 管理機制，將所有佇列的資料分類到各個 CoS。接著演算法分成兩個 round 去計算分配頻寬。在 round 1 中，OLT 先以 CoS1 給予所對應的 ONUs 頻寬後，再依 CoS2 給予所對應 ONUs 頻寬，依此順序，直到 CoS n。而在 round 2 中，OLT 計算還有剩的頻寬才分配給 CoS2。接著 OLT 才會發出 grant 給各個 ONU 依分配到的頻寬上傳資料。這樣的動態頻寬分配演算法，已經將 QoS 的問題考慮在內，不過 DBA 過於繁雜，而且 CoS 等級低的資料，可能會有送不出去的情況發生。此外，此演算法還是傾向於充分利用所有頻寬，如果某個

ONU 一直有大量 CoS1 的資料要上傳，在每次 cycle 它依然會分配到頻寬，對於其它的 ONU 來說，一段時間後總得的頻寬少於此 ONU 所獲得的頻寬，並不能提供 fairness 的要求。

- Decentralized Control

之前的研究焦點皆放在 OLT 對所有 ONU 之間的流量控制，而 decentralized 這類 DBA 則除了 OLT 的頻寬分配外，再加上 ONU 內部的 scheduling 以提供更精確的頻寬分配，基本上可分為 inter ONU scheduling 及 intra ONU scheduling。在[15]中作者即針對在 intra-ONU 作頻寬分配，假設有 N 個 ONU，每個 ONU 皆有 M 個 queue，以因應不同的 service level agreement，例如 throughput、delay 等等。作者提出 M-SFQ 演算法將封包排程演算法 SFQ 改良成適用於 PON 上的排程演算法，使 M 個 queue 在上傳封包得到公平性的分配，但僅止於單一的 ONU 內部。在 OLT 與 ONU 之間的動態頻寬分配，仍然很難達到理想的 QoS，主因仍在於眾多的 ONU 僅能共享一個上傳通道，當每個 ONU 皆有最高優先權的封包要傳送，全部都服務完也得經過 M 次 RTT，無法滿足 delay 要求嚴格的封包。而在[16]中作者提出整合 inter ONU scheduling 與 intra ONU scheduling 的演算法。在 inter-ONU scheduling 部分，利用 EPON 上的 REPORT-GATE 機制，與以往的 REPORT 不同之處在於，REPORT 內包含了最大及最小 window size 需求。OLT 在收到所有的 REPORT 後，除了滿足最小需求外，額外的頻寬會根據其權重分配給 ONU 或是 queue，並保證任何 ONU 或 queue 獲得的頻寬皆不可超過最大需求。這樣的演算法能夠提供每個 ONU(或 queue)保證最小頻寬配置。就 throughput 及 delay 而言，也能保證較高優先權的 queue 有較少平均延遲，而相同優先權的 queue 能滿足公平性。

存取控制機制與 DBA 的目的是要支援接取端(ONU)最重要的多媒體服務(triple-play)，因此除了需要達到各 ONU 間的公平性外，還需要滿足各種不同應用媒體，如 internet phone、IPTV、on-line gaming、peer-to-peer 應用等，對頻寬、

封包漏失率、平均時間延遲、以及延遲變異量的嚴格要求。除此，在高速環境下網路狀況變化極快，存取控制需在極短時間動態決定頻寬分配，因此演算法複雜度必須降低。現存 DBA 皆是基於 TDM-PON 或是 WDM-PON 的架構，然而 OFDMA-PON 在頻寬單位(sub-carrier 或是 symbol)的細緻度(granularity)以及頻寬使用的方式(frequency domain + time domain)皆與前兩者截然不同，而且還要考量控制頻道(control channel)頻寬有限，symbol 分配必須保持連續性以減少控制訊息等限制。因此，在網路架構本質上與傳統架構差異頗大，過去的 DBA 將無法直接應用於本系統，我們在下節將提出在該架構上適用的頻寬分配機制，能提供使用者具有公平性及高 throughput 的服務品質保證。

2.2 動態頻寬分配介紹

首先，需說明在此架構下，我們定義每個 ONU 可以得到的頻寬單位，為在每個 cycle 內可獲得的 OFDMA 子載波數目，而 cycle 是下行傳輸由 ONU-1 到 ONU-32 繞一圈收集完資料的時間；若於每個 cycle OLT 提供 512 個子載波給 ONUs 使用，假設 ONU-1 有 100 個封包需傳送，到 ONU-2 就剩下 412 個頻寬單位，以此類推。因此，若不給予任何頻寬分配的機制，當下行傳輸於前端數個 ONUs 取得資料，後端的 ONU 會有極高的機率因為剩餘頻寬不足，而導致封包等待時間(delay)過長。

因此，我們提出動態頻寬分配機制：分配每個 ONU 一個虛擬的 buffer(緩衝區)，其容量為 PQS(Permit Queue Size)，限制最大庫存的傳輸許可，另於每個 cycle 以速率 PR(Permit Rate)放入允許 ONU 傳輸的封包數量，限制固定配給之傳輸許可。圖 2.1～圖 2.5 是一個例子用來說明在一個 cycle 內的頻寬分配機制，假設 OFDMA 每個 cycle 提供 10 個子載波，PON 上有三個 ONU，其(PR, PQS)依序是(2, 2)、(4, 8)、(3, 10)，等待上傳之封包數則分別為 4、5、4，詳細操作說明如下：

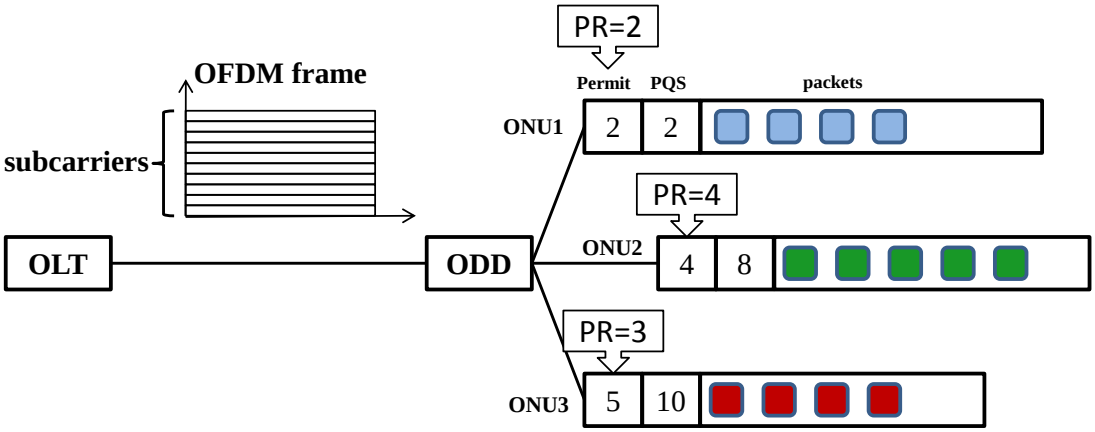

圖 2.1 An example of DBA: step1

Step1，OLT 提供一個 10 個子載波的 OFDM frame，每一個子載波供一個封包上傳，同時，每個 ONU 添入在這個 cycle 中可以獲得的 Permit，PQS 則是一開始就設定可以放入的最大 permit 數量，若是超過這個值，多餘的 permit 就會被剔除，ONU 後方則是表示有幾個封包等待上傳。

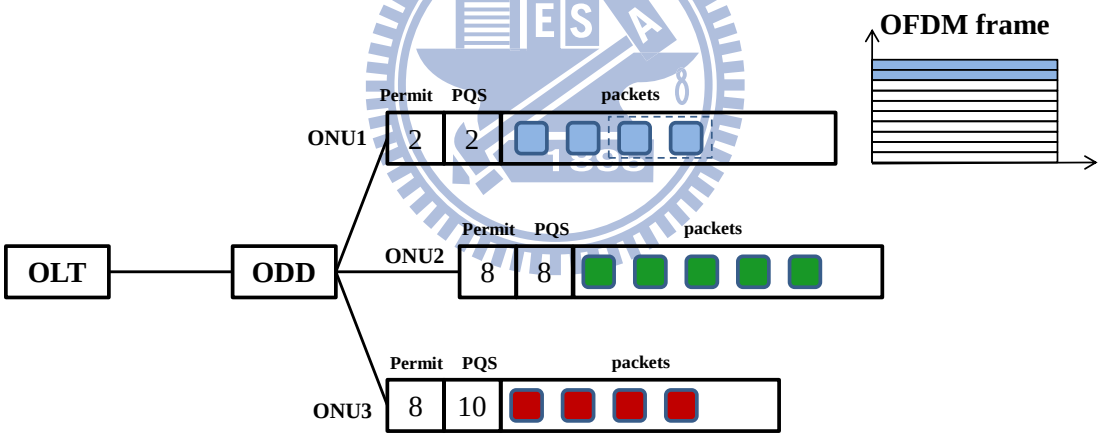

圖 2.2 An example of DBA: step2

Step2，需先注意各 ONU 中 permit 的數目，ONU1 原本有 2 個 permit，PR＝2，但因 PQS＝2，因此剔除多餘的 2 個 permit，總數仍為 2；ONU2 及 ONU3 則都尚未達到 PQS 上限，因此 PR 提供的 permit 可以全數放入。接著，ONU1 首先上傳，OFDM 剩餘頻寬為 10，ONU1 permit 數量為 2，因此 4 個 ONU1 的封包只能上傳 2 個。

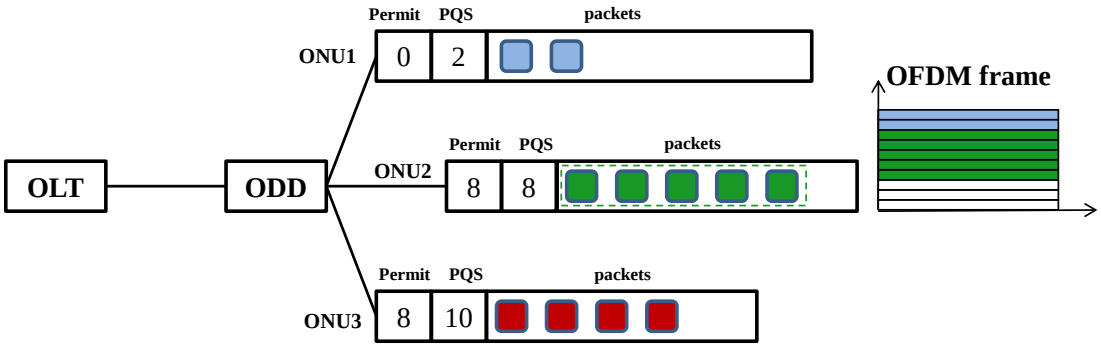

圖 2.3 An example of DBA: step3

Step3，首先需注意 ONU1 的 permit 因上傳兩個封包而扣除 2，目前為 0。接著判斷 ONU2 上傳，此時 OFDM 頻寬剩餘 8，ONU2 擁有 8 個 permit，5 個待上傳封包，因此所有封包皆可上傳。

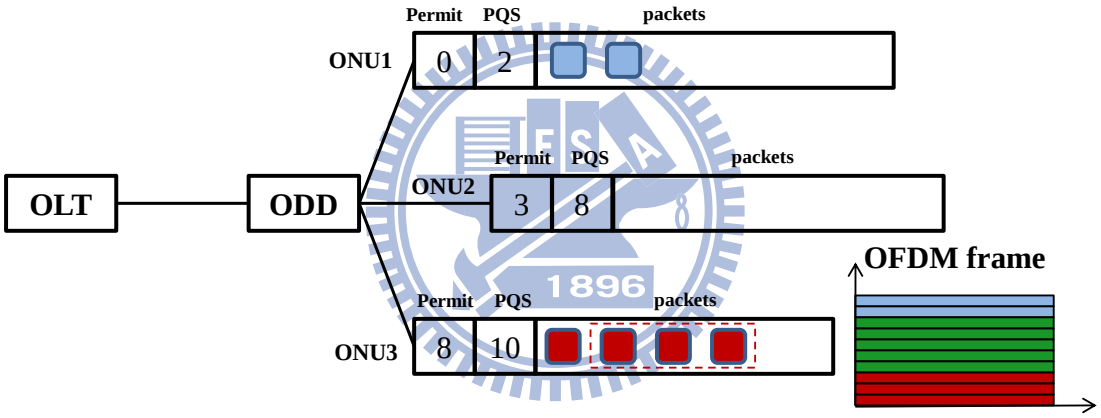

圖 2.4 An example of DBA: step4

Step4，ONU2 的 permit 因上傳 5 個封包，由 8 降至 3。此時 OFDM 頻寬只剩 3 個子載波，因此，ONU3 雖然有 8 個 permit 足以上傳佇列中 4 個封包，但仍只能上傳 3 個封包。

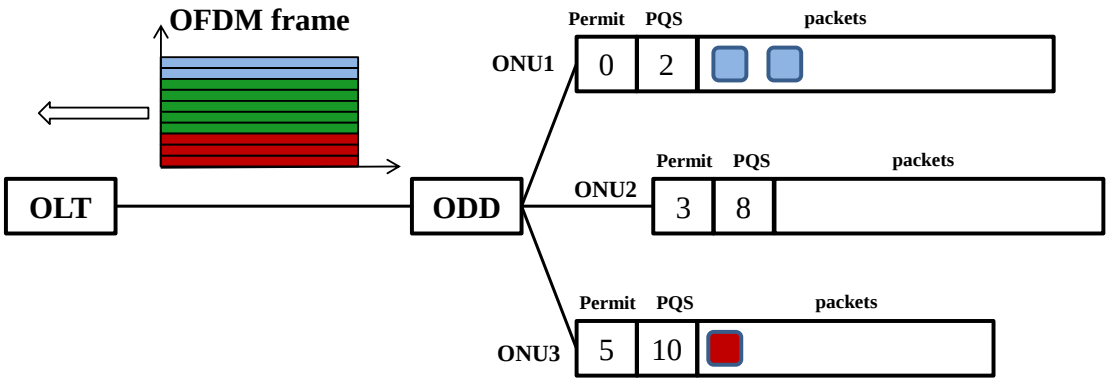

圖 2.5 An example of DBA: step5

Step5，ONU3 因上傳 3 個封包，permit 剩下 5，此時 OFDM 已接受所有 ONU 之上傳，因此上傳至 OLT，而各 ONU 剩餘之封包數則分別為 2、0、1。

由此方法控制 ONUs 上傳資料的頻寬分配，產生下列判斷上傳封包數之流程，如圖 2.6：

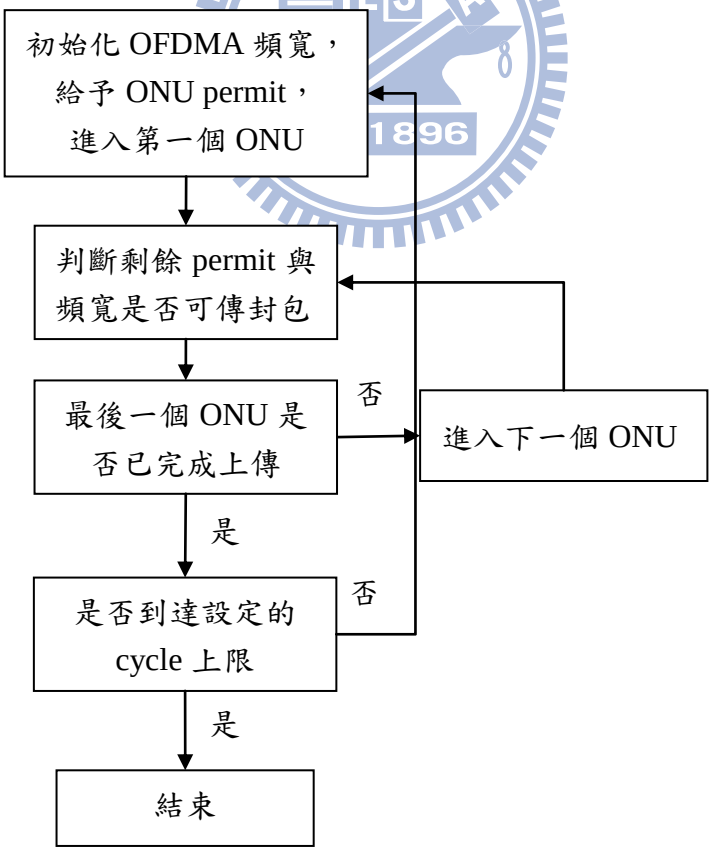

圖 2.6 頻寬分配機制流程圖

- Step 1：cycle 開始，於 ONU-1 填入預定之 Permit Rate，若達到上限 PQS，則剔除多餘的 permit。
- Step 2：ONU-1 以 buffer 剩餘之 permit 和可用之 OFDMA subcarriers 判斷可上傳之封包數量；若欲上傳之封包數≦permit 且 subcarrier 足夠，則上傳所有封包，並剔除相對封包數量的 permit；若欲上傳之封包數＞permit 且 subcarrier 足夠，則上傳數量為 permit 之封包，並剔除所有 permit。
- Step 3：將 OFDMA subcarrier 之數量扣除步驟 Step 2 之 permit 數量，剩餘之子載波留於 ONU-2 使用，同時開始接受 ONU-2 資料之上傳判斷。
- Step 4：ONU-2 重複步驟 Step 1 至 Step 3，並將步驟 Step 3 中對象改為 ONU-3，重複循環以上動作，直到最後一個 ONU，結束此 cycle 資料之上傳。

為了判斷在該動態頻寬分配機制下，各個 ONU 所能得到之服務品質(Quality of Service, QoS)，我們定義封包之 delay 如下：

若某封包於 cycle T_1 到達所屬之 ONU 等待上傳，直到 T_2 被判斷可以接受上傳，則封包之 mean delay 就等於($T_2 - T_1$)；當觀察之目標為 ONUi 時，則該 ONU 封包之平均 mean delay

$$\mathrm{MD}i = \frac{\text{所有已上傳封包之 delay 總和}}{\text{所有已上傳封包之總數}} \text{。} \qquad \text{(式 2.1)}$$

而後，我們以各個 ONU 相對應之 MD 判斷各其是否有得到公平之頻寬分配，目標是在觀察網路流量的同時，即時且動態的分配 PQS 與 PR，使得所有 ONU 能取得最低且相等之 MD，達到 fairness 與 QoS 之要求。經由這樣的設計，若位於前端的 ONU 有大量封包要傳送而佔用大量頻寬時，PQS 將限制前端 ONU 一次可送出的最大封包數，一但 permit 用盡，就必須再由 PR 重新累積，此時後端 ONU 就會有機會傳送。或者，我們可以為後端的 ONU 設定較高的 PQS 和 PR，使其能累積較多的 permit 或是更快累積 permit，當輪到後端 ONU 傳送時可以一

次送出較多的封包，而不至於造成過長的延遲。

2.2 控制動態頻寬分配之困難

在此架構下，由於頻寬資源是從第一個 ONU 分配至最後一個 ONU，且於同一個 cycle，前端之 ONU 無法得知後端 ONU 頻寬之需求。因此，最直接的做法即是由 ONU-2 開始，以 ONU-1 之 MD 判斷，若 $MD_2 < MD_1$，則縮減(調低)ONU-2 之 PQS 或 PR，直到 MD2 = MD1；反之，若 $MD_2 > MD_1$，則放寬(調高)ONU-2 之 PQS 或 PR，使 MD2 = MD1；以此方式由 ONU-1 至 ONU-N，逐步使後端 ONU 之 MD 皆等於 MD_1，達到最初步之公平原則，亦即平均而言所有 ONU 封包皆處在相同等級之 delay。

但若深究此種做法，可以發現為了使所有 ONU 之 MD 相同就需花費相當心力，且需人力耗時守在機器前手動調整。再者，若要達到進一步的 QoS，提高整體 throughput，此種做法將會變得不可行，因為包括 PR、PQS、MD 皆為連續實數，以人工方式進行調整相當不切實際。因此，我們將 PQS、PR、MD 視為一最佳化問題，換言之，找尋每個 ONU 的 PQS 與 PR 使得所有 ONU 之 MD 可以彼此相等且最小的最佳解。循此思維，我們認為非支配型排序基因演算法(Non-dominated Sorting Genetic Algorithm-II, NSGA-II)的特性，可以幫助我們快速且自動地解決這個最佳化問題，將於下個章節介紹。

3 非支配型排序基因演算法-II

3.1 基因演算法

基因演算法[17]的想法來自於達爾文的進化論，遵循「適者生存，不適者淘汰」的原則，仿效大自然演化的方式，用於科學上，在廣大的解空間裡，經由搜

尋並剔除不適當的解進而保留適當的解，以解決問題並為最佳化問題找到最佳的解。如同生物的演化論，基因演算法將問題的可能組合放入染色體，或稱為基因(gene)，每一個個體(individual)擁有自己獨特的基因，並給予個體用以評估基因好壞程度的適應值(fitness)，經由挑選(selection)、增殖(reproduction)、交換(crossover)、突變(mutation)等過程，組成下一代。而所謂演化就是指每個世代(generation)間基因群體的變化，適者生存是使下一代的基因排序優於上一代而更適合生存，用於解決最佳化問題則代表更有機會找到最佳解，基因演算法的優點在於多點搜尋，避免問題陷入區域(local)最佳解。一般基因演算法的流程如圖3.1：

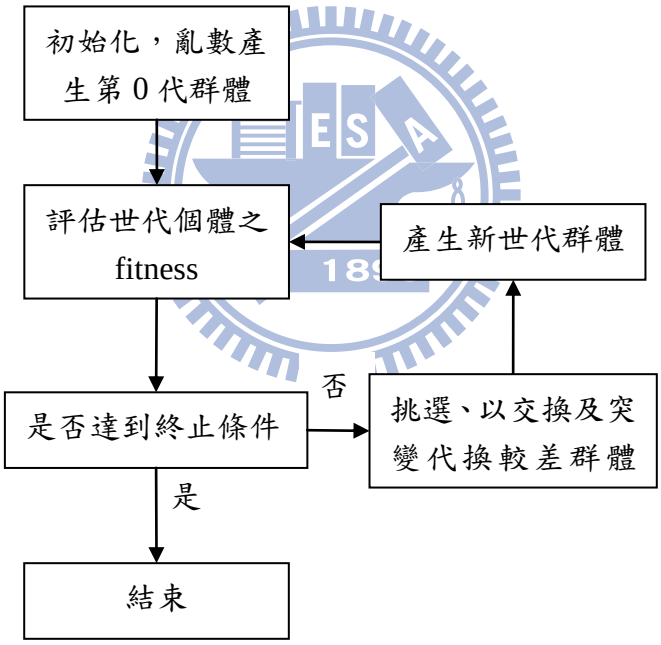

圖 3.1 基因演化法流程圖

- Step 1：初始化產生第 0 代群體(population)，通常由可能的解空間中使用均勻分布隨機產生(uniform distribution in feasible region)，或是人為針對問題使用已知的知識(pre-known knowledge)給予初始值，使其能更快收斂，但也可能使問題解陷入部分區域產生偏差解(solution bias)而無法順利找到整體最佳解。

- Step 2：評估個體基因優良與否，一般是由針對問題而設計之 fitness 做判斷，其中一種做法是給予排序(rank)，經由挑選，留下較好的解於下步驟產生後代。其中，若保證該世代中最佳個體能夠存活至下世代，稱為含精英政策(elitism)。值得一提的是，產生個體之 fitness 並評估的動作，通常決定了整個基因演算法的運算時間。
- Step 3：對決定保留之群體進行增殖，包括交配與突變。交換是指將兩個個體的染色體做交換合併而產生子代(offspring)，藉此希望能產生 fitness 更高的個體，雖有可能產生較差的個體，但最後仍會經由選擇的機制被淘汰。突變則是對個體之染色體做大幅度的改變，使其偏離個體原本的傾向，藉此搜尋其他可能解的區域，避免問題陷入區域最佳解。用這些新產生的群體，替換掉 Step 2 中 fitness 不良的群體，就是代換(replace)的動作。
- Step 4：判斷是否已達到終止條件，若否，則繼續產生新世代，反之則結束該演化過程。終止條件一般會由使用者設定欲模擬之世代數量，或是判定世代中個體之 fitness 是否已經無法再做改善。

以上介紹的基因演算法是最為基礎的原型，但我們的頻寬分配機制並非只需找到單一 fitness 之最佳解，例如：我們可能需要所有 ONU 之 MD 相同，但 MD 又要為最小值，這樣就成為兩個最佳化問題。針對有多個目標需要最佳化之問題，需要先介紹下節的多目標最佳化問題(Multi-Objective Optimal Problems)。

3.2 多目標最佳化問題

圖 3.2 為決策空間(decision space)映射至目標空間(objective space)之示意圖，能簡單說明何謂多目標最佳化問題[18][19]。多目標最佳化問題將決策解以適應函式(fitness function)映射至目標空間，於目標空間中以適應值判斷決策解之優劣。日常生活中，我們對單一物品之價值可能有不同面向的考量，若以買車為例，目標空間橫軸之 fitness 代表外觀，縱軸之 fitness 代表價格之倒數，則點”1”代表外

觀較差但價格較低，點”2”表示有較佳的外觀但價高，價格高低與外觀優劣是適應值，低價格與優良外觀則是最佳化問題的目標。

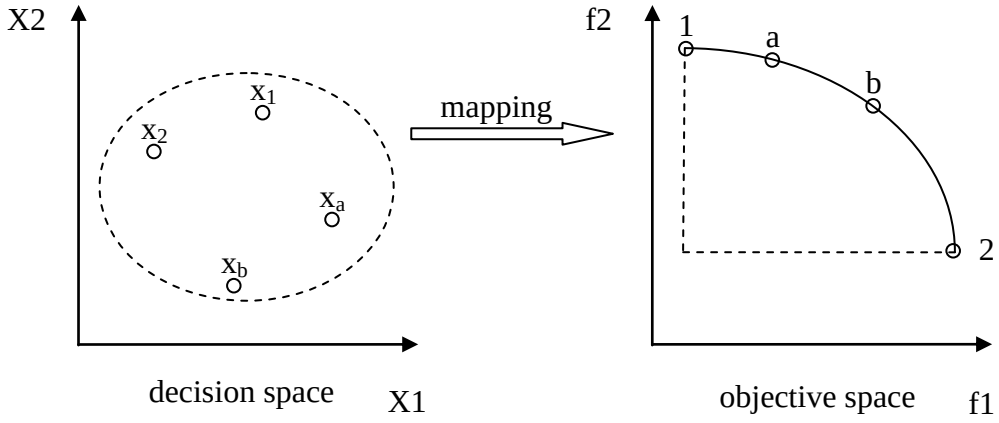

圖 3.2 說明多目標最佳化問題之簡例

數學上的定義如下，藉著此定義設定等式及不等式，最佳化即可用來解決多目標最佳化問題：

$$\min_{x\in\Re^n}\{y = f(x): h(x) = 0, g(x) \le 0, x^L \le x \le x^U\} \qquad \text{(式 3.1)}$$

y：目標空間中對應決策向量之目標向量(objective vectors)

x：最佳化問題之決策向量(decision vectors)

f(x)：適應值函式(fitness function)

h(x)：等式限制(equality constraint)

g(x)：不等式限制(inequality constraint)

x^L, x^U：分別表示決策解之上下限(variable bounds)

圖 3.2 中，目標空間點”1”至點”2”連線上任兩點，以兩個最佳化目標而言，並沒有任何一點可完全支配(dominate)另一點，(價格較優者外觀必定較另一點差，反之亦然)，因此陷入需要取捨(trade-off)的狀況。於是，在多目標最佳化問題中，點”1”至點”2”的連線被定義為 Pareto-optimal front：表示在目標空間中，找不到

其他點可於兩個最佳化目標皆優於該 front 上的某點，該連線同時也被稱做非支配集合(non-dominated set)，而決策空間中對應非支配集合之決策解集合則稱為 Pareto 最佳化集合(Pareto optimal set)。

非支配型排序基因演算法-II (Non-dominated Sorting Genetic Algorithm-II, NSGA-II)結合基因演算法與多目標最佳化問題，適合用來處理我們的頻寬分配機制，將於下節介紹。

3.3 非支配型排序基因演算法-II

非支配型排序基因演算法-II[20][21]是一種多目標基因演算法(Multi-Objective Genetic Algorithm, MOGA)，其擁有精英保留策略，以及維持決策解之多樣性的機制，並且可用來處理多目標最佳化問題，因此被我們選擇用來配合頻寬分配機制。圖 3.3 為 NSGA-II 流程圖，並說明如下：

- Step 1：結合 N 個親代 Pt 個體和 N 個子代個體 Qt 為 Rt，並且對 Rt(總個體數為 2N)執行非支配排序(non-dominated sorting)，同時依序為每一個 Pareto front 命名為 Fi。
- Step 2：設定下一代親代 P_{t+1} 為空集合，依序放入 $F_1, F_2, F_3, \ldots$，直到若將 F_i 放入後會使 P_{t+1} 總數超過 N 個，則不放入 Fi，並進行 Step 3。
- Step 3：對 Step 2 中之 F_i 進行擁擠距離排序(crowding distance sorting)，選擇分散度最高之前$(N - |P_{t+1}|)$個體放入 P_{t+1}，至此選定 P_{t+1} 中之 N 個個體。

- Step 4：對 P_{t+1} 世代執行擁擠度競賽選擇(crowded tournament selection)、交換、突變以產生新子代 Q_{t+1}。

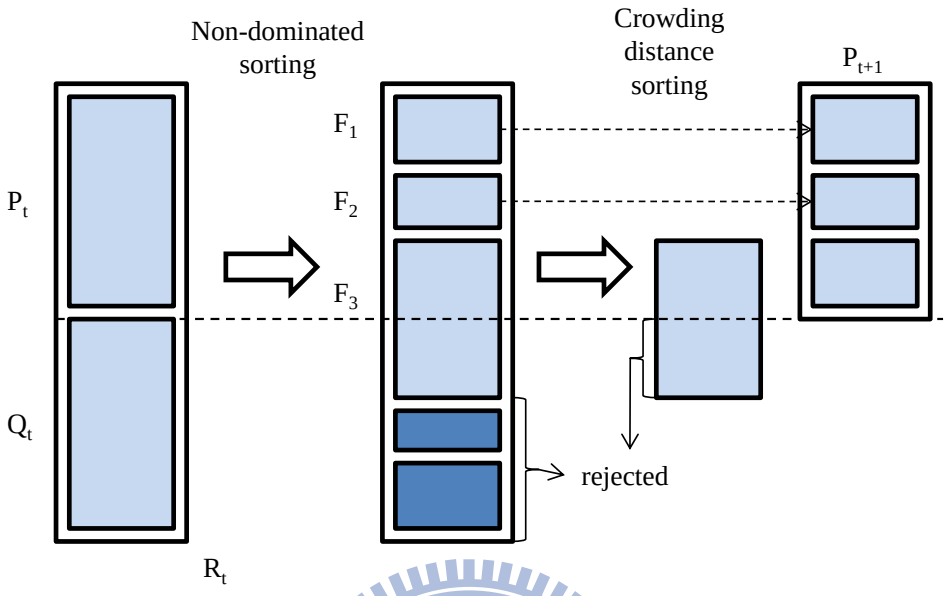

圖 3.3 NSGA-II 流程

Step 1 中之非支配排序的目的是將 R_t 依照非支配集合的好壞分類排序，也就是說，Pareto front F_1 上的所有解皆會優於 Pareto front F_2，對於 Fi 給予排序(rank) $r_i = i$，藉此可以更容易決定哪些優秀的個體可以被留下來，NSGA-II 就是以此法做為精英政策(elitism)。非支配排序的做法如下：在 R_t 中兩兩比較個體的 fitness，找到 F_1 後，先將 F_1 剔除於 R_t 外，再於剩餘的集合(R_t - F_1)中尋找下一個 Pareto front F_2 並剔除 F_2，以此類推，就能將 Rt 中所有個體分類排序至各 front。

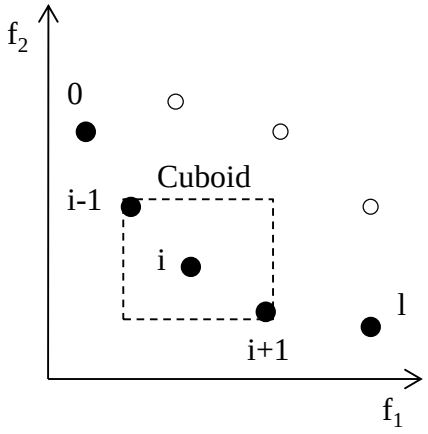

圖 3.4 擁擠距離算法

Step 3 中之擁擠距離排序是為了保存決策解的多樣性(diversity preservation)，若我們想要以有限數目的決策解之連線來表示所有可能的最佳解，那麼這些解彼此間的距離越大，越可能達到這個目的，多樣性同時也跟突變一樣可以確保基因演算法搜索決策解的能力。如圖 3.4，以兩個目標問題為例，所有實心的點皆處在同一 front 上，對於解 i 之擁擠距離，首先以相鄰解 i 之解 i-1、i+1 構成方形面積(虛線)，相對於解 i 之擁擠距離 d_i 就定義為該方形之邊長的平均值。對於超過兩個目標的多目標問題，NSGA-II 以下式計算擁擠距離，其中 M 為目標的數量，其意義為：解 i 之擁擠距離 d_i，為相鄰之兩解 i+1、i-1 於各目標方程式上 fitness 之差分別標準化後相加。

$$d_i = d_{i,m} + \frac{f_{m(i+1)} - f_{m(i-1)}}{\max(f_m) - \min(f_m)}, \text{for } m = 1,2,\ldots,M \qquad \text{(式 3.2)}$$

Step 4 中之擁擠度競賽選擇，首先將親代 P_t 兩兩選為一對舉行競賽，接著比較其排序 r_i，較小者獲勝；若 r_i 相等則表示兩個體處於同一 Pareto-optima front，此時再比較擁擠距離 d_i，d_i 較小者因代表該個體較能保存群體之多樣性而獲勝。在擁擠度競賽選擇中獲勝之個體，接續進行交換與突變的動作產生子代。

3.4 處理大量 fitness 所產生的問題

上一節介紹之 NSGA-II 具有保存優良個體以及維持群體多樣性的特點，並且具有處理多目標最佳化問題的能力，藉此，我們可以將動態頻寬分配機制轉換為多目標最佳化問題處理。但如 3.1 小節所提到的，基因演算法中需要最多時間進行運算的部分是計算個體的 fitness，隨著問題的規模和複雜度提高，其所需的搜尋時間將大幅增長，若採用演算法中以串列(series)的方式逐步進算每個個體的適應值，所要計算的 fitness 數量將是世代與群體之積。因此我們考慮，搭配平行運算(parallel computing)，將群體的 fitness 計算以平行的方式多個同時進行，將能節省大量的模擬時間，我們採用的方法即是下個章節介紹的訊息傳遞界面(Message Passing Interface)。

4 訊息傳遞介面

4.1 MPI

MPI (Message Passing Interface)[22][23]是第一個標準化的 Message Passing 平行語言，可以支援多種程式語言，包括 Fortran、C、C++等，MPI 的目標是提供使用者可靠、有效率的訊息通訊介面，其可用在分散式記憶體(distributed-memory)架構或是共用記憶體(shared-memory)的叢集(cluster)運算架構上。MPI 是一個標準(standard)，適用於各種不同的傳遞界面，將電腦視為一個主機(host)，host 上具有運算能力的核心都視為一個處理單位 process，且不論這些 process 是否位於同一個 host 上，都能為其提供通訊，處理彼此的資料交換。

4.2 MPICH2

MPICH2[24]則是符合 MPI 標準通訊協定所衍生的一套軟體，其提供使用者可選用的函式庫以及工具，經由這些函式和軟體，使用者可以在不同的平台

(platform)上使程式有效率地平行化，其主要幾個函式介紹如下：

- MPI_Init()：初始化 MPI 環境，包含所有 process 及提供傳遞訊息所需的資料，每次平行化程式開始時都要執行。
- MPI_Finalize()：平行化程式結束之前呼叫。
- MPI_COMM_WORLD：MPI 內定的 communicator，辨認所有加入平行運算的 process，屬於同一個 communicator 的 process，才能進行訊息傳遞與通訊。
- MPI_Comm_size()：取得參與平行運算的 process 數目。
- MPI_Comm_rank()：取得參與平行運算的 process 之名稱(id)，當想要在平行程式內分配工作給不同的 process 時，可以根據 id 呼叫指定的 process 執行。
- MPI_Send()：MPI 點對點通訊(point-to-point communication)的指令之一，可以將欲傳遞的資料由指定的 process 送至另一 process。
- MPI_Recv()：MPI 點對點通訊(point-to-point communication)的指令之一，可以接收由指定的 process 送來的資料。
- MPI_Bcast()：MPI 集體通訊(collective communication)的指令之一，Bcast 是廣播(broadcast)的縮寫，可將資料由指定的 process 送至 communicator 內所有的 process。

透過以上所介紹的指令，加入 NSGA-II 模擬的程式中，假設我們有 10 個 process，就可以將基因演算法中，每 10 個個體為一組，在同一段時間內平行運算，理想的模擬時間就會縮減為原本的十分之一。下一章，將介紹我們的動態頻寬分配機制如何整合 NSGA-II 與 MPI，提供使用者完整的服務品質方案。

5 平行化多目標基因演算法動態頻寬分配設計

在本章我們提出一整合多目標基因演算法與平行化計算的動態頻寬分配機制，用於下世代 OFDMA-PON 上，以下將依序介紹頻寬分配機制與 NSGA-II 之整合、NSGA-II 與 MPI 之整合以及如何進行動態頻寬分配。

5.1 整合 DBA 與 NSGA-II

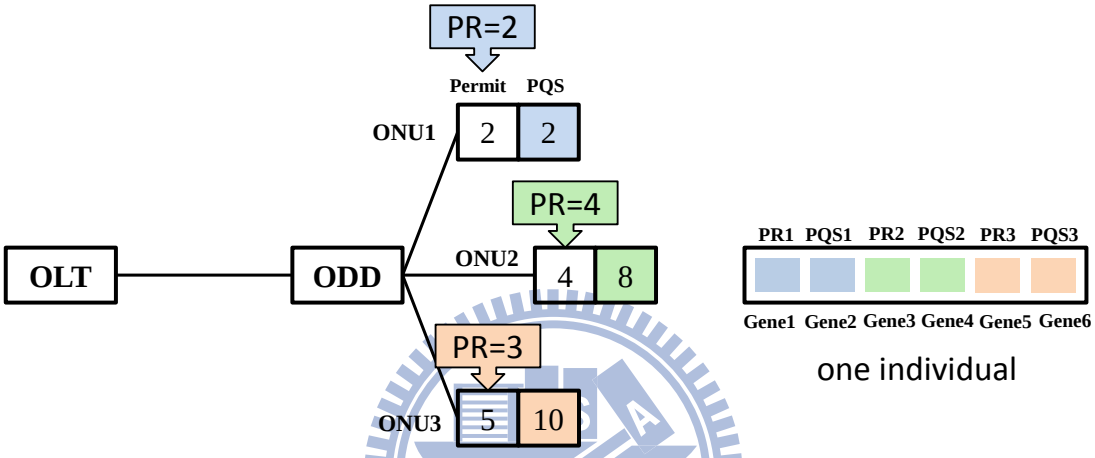

圖 5.1 mapping between (PR, PQS) and NSGA-II

整合 NSGA-II 的首要步驟是將(PR, PQS)所在的 decision space 轉換至 NSGA-II 所在的 objective space，如此，在 objective space 上就能直接以 objective 的 constraint 與 fitness function 經由多目標基因演算法進行處理，最直接的做法是依序將每個 ONU 的 PR 與 PQS 放入基因演算法中個體的基因序列，如圖 5.1。隨後，以這樣的設定進行 3.3 節介紹的 NSGA-II，經由 non-dominated sorting 與 crowding distance sorting 選擇存活到下一世代的 parent，再由 crowded tournament selection 選擇產生子代的配偶，經由最後 crossover、mutation 產生新的子代，如此持續進行，直到指定的世代數或是 fitness 已經無法再有進步的情況。

此處必須注意的是，一般基因演算法使用的 crossover 與 mutation 是使用 binary-coded，以有限位數的 binary string 來表示實數，例如以 7 個位元數來表示 (3, 130)之間的數值，那麼精準度就是

$$\frac{(130-3)}{1111111_B-0000000_B}=\frac{127}{127}=1 \qquad \text{(式 6.1)}$$

對於我們的 PR 與 PQS 來說，這樣的精準度尚嫌不足，因此我們採用 Simulated Binary Crossover(SBX)[25]來做為 real-coded 的 crossover。

在介紹 SBX 之前，必須先了解 binary-coded 中 one-point crossover 的特性，

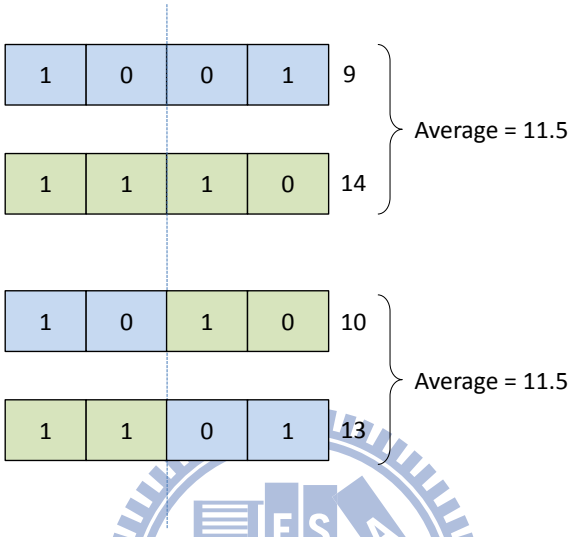

圖 5.2 one-point crossover

one-point crossover 的作法是先將兩個親代取一相同的位元間隙，由此間隙選擇一邊將兩段基因交換，如圖 5.2 上方兩段基因即是親代，下方即是產生的子代。這裡可以發現 one-point crossover 的第一個特性：兩個親代的平均值會與兩個子代的平均值相同，且此特性不會因為選擇的位元間隙而改變。

說明 one-point crossover 第二個特性之前，必須先介紹一個參數 Spread factor(β)：

$$\beta=\left|\frac{c_1-c_2}{p_1-p_2}\right| \qquad \text{(式 6.2)}$$

如圖 5.3，β 是用來表示一對親代產生之子代後，子代與原本親代的關係，$\beta<1$ 表示子代位於兩個親代之間，$\beta>1$ 則位於兩個親代之外，$\beta=1$ 則是剛好與兩個親代相同，

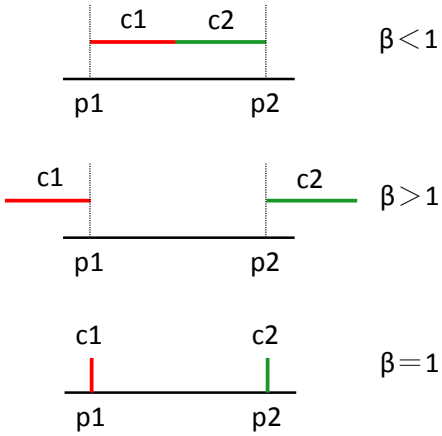

圖 5.3 spread factor

而對於位元數為 15 的基因組合來說，其 spread factor 的分布如圖 5.4，此為 one-point crossover 的第二個特性，可以發現 spread factor 幾乎都集中在 1 附近，

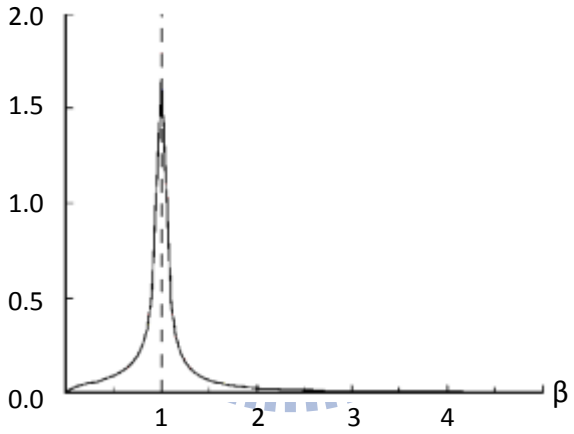

圖 5.4 distribution of spread factor of 15bits binary code

因此 SBX 採用一 polynomial 機率分布用來模擬該 spread 分布：

$$c(\beta) = 0.5(n+1)\beta^{n}, \beta \le 1 \qquad \text{(式 6.3)}$$

$$c(\beta) = 0.5(n+1)\frac{1}{\beta^{n+2}}, \beta > 1 \qquad \text{(式 6.4)}$$

當 n＝2 時，其分布如圖 5.5，SBX 的作法，即是各取兩親代之一個 gene，依照該機率分布，隨機取一 spread factor，再以式(6.5、6.6)代入，即可用來進行 crossover 以產生子代，且兩子代之平均與兩親代之平均相等，符合第一個特性。

$$c_1 = 0.5(p_1 + p_2) - 0.5\beta(p_1 - p_2) \qquad \text{(式 6.5)}$$

$$c_2 = 0.5(p_1 + p_2) + 0.5\beta(p_1 - p_2) \qquad \text{(式 6.6)}$$

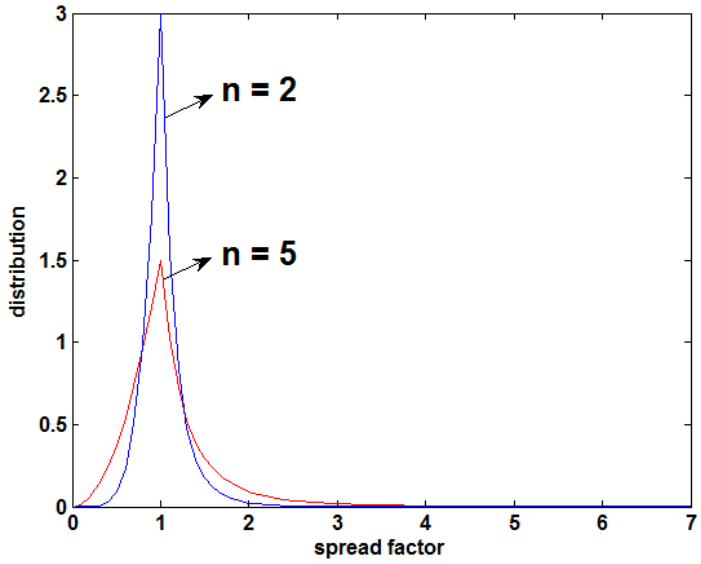

圖 5.5 probability distribution of polynomial model

此外，mutation 的部分，則採用下式，c 是子代，p 是親代，δ則與圖 5.5 之分布相同，唯中心點由 1 調整至 0。

$$c = p + \delta(x_{max} - x_{min}) \quad \text{(式 6.6)}$$

如此，經由設定最佳化問題之 constraint，我們就可以將連續實數最佳化問題(PR, PQS)轉至多目標基因演算法處理。

5.2 整合 NSGA-II 與 MPI

多目標基因演算法中，最耗時間計算的部分在於 fitness，因此我們以圖 5.6 的方式，先設定 MPI 中 process 的數量，給予每個 process 專屬的 id，並將 process 0 設定為 master，其餘則為 slave，由 master 進行 NSGA-II 主體程式，每當要進行計算 fitness 時，則將所有個體的基因組合經由廣播(綠色箭頭)告知從屬 process，並分配各 slave 應計算的部分，由這些 process 進行運算並回傳(紅色箭頭)給 master。

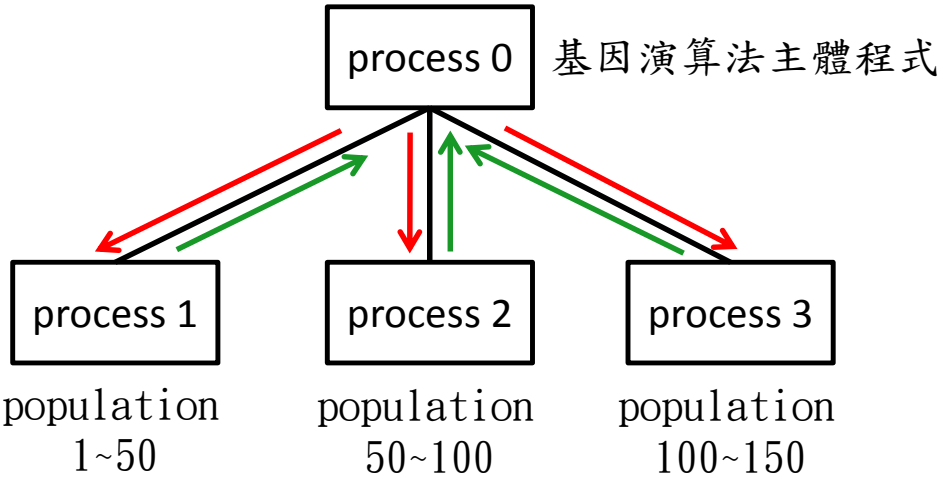

圖 5.7 MPI 平行化計算 fitness 示意圖

5.3 整合 NSGA-II 與 MPI 之動態頻寬分配機制

整合 NSGA-II 與 MPI 之動態頻寬分配機制，流程說明如下以及圖 5.7：

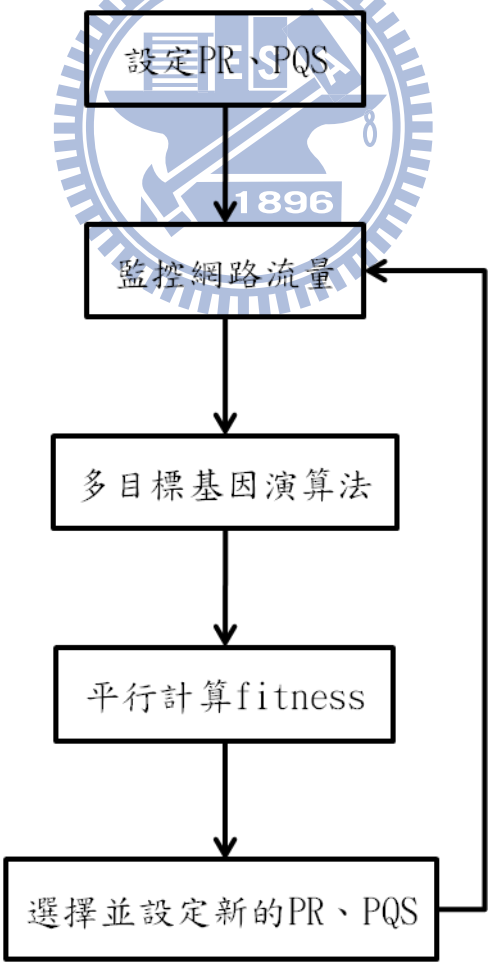

圖 5.8 整合多目標基因演算法與 MPI 之動態頻寬分配機制流程圖

- Step 1：初始化。監控 OFDMA-PON 中各 ONU 封包之流量(traffic)狀態，設定 OFDMA-PON 子載波數目；設定 NSGA-II 之群體數與世代數；設定 MPI 中 process 的數量，給予每個 process 專屬的 id，並將 process 0 設定為主控(master)，其餘則為從屬(slave)。
- Step 2：將 ONU 中欲調整之 PQS 與 PR 放入 NSGA-II 的基因序列中，並設定上下限；設定 NSGA-II 的適應值方程式以及限制，這些方程式產生的適應值用來判斷頻寬分配機制給予的 PQS、PR 是否能滿足使用者者的服務品質。因此，NSGA-II 中個體的基因即為 PQS 與 PR 之組合，個體的適應值代表該基因組合設定的 PQS 與 PR 對於頻寬分配機制的優劣。
- Step 3：開始進行 NSGA-II。由 process 0 進行 NSGA-II 主體程式，每當要進行計算 fitness 時，則將所有個體的基因組合經由廣播告知從屬 process，由這些 process 進行運算並回傳給 master。
- Step 4：當達到設定的世代數目後，NSGA-II 結束，MPI 結束。找出最後一個世代中適應值最優個體，將其基因組合做為 Step 1 中 PQS 與 PR 的設定。並持續監控各 ONU 網路流量，一旦開始有 throughput 降低或不 fair 的狀況發生，則進入 Step 1 開始調整頻寬分配機制。

經由以上的流程，將頻寬分配機制轉換為多目標最佳化問題，設定適應值方程式以及限制，並且以基因演算法處理之，NSGA-II 就能找出可能的最佳(PQS, PR)組合，MPI 則用來加速運算 NSGA-II 的適應值，即時給予使用者頻寬分配的設定。且每當 ONU 有不同的頻寬需求時，我們也可以根據當時的網路流量以及延遲需求更改適應值方程式和其限制，提供使用者不同的服務品質。

6 實驗結果與討論

6.1 實驗設置

首先，對於 OFDMA-PON 我們有以下的設置：

Bandwidth (number of subcarriers)	512
Number of ONU	32
Traffic load	0.9~0.95
Burstiness	8.0~16.0
Simulation time (number of cycle)	10^8
Number of process in MPI	32
Generation	30
Population	60
Probability of mutation	0.08
Probability of crossover	0.9

需要注意的是 Burstiness 這個參數，對於每個 ONU，為了能模擬實際封包至各個 ONU 的狀態，我們以 two-state bursty traffic with Poisson distribution 做為流量模型(traffic modal)，如圖 6.1。

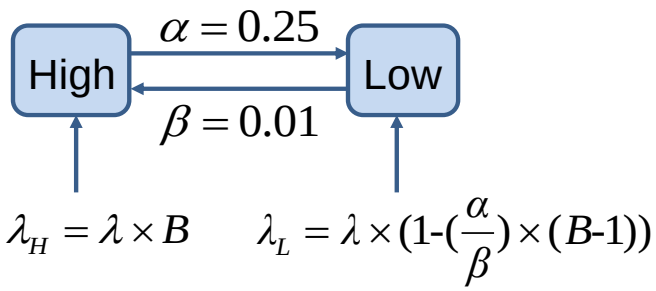

圖 6.1 bursty traffic 示意圖

參數設定如下：

$p(X = k) = \frac{e^{-\lambda} \times \lambda^{k}}{k!}, k = 0,1,2, \ldots$：封包進入 ONU 的機率為 Poisson 分布；

λ :進入 ONU 的平均封包數目 (average arrival rate)；

λ_H, λ_L：分別表示高流量和低流量狀態的平均封包數量；

α, β：分別表示從高流量轉至低流量狀態、低流量轉至高流量狀態的機率；

參數彼此關係如下：

$$\lambda = \frac{\lambda_H \times \frac{1}{\alpha} + \lambda_L \times \frac{1}{\beta}}{\frac{1}{\alpha} + \frac{1}{\beta}} \quad \text{(式 6.1)}$$

$$B = \frac{\lambda_H}{\lambda} \quad \text{(式 6.2)}$$

圖 6.1 說明 bursty traffic 如何模擬網路中封包實際的流量狀況，我們假設網路流量分別有高流量與低流量兩種狀態，平時處於低流量狀態，忽然有大量資料要傳送時才會轉換至高流量狀態，(式 6.1)即用來計算在這種狀況下進入 ONU 之平均封包數目，其中 α、β 之倒數的意義分別是高流量低流量狀態停留的時間；(式 6.2)則如圖 6.2 所示，Burstiness(B)用來形容封包到達 ONU 的狀況是否密集，case2 的 Burstiness 大於 case1，且在接近使用者端時，case2 較符合網路流量的狀況。

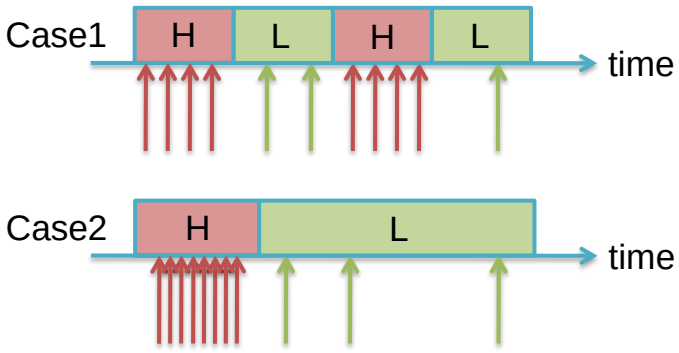

圖 6.2 Burstiness 之意義

此外，對於每一個 ONU，都是分別給予一個獨立之 traffic modal，因此 traffic

load＝0.9，則代表對於每個 ONU，在每個 cycle，流入每個 ONU 的平均封包數為式(6.3)：

$$\lambda = \frac{512(\text{bandwidth}) \times 0.9(\text{traffic load})}{32(\text{number of ONU})} \qquad \text{(式 6.3)}$$

這種設計的好處是，我們可以根據不同的 ONU 模擬不同的網路流量，並且可以根據該 ONU 要求的 QoS，給予所需的頻寬分配機制。有了本節的說明，我們就可以開始進行模擬結果的討論。

6.2 手動調整 PQS、PR

圖 6.3、6.4 尚未使用 NSGA-II，單純以手動固定 PQS、變動 PR，希望能到使所有 ONU 之 MD 相等的組合。可以發現三個問題：(1)當固定 PR，調整 PQS 時，手動有一定的極限，觀察 MD 降到 20 與 10 時，後端 ONU 之 PQS 所需調幅已經太大，其解可能不存在也無法輕易取得；(2)MD、PQS、PR 皆為連續實數，若我們想知道能使所有 ONU 之 MD 皆為相等且最低的值，手動調整不可行；(3)同時調整 PQS 與 PR 十分困難。因此，我們需要將問題轉換為多目標最佳化問題，由 NSGA-II 的人工智慧代替手動來處理。

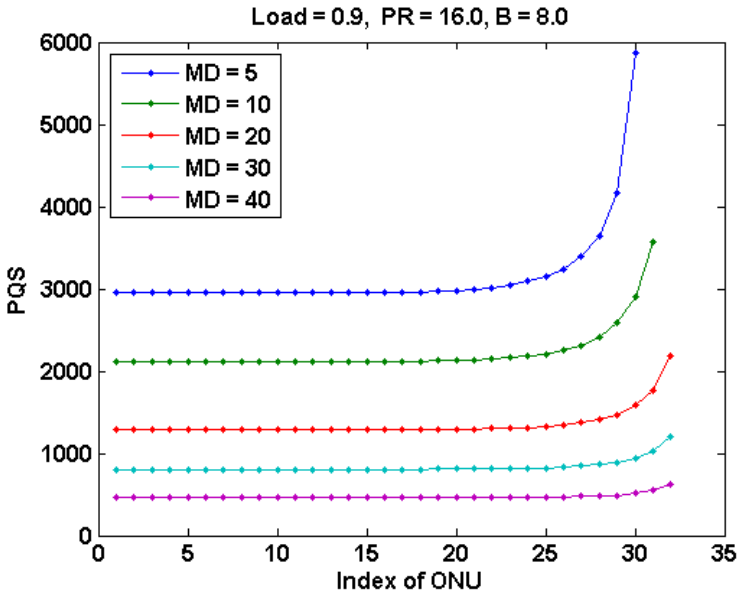

圖 6.3 手動調整 PQS，固定 PR

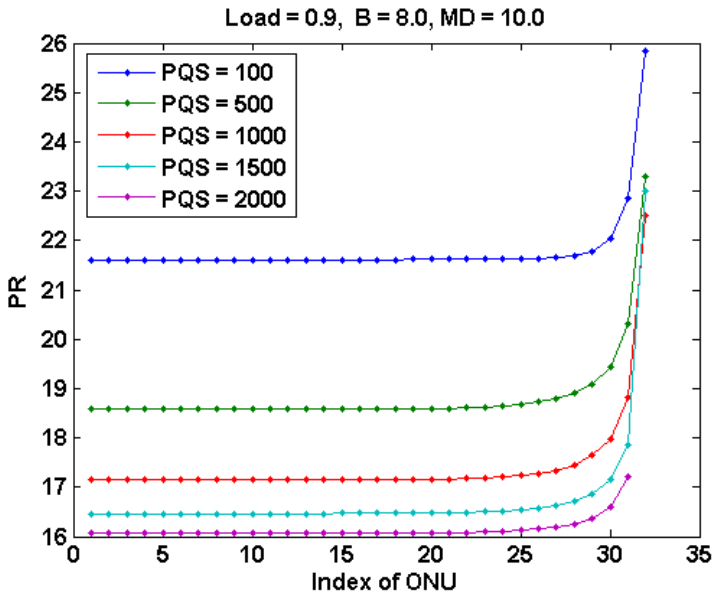

圖 6.4 手動調整 PR，固定 PQS

6.3 以 PQS 做為 Gene

在此階段，我們開始將 MOGA 使用在解決 DBA 的問題上，實驗設置如下：

(i) Load = 0.9；B = 8.0

(ii) Generation = 30；Individual = 60

(iii) Input variable：最後 10 個 ONU 的 PQS

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$ = 所有 ONU 之 MD 對 $\overline{\mu_{10}}$ 的標準差，後 10 個 ONU 給予 weight = 1～10，最後以 $\overline{\mu_{10}}$ 標準化

(v) PR = 16.0 for all ONU; $PQS_{1\sim22}$ = 500; $PQS_{23\sim32}$ = Output of MOGA

需要討論的部分在於設置(iv)中，我們在 MOGA 中 multi-objective 的目標是最小化 fitness，所以我們設定的 fitness 需要能表達出其值越小，效果越佳。所以，fitness 1 是所有 ONU 之 MD 的平均，其值越小則整體 throughput 越高；Fitness 2 是將所有 ONU 之 MD 與前 10 個 ONU MD 之平均做標準差的計算，因為前 10

個 ONU 的 MD 在我們的 traffic 設定下基本上會相等，並且對後端 10 個 ONU 之 MD 做 weight，越接近尾端越加重，目的在針對造成 MD 不平的主因做處理，且 fitness 2 越小，代表 MD 曲線為平整的機會越大，換言之，fairness 會越高。

而在設置(v)中 PR 之設定是將 512 個 subcarriers 平均分給 32 個 ONU，則每個 ONU 在一個 cycle 內可以得到 16 個 subcarriers；前端 22 個 ONU 之 PQS 固定為 500，是考慮在 6.2 節中，手動調整之 PQS 分布結果，前 22 個 ONU 之 PQS 是相等的。

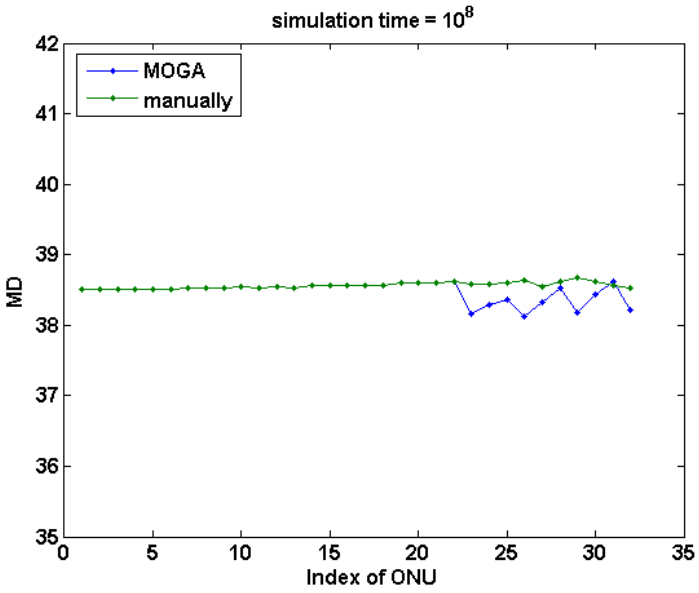

圖 6.5 MOGA 以 PQS 做為 Gene

結果如圖 6.5，MOGA1~3 是挑選 MOGA 最後一個 generation 中幾個較好的結果，與手動調整之 MD 曲線比較，結果並不理想，後端 10 個 ONU 之 MD 扭曲過大。觀察圖 6.6，可以發現 MOGA output 與手動調整選擇之 PQS 差距雖然不算大，但結果差強人意，並且若對 QoS 之要求有所改變，則需要再增加 input 的數目，對 MOGA 而言並非好事。因此，在下一小節的討論，我們將對 input variable 做調整。

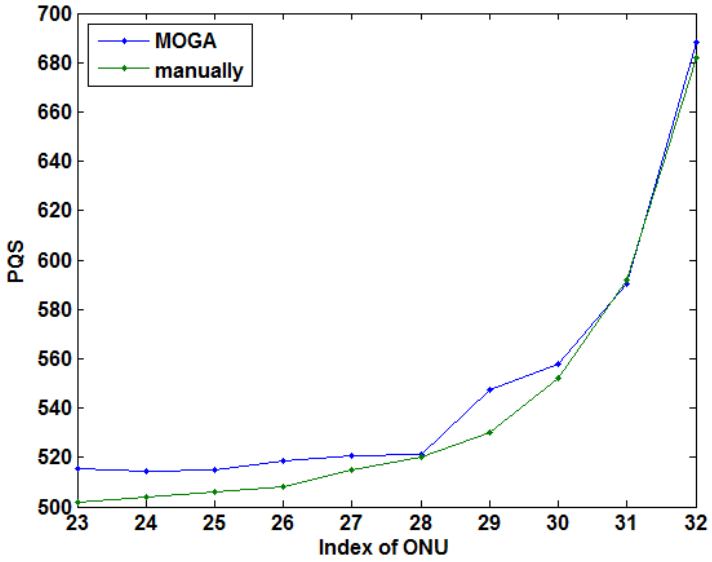

圖 6.6 比較 MOGA output 與手動調整之 PQS

6.4 以 exponential 參數代替 PQS 做為 Gene

(i) Load = 0.9；B = 8.0

(ii) Generation = 30；Individual = 60

(iii) Input variable：

PR = 16.0 for all ONU

$PQS_i = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～20；b = -20～0；c = 0～5；d = -1～1；e = 500～2000

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

延續 6.4 節的討論，我們觀察圖 6.3、圖 6.4，思考若以某種 function 表示各 ONU 之值，以這個 function 的參數做為 MOGA 的 gene，有兩個好處：一是能減少 MOGA 之 input 數，二是該 function 能比直接帶入 PQS 為 gene 來的平滑。以圖 6.6 來看，PQS 之分部可視為一平滑曲線，因此我們首先考慮以 exponential

function 來 fit PQS 曲線，利用 Matlab 的"curve fit tool"，並且與原本手動調整之曲線比較，如圖 6.7~6.9。圖 6.7 使用的 function 是 exp(a*i + b) + c，圖 6.8 則是 exp(a*i + b) + exp(c*i + d) + e，觀察兩圖，curve fit 之好壞程度差別不大，因此圖 6.9 將兩者 fit 之值直接以 simulation 比較最後之 MD 曲線，可以發現 5 個參數的 exponential 函數，其 fairness 明顯較高，因此我們選定了 5 個參數的版本。

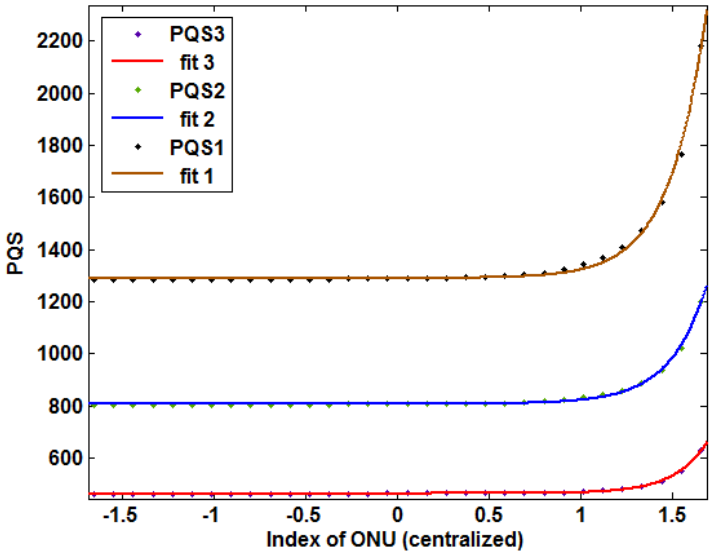

圖 6.7 以 exp(a*x+b)+c fit 手動調整之 PQS

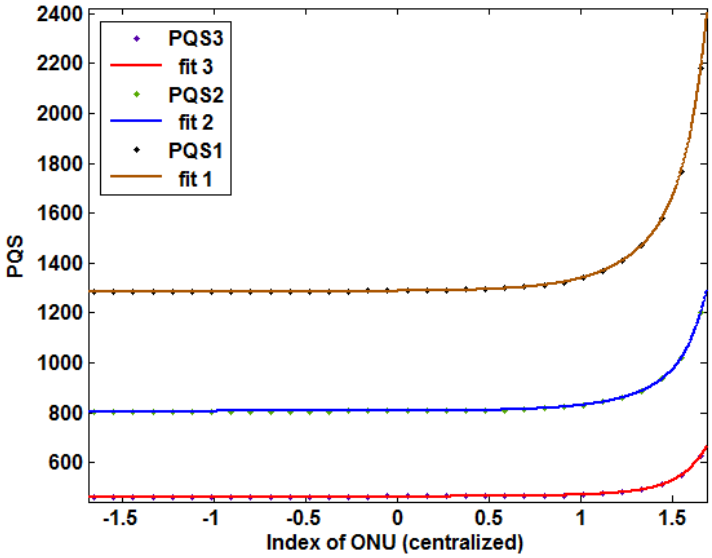

圖 6.8 以 exp(a*x+b)+ exp(c*x+d)+e fit 手動調整之 PQS

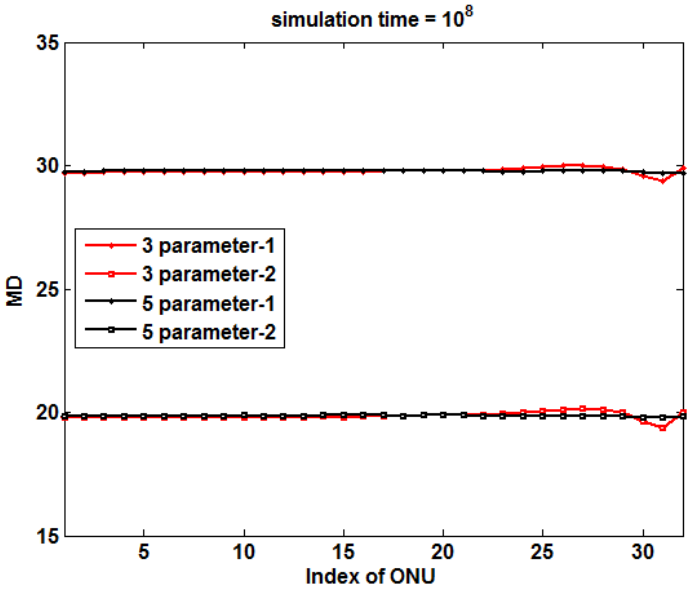

圖 6.9 比較兩種 fit 方式產生之 MD

在 exponential function 五個參數(a～e)的選擇上，是以 matlab 對手動調整的幾條曲線選定的值，將其涵蓋的範圍稍微擴大，希望能盡量包含所有曲線的可能性，但又不會對 MOGA 產生太大的負擔。實驗結果如下：

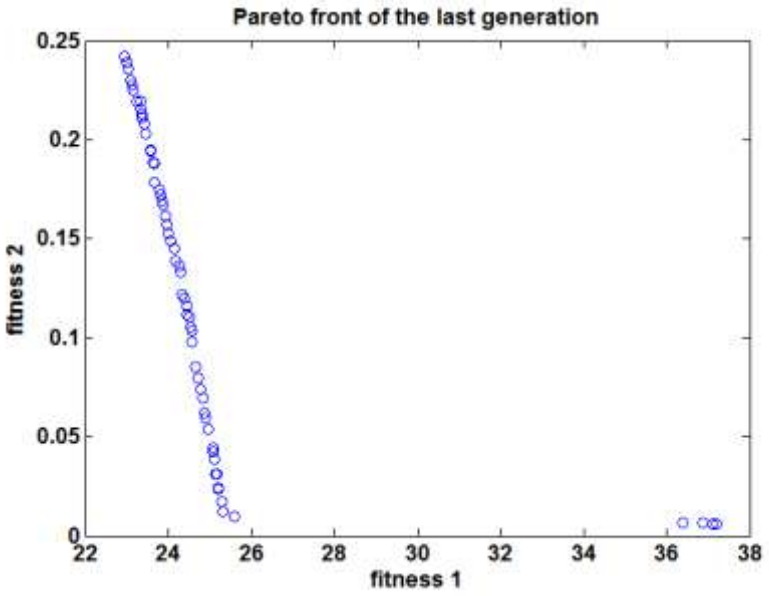

圖 6.10 Pareto front of exp. PQS

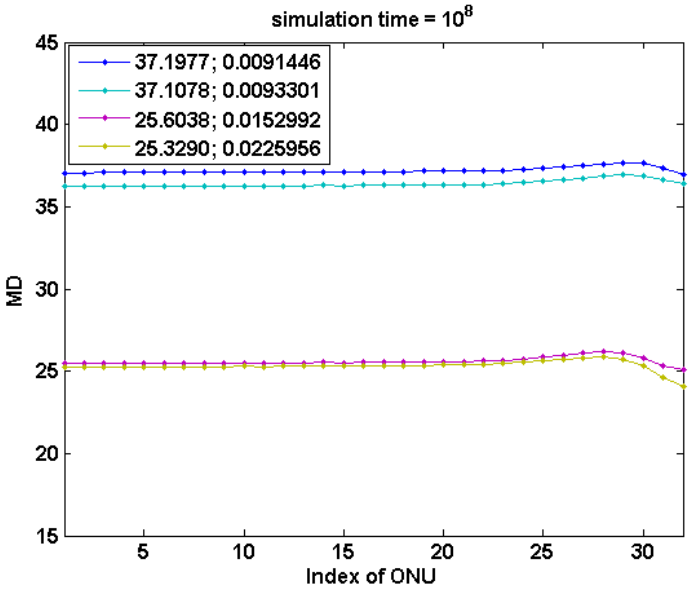

圖 6.11 exp. PQS 挑選之結果

圖 6.10 為 MOGA 最後挑選結果之 Pareto front，圖右下方的 individual 代表其有較高的 MD 但 deviation 較低，符合我們原先期望的結果；同時將圖 6.11 與圖 6.5 比較，可發現利用 exp. PQS 能得到較低的 MD 並同時兼顧 fairness。

不過在這個設定中，觀察圖 6.10，可以發現 individual 大多集中在低 MD 的區塊，高 fairness 的數目並不多，因此我們希望能對這個部份做出改善。

6.5 對 fitness 2 加入 constraint

針對 6.4 節產生的問題，我們在 MOGA 中，採用一個新的功能，對 Fitness 增加一個 constraint，MOGA 會依據該 constraint，盡可能保留符合 constraint 的 individual，我們希望藉由這樣的限制，找到較多低 MD 高 fairness 的 individual。並且在這個階段，我們採用三種不同的 PQS 與 PR 組合，測試何者結果較適合我們的 DBA。

- Method 1：exponential PQS，固定 PR = 16.0

(i) Load = 0.9；B = 8.0

(ii) Generation = 30；Individual = 60

(iii) Input variable：

PR = 16.0 for all ONU

$PQS_i = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～20；b = -20～0；c = 0～5；d = -1～1；e = 500～2000

(iv) Fitness 1 = $\frac{\sum_1^{32} \mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

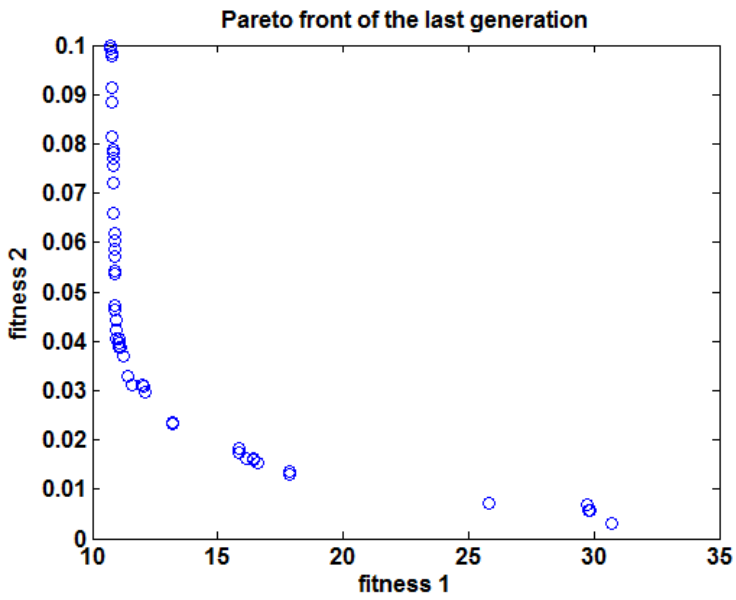

圖 6.12 Pareto front of method 1 with constraint on fitness 2

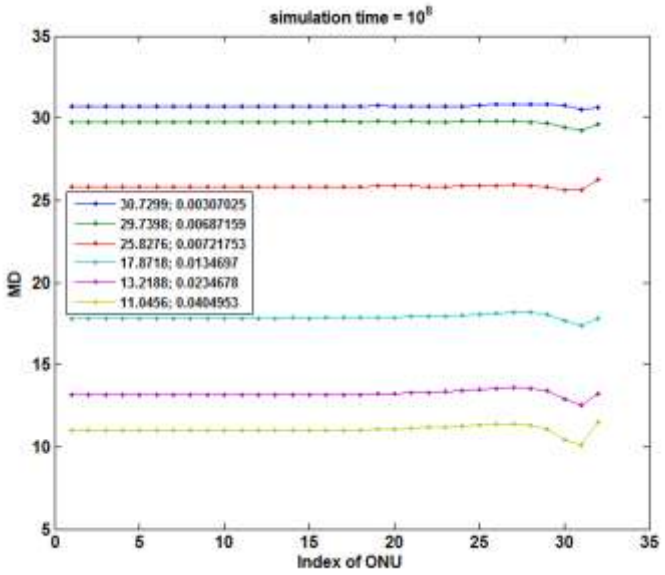

圖 6.13 Some individuals of method 1 with constraint on fitness 2

圖 6.12 與圖 6.10 比較，顯示加入 constraint 的效果，MOGA 最後挑選的 individual 之 fitness2 皆小於 0.1，並且 MD 也比未加入 constraint 前更低，如圖 6.13。

- Method 2：固定 PQS = 500，exponential PR

(i) Load = 0.9；B = 8.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = 500$

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

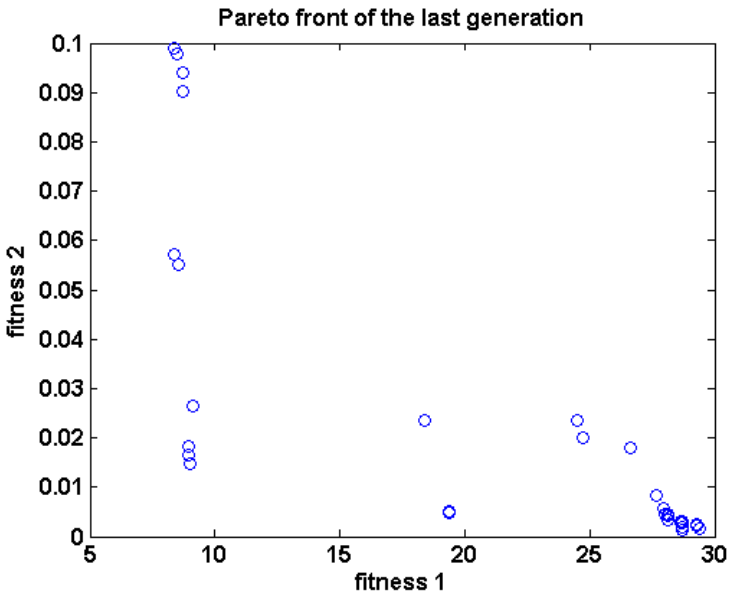

圖 6.14 Pareto front of method 2 with constraint on fitness 2

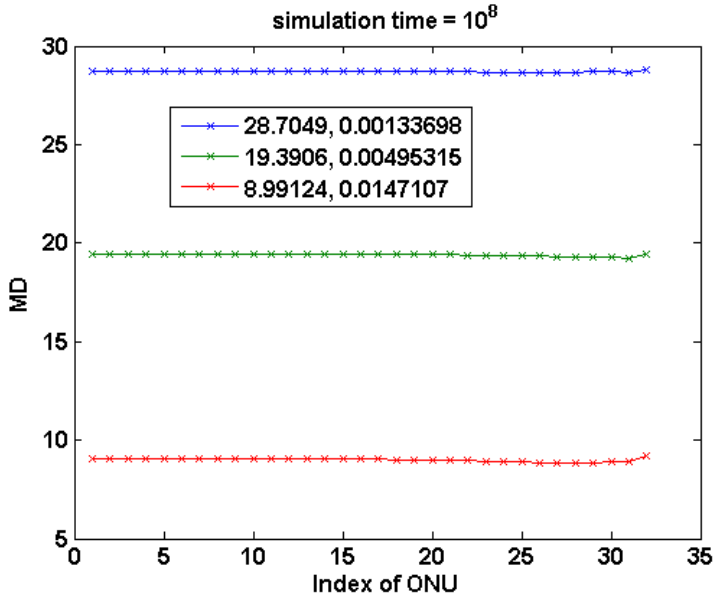

圖 6.15 Some individuals of method 2 with constraint on fitness 2

Case 2 採用 exponential 型態的 PR，其做法與 6.4 節相同，將手動調整的幾條 PR 曲線，利用 Matlab 的 curve fitting tool 做 fit，並將其參數範圍擴大做為 MOGA 的 input。至於將 PQS 固定為 500，則是考慮圖 6.4 中，雖然 PQS 不同，但不同的 PR 曲線仍然可以產生相同的 MD＝10，因此將 PQS 固定，同時可減少一個 MOGA 之 input variable。

將圖 6.14 與圖 6.12 比較，case 2 中最低的 MD 仍然在 30 附近，但其高 fairness

的 individual 數較多，除此之外，圖 6.15 顯示 method 2 在 MD 相當低的地方也找到了不錯的結果。

- Method 3：每個 ONU 之 PQS = 500~2000 且相等，exponential PR

(i) Load = 0.9；B = 8.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

PQS_i = 500～2000

PR = 16.0 for all ONU

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

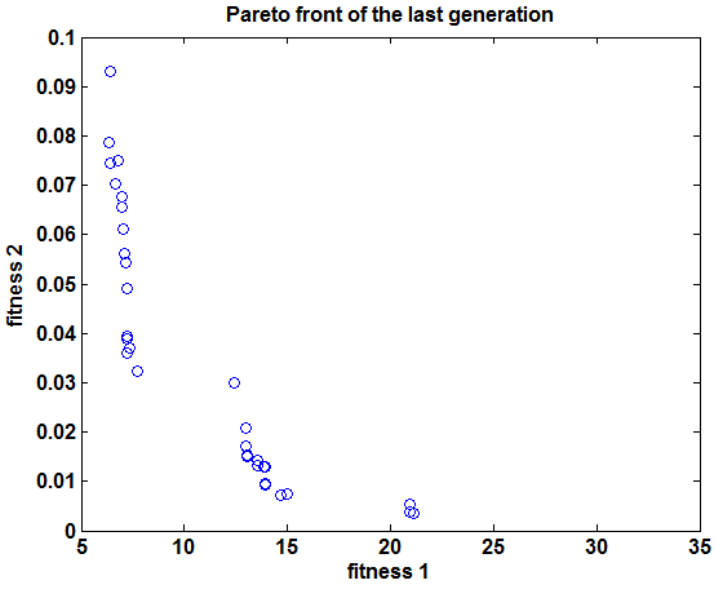

圖 6.16 Pareto front of method 3 with constraint on fitness 2

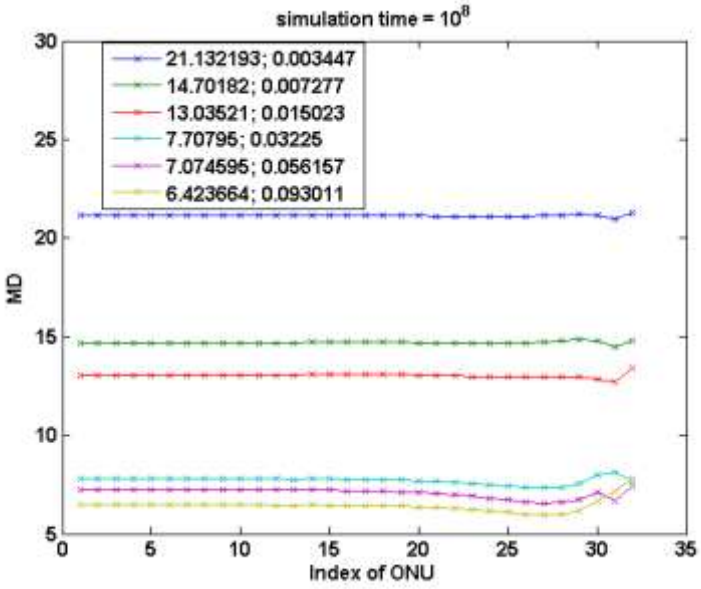

圖 6.17 Some individuals of method 3 with constraint on fitness 2

Method 3 中，我們考慮剩下的可能性，在 PR 為 exponential 型態下，使各 ONU 之 PQS 可以變動，我們預期這個方法可以找到更多好的 individual，但為了避免 MOGA 的 input 過多，令所有 ONU 之 PQS 為相等。由圖 6.16~圖 6.17 發現這種方法找到的 MD 較低，且 fairness 的結果不亞於其他方法，因此，在接下來的討論中，我們會繼續採用 method 3。

6.6 提高 Burstiness

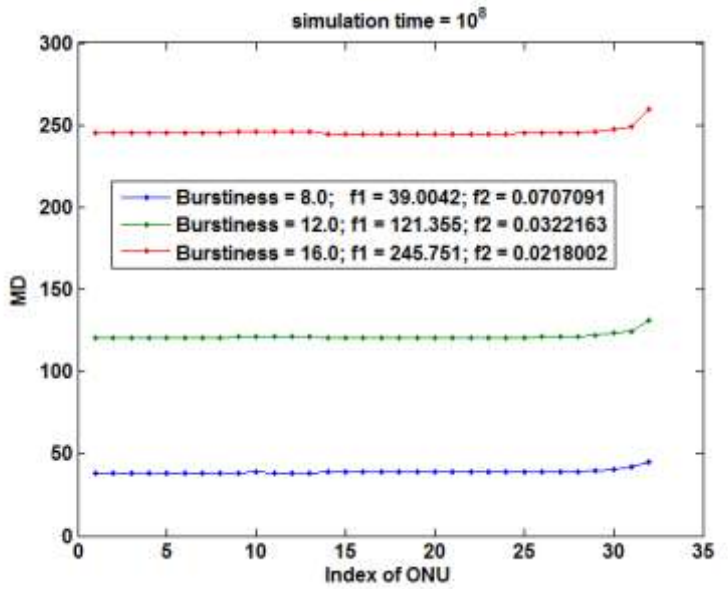

圖 6.18 未使用 MOGA，Burstiness 對 MD 之影響

在 6.5 節中，我們對 PQS 與 PR 使用三種不同的組合做測試，最後選擇的第 3 個 case ，基本上已經可以應付一般的網路流量，在這個小節的討論，我們將 Burstiness 提高 1.5 倍至 2 倍，圖 6.18 是不採用 MOGA 並將 burstiness 提高的結果，可知雖然 load 不變，但 B 提高，使封包來的更集中，對於各 ONU 之 queue 壓力也相對增大，因此 MD 有顯著的上升，這個小節的目的即是要了解我們方法是否能應付這種變化。

- Case 1：B＝12.0

(i) Load = 0.9；B = 12.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

PQS_i = 500～2000

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}}{22+55}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

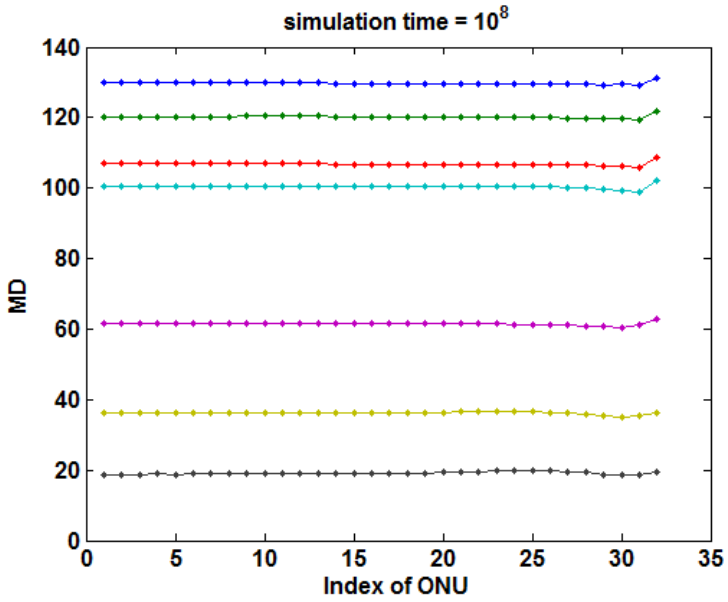

圖 6.19 Burstiness＝12.0 using method 3

- Case 2：B＝16.0

(i) Load = 0.9；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

PQS_i = 500～2000

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}}{22+55}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

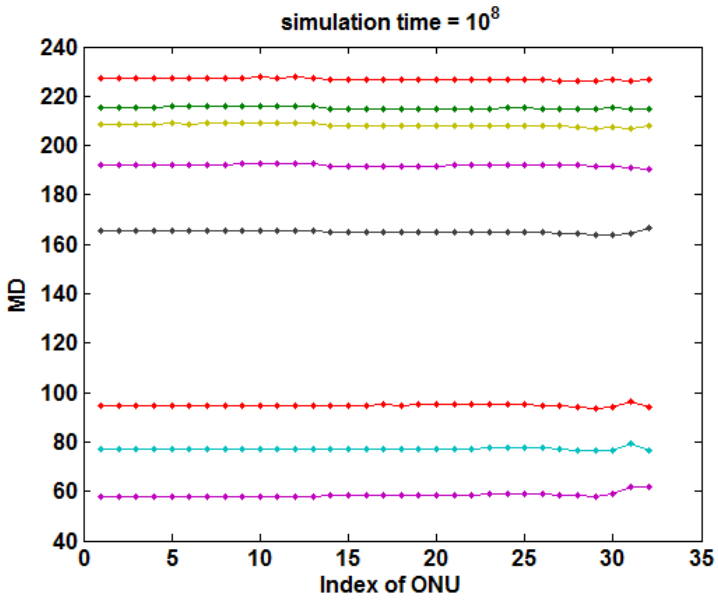

圖 6.20 Burstiness＝16.0 using method 3

圖 6.19 和圖 6.20 為 burstiness 提高後，MOGA 最後一個 generation 挑選幾個代表性的 individual，分別與圖 6.18 比較，我們依然可以找到比標準更低的且更平的 MD 曲線，代表在 burstiness 提高的情況下，我們提出的方法不會受到太大的影響，雖然整體 MD 提高，但仍然可以找到 fairness 不錯的結果。

6.7 提高 Load＝0.95

前一節我們提高 burstiness，並說明我們的方法可以應付這種變化，接下來，我們固定 burstiness 為 16.0 且直接將 load 提高至 0.95，藉由將網路流量加大，繼續測試我們的方法是否可行。需要說明的是，在此我們先採用 6.5 節的 method 2，也就是 PR 為 exponential 函數，PQS 固定為 500，做為比較以及再次確認這個方法的問題。

- Method 1：固定 PQS＝500，exponential PR

(i) Load = 0.95；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = 500$

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ ＝ 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.1

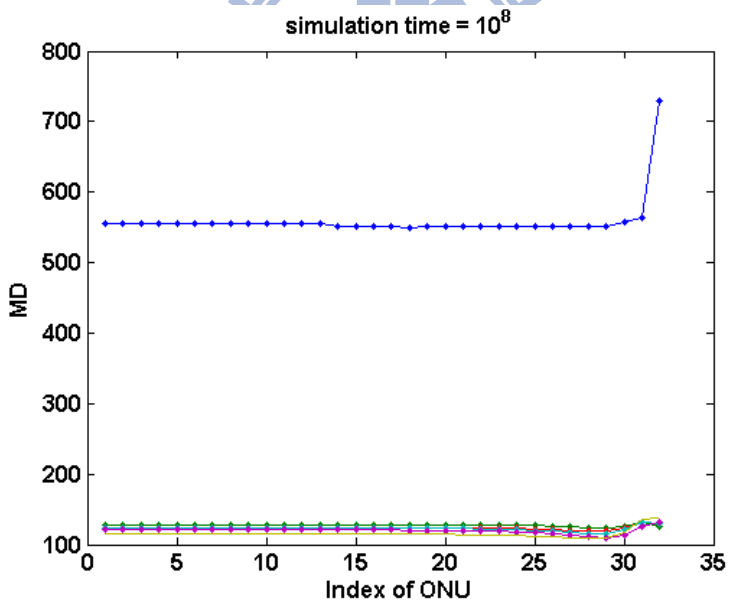

圖 6.21 Some individuals of method 1 with load＝0.95

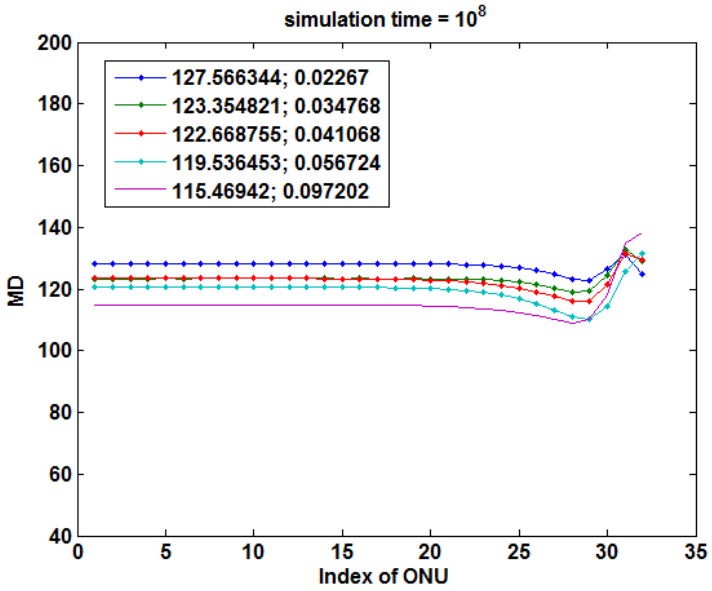

圖 6.22 Some individuals of method 1 with load＝0.95

圖 6.21 中 MD 最高的曲線為未使用 MOGA 之結果，表示 load 由 0.9 提高為 0.95 之後，MD 大約增為兩倍，且後端 ONU 之 MD 與前端差距甚大，下面幾條曲線則是經過 MOGA 後挑選的幾個結果，在圖 6.22 中放大來看，結果不盡理想。在此我們考慮一個可能性：對 fitness 2 之 constraint 設定過嚴，導致在 load 提高的狀況下，個體之多樣性被壓低，因此我們使用 method 2 做測試。

- Method 2：固定 PQS＝500，exponential PR，放寬 constraint

(i) Load = 0.95；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = 500$

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32} \mu_i}{32}$ ＝ 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_{1}^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.2

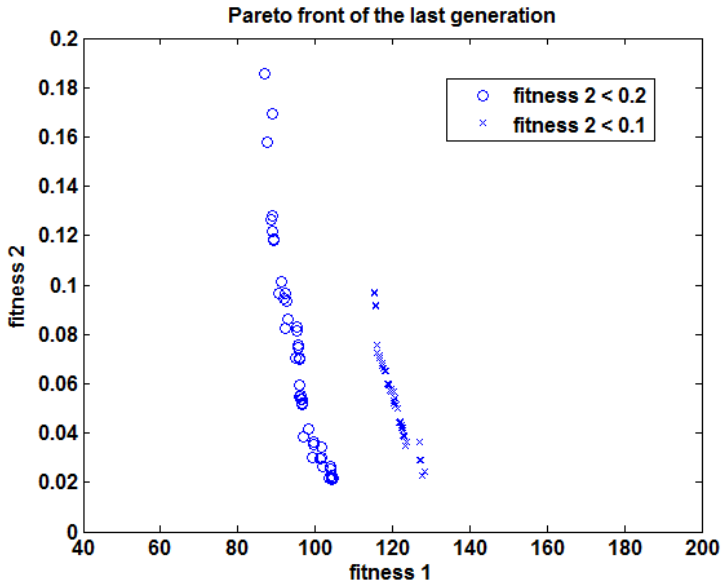

圖 6.23 Pareto front of method 2 with load＝0.95

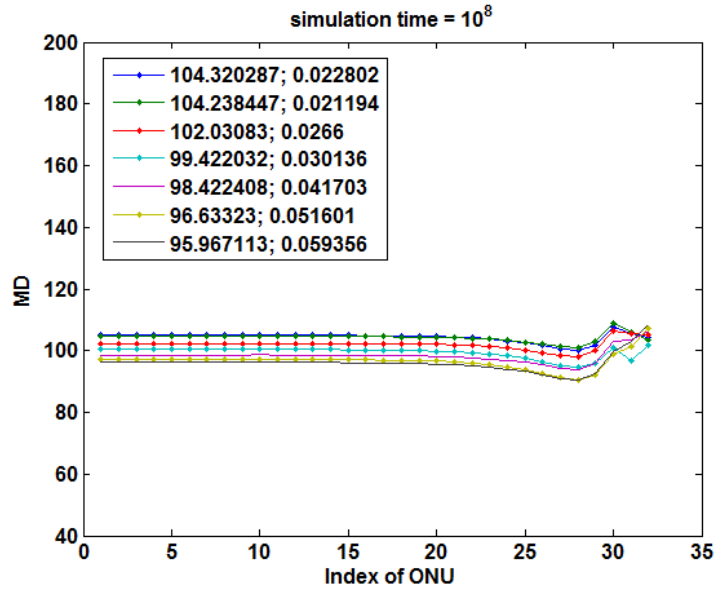

圖 6.24 Some individuals of method 2 with load＝0.95

由圖 6.23～圖 6.24 可以發現，個體之分布在 fitness 1 的部分稍微降低，但 fitness 2 之底限仍然在 0.2 附近，也就是說，MOGA 找到的 individual MD 下降但 fairness 並未改善，因此我們排除是 constraint 造成 MOGA 效果不佳。

- Method 3：每個 ONU 之 PQS = 300～1500 且相等，exponential PR

(i) Load = 0.95；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

PQS_i = 300～1500

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}}{22+55}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.2

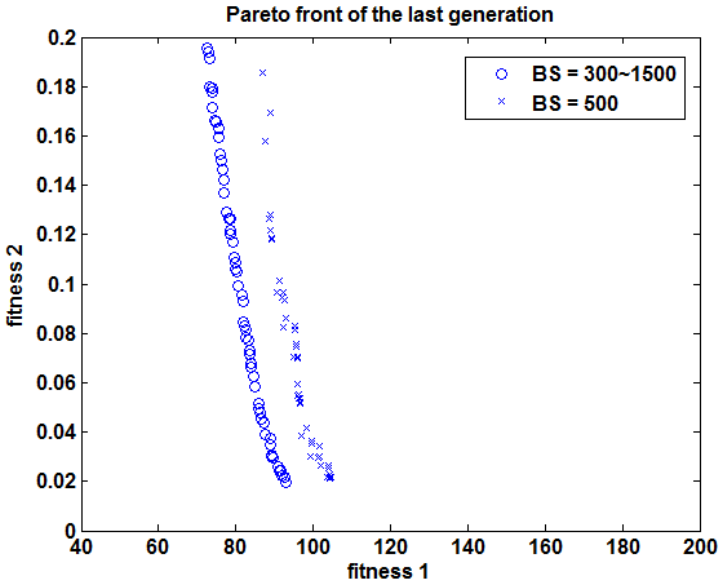

圖 6.25 Pareto front of method 3 with load＝0.95

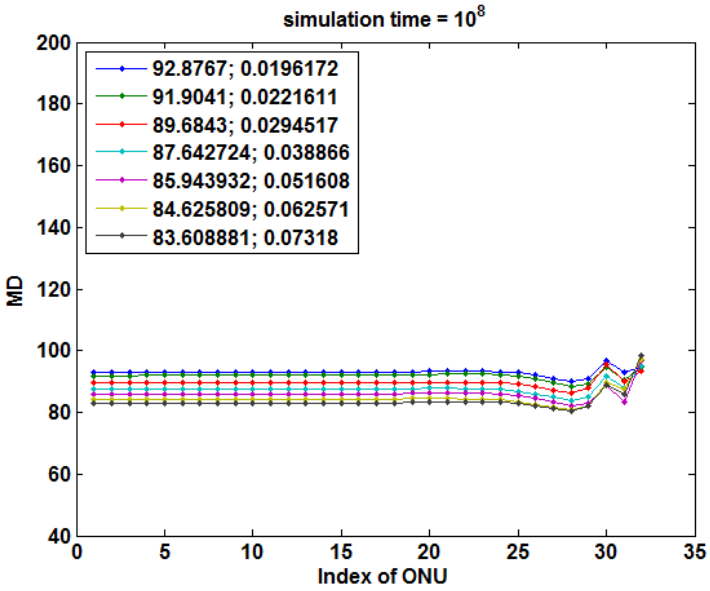

圖 6.26 Some individuals of method 3 with load＝0.95

前一個部份我們確定了 constraint 對 MOGA 造成的影響，但結果仍然不理想，這個部份我們改採用 6.5 節中的 method 3，也就是 PR 使用 exponential 函數，且每個 ONU 之 PQS 相等但可變動。圖 6.25～圖 6.26 顯示使用該方法後，整體 MD 下降但 fairness 未得到改善，因此仍不可行。

- Method 4：exponential PQS，exponential PR

(i) Load = 0.95；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～20；b = -20～0；c = 0～5；d = -1～1；e = 300～1500

PR = exp(a*i + b) + exp(c*i + d) + e

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32} \mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_{1}^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.2

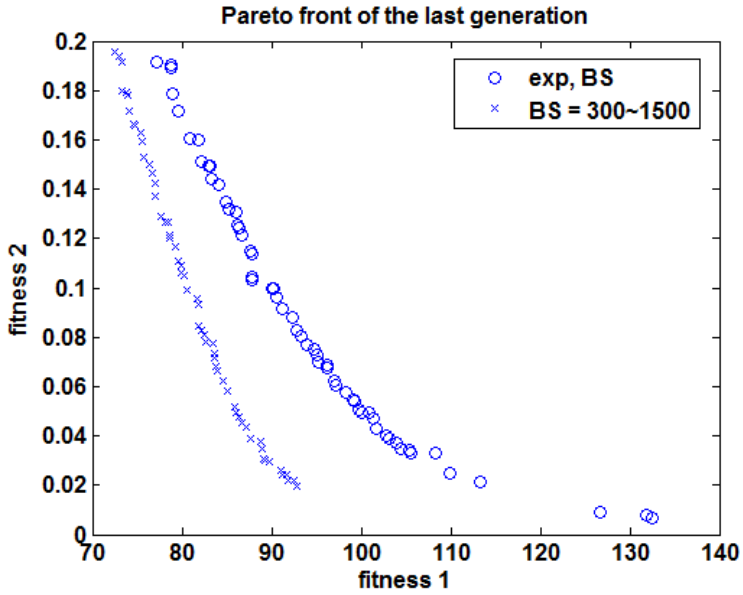

圖 6.27 Pareto front of method 4 with load＝0.95

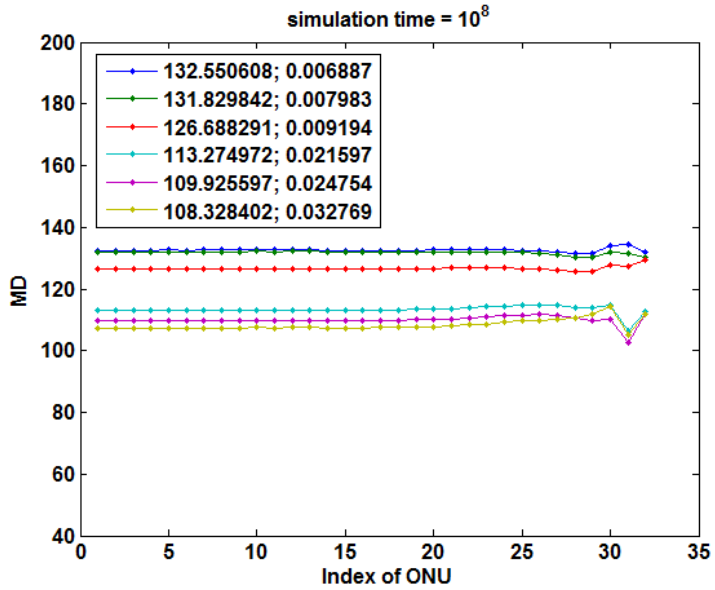

圖 6.28 Some individuals of method 3 with load＝0.95

在最後這個部份，考慮前面幾個方法皆不可行，因此我們決定放棄對 MOGA input 數的要求，將原本所有 ONU 相等之 PQS 也改為 exponential 函數，希望藉由 MOGA 來找到更多優良個體，結果如圖 6.27～圖 6.28，與 method 3 比較可以

發現，MOGA 有能力找到更多優良個體，雖然 MD 提高，但 fairness 有很大的改善，圖 6.28 上面幾條 MD 曲線大致可以符合我們的需求，且與未使用 MOGA 之前的 MD 曲線相比，證明我們提出之方法對於 DBA 在 throughput 與 fairness 上都有極大的幫助。

6.8 提高 Load＝0.98

最後這個階段，我們將 load 再次提高至 0.98，並延續 6.8 節使用的設定。

- Method 1：exponential PQS，exponential PR

(i) Load = 0.98；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～20；b = -20～0；c = 0～5；d = -1～1；e = 300～1500

$PR = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～25

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.2

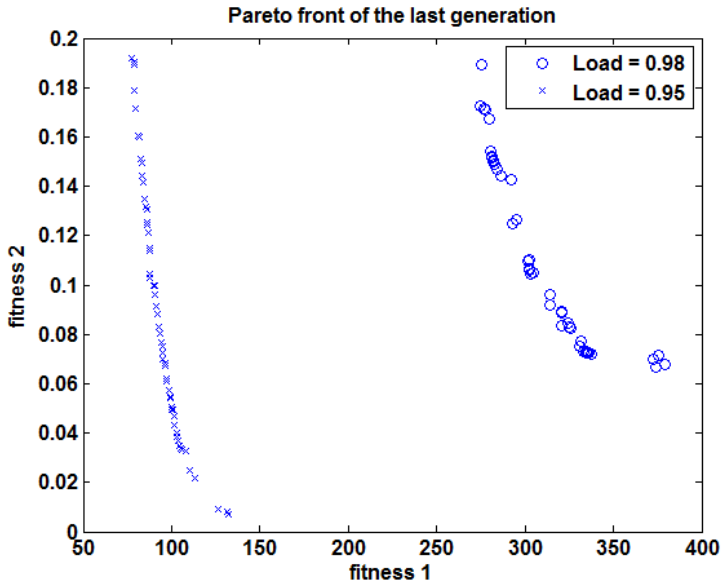

圖 6.29 Pareto front of method 1 with load＝0.98

圖 6.29 為 MOGA 最後一個 generation 的結果，同時與 load＝0.95 時比較，可以發現 fitness 1 與 fitness 2 皆上升不少，並且可以預期挑選出的 individual 並不會太理想。

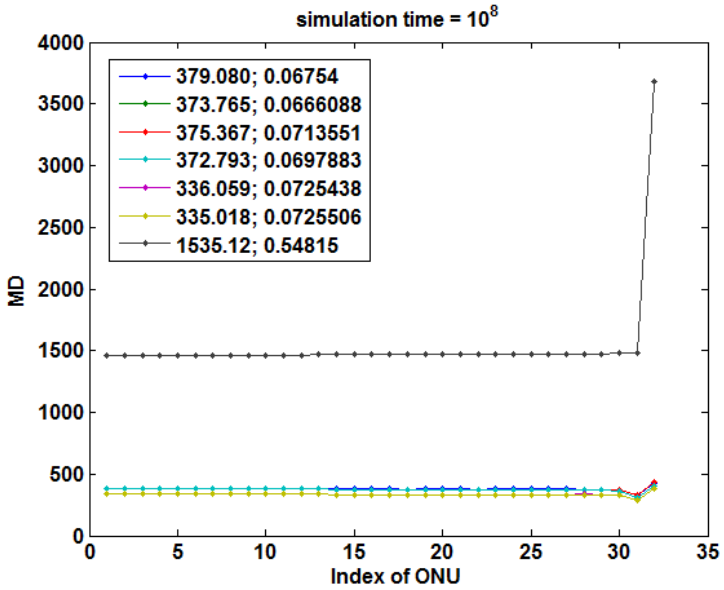

圖 6.30 Some individuals of method 1 with load＝0.98

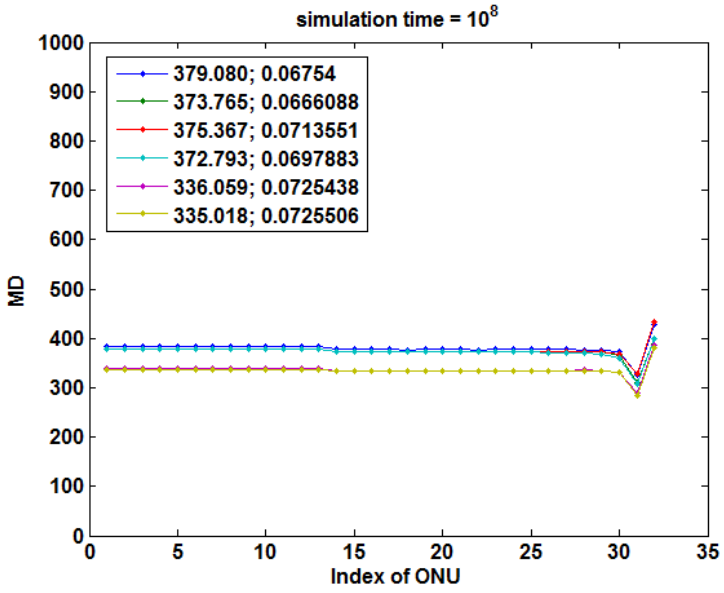

圖 6.31 Some individuals of method 1 with load＝0.98

圖 6.30～圖 6.31 為 MOGA 結果中最後挑選的幾個 individual，圖 6.30 中 MD 最高的曲線為未經過 MOGA 產生之結果做為比較，可以發現 MOGA 產生之結果，雖然 MD 大約只是原本的三分之一，但圖 6.31 顯示曲線並不平整，也就是說，fairness 還不足夠符合我們的要求。

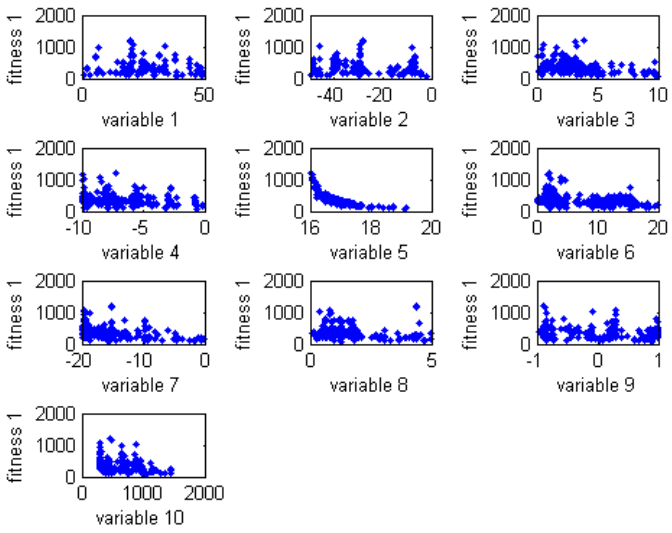

圖 6.32 Exponential variables of PR versus fitness

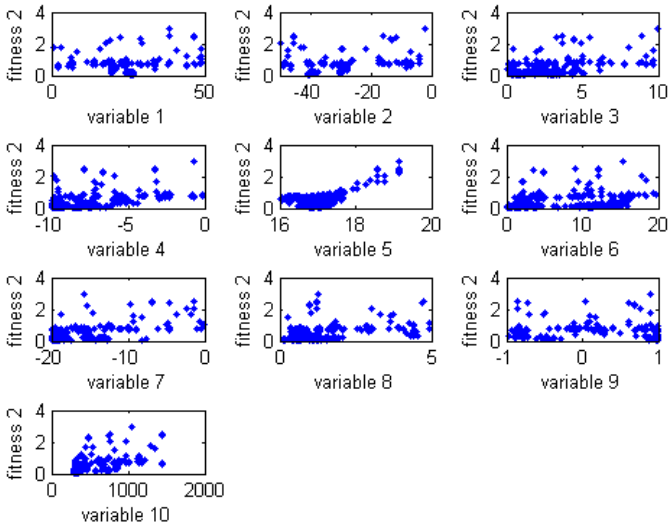

圖 6.33 Exponential variables of PQS versus fitness

因此，在這邊我們再深入分析 exponential 函數與 fitness 的關係，並試圖找出可以再做改善的地方，圖 6.32～圖 6.33 分別是 PR 與 PQS 之 exponential 函數中變數 a～e 與 fitness 的關係圖，做法是將 MOGA 中每個 generation 產生的所有 individual 對 fitness 做分布圖；仔細觀察 PR 之參數 e（variable 5）與 fitness 1 大致成反比且與 fitness 2 大致成正比，因此我們考慮如果將 PR 之參數 e 的範圍做些微的調整，或許可以將 individual 集中在 MD 稍高但 fairness 相對低的區塊內。

- Method 2：exponential PQS，exponential PR

(i) Load = 0.98；B = 16.0

(ii) Generation = 30；Individual = 60

(iii) Input variable:

$PQS_i = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～20；b = -20～0；c = 0～5；d = -1～1；e = 300～1500

$PR = \exp(a*i + b) + \exp(c*i + d) + e$

a = 0～50；b = -50～0；c = 0～10；d = -10～0；e = 16～18

(iv) Fitness 1 = $\frac{\sum_1^{32}\mu_i}{32}$ = 所有 ONU 之 MD 的平均

Fitness 2 = $\frac{\sqrt{\frac{\left(\sum_1^{22}(\mu_i-\overline{\mu_{10}})^2+\sum_{23}^{32}(i-22)(\mu_i-\overline{\mu_{10}})^2\right)}{22+55}}}{\overline{\mu_{10}}}$

(v) Constraint：Fitness 2＜0.2

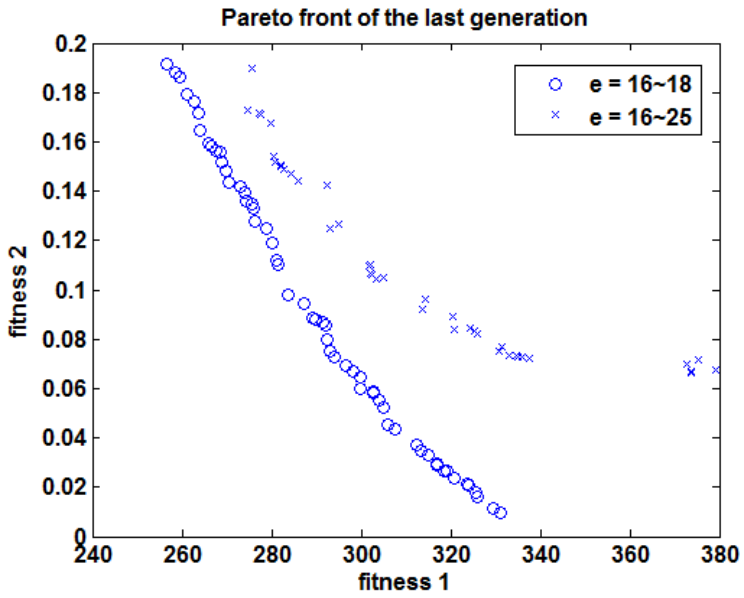

圖 6.34 Pareto front of method 2 with load＝0.98

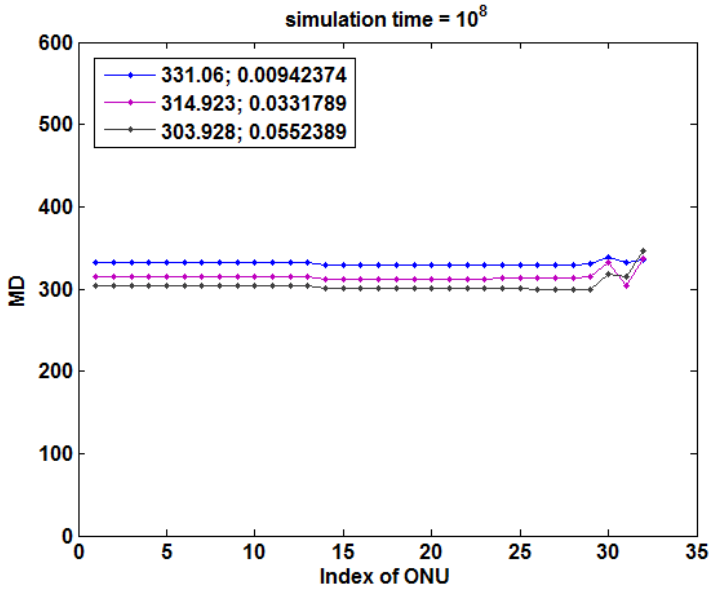

圖 6.35 Some individuals of method 2 with load＝0.98

在這個部分，我們實施了 method 1 最後提到的，將 PR exponential 函數的參數 e 稍微做了調整，由原本的 16～25 修正為 16～18，希望盡可能將結果集中在我們預期的區塊，由圖 6.34 可以看到這樣修正帶來的好處，MD 與 deviation 同時降低，圖 6.35 中也顯示其產生的個體符合我們的需求，證實控制變數的做法確實可以更加增進我們的方法。

6.9 Future Work

在前面的討論中，我們發現對 fitness 2 之 constraint 若太小會使 MOGA 不易找到合適的 individual，反之則無法快速在最佳解附近收斂。而我們到目前為止的實驗，都將模擬時間設定在 10^8，在此種情況下，產生的 mean delay 值為穩態，也就是說，將模擬時間再拉長，mean delay 不會有太大的變化；但實際網路流量的狀態會隨著時間改變，load、burstiness 皆是。

因此，我們接下來將嘗試在模擬時間為 10^8 時，在 MOGA 不同的 generation，讓 load 也隨著改變，並且使用自動調整 constraint 的機制，藉此測試 MOGA 在 online 時，是否能夠應付不同的 traffic。另外我們也將嘗試將觀察時間縮短，若 MOGA 可以在模擬時間 10^8 以下就找到適合網路流量的調整機制，時間越短代表我們的方法對 traffic 能有更即時的幫助。

此外，對於 PQS 與 PR 之 exponential 函數的控制，目前我們皆是以人力觀察控制，我們希望接下來的 work，可以找出更多參數與 traffic 的關係，再利用較 intelligent 的方法，例如 fuzzy，給予 MOGA 參數的範圍，希望能將整個動態頻寬調整機制改進到完全自動化且有更即時的反應。

7 結論

在此論文我們提出一個最佳化、快速的動態頻寬分配設計，用於下世代分頻多工存取被動光網路(Next-Generation OFDMA-PON)上，此架構具有高頻寬、低成本、節能的優點，同時可整合無線網路訊號的傳輸，我們以多目標基因演算法達到最佳化，並使用平行化進行快速運算，能即時監控網路流量並調整頻寬分配，讓在該架構上的使用者獲得高 throughput 與高 fairness 的服務品質保證。我們將 PQ－PQS 頻寬分配機制轉換為較易處理與理解之最佳化問題，利用多目標基因演算法，仿效大自然適者生存的法則以取得最佳解，而訊息傳遞介面則用來加速處理最佳化問題的運算速度。我們結合以上幾種方法，並且做了諸多改善，包含對於 MOGA input 函數的 exponential 近似、fitness function 之 constraint 以及對其近似函數之參數做控制與調整，最後產生的結果符合我們對提出之方法的期望，期能提供在此新穎的下世代光網路架構的使用者，一套完整且優良的動態頻寬分配與服務品質方案。

8 Reference

[1] D. Qian, J. Hu, P. Ji, L. Xu, T. Wang, M. Cvijetic, T. Kusano, “Experimental Demonstration of a Novel OFDM-A Based 10Gb/s PON Architecture,” ECOC 2007.

[2] Yu-Min Lin, “Demonstration and Design of High Spectral Efficiency 4Gb/s OFDM System in Passive Optical Networks,” Optical Fiber Communication and the National Fiber Optic Engineers Conference 2007, 25-29 March 2007, pp. 1-3.

[3] A.J. Lowery, L. Du and J. Armstrong, “Orthogonal frequency division multiplexing for adaptive dispersion compensation in long haul WDM systems,” in Proc. OFC 2006, PDP39.

[4] Jianjun Yu, Ming-Fang Huang, Dayou Qian, Lin Chen, Gee-Kung Chang, “Centralized Lightwave WDM-PON Employing 16-QAM Intensity Modulated OFDM Downstream and OOK Modulated Upstream Signals,” Photonics Technology Letters IEEE, Sept. 15, 2008, Vol. 20, Issue 18, pp.1545-1547.

[5] C. W. Chow, C. H. Yeh, Y. T. Li, C. H. Wang, F. Y. Shih, Y. M. Lin, C. L. Pan, and S. Chi, “Demonstration of High Spectral Efficient Long Reach Passive Optical Networks Using OFDM-QAM,” in Conference on Lasers and Electro-Optics/Quantum Electronics and Laser Science Conference and Photonic Applications Systems Technologies, OSA Technical Digest (CD) (Optical Society of America, 2008), paper CPDB7.

[6] R. A. Grifn, P. M. Lane, and J. J. O'Reilly, “Radio-Over-Fiber Distribution Using an Optical Millimeter-Wave/DWDM Overlay, ” Proc. OFC/IOOC 99, vol. 2, pp. 70.

[7] Chae-Sub Lee, D. Knight, “Realization of the next-generation network,” IEEE Commun. Mag., Oct. 2005, Vol. 43, Issue 10, pp. 34–41.

[8] Gangxiang Shen, Tucker R.S., Chang-Joon Chae, “Fixed Mobile Convergence Architectures for Broadband Access: Integration of EPON and WiMAX,” IEEE Communications Magazine, Aug. 2007, Vol. 45, Issue 8, pp.44-50.

[9] Junqiang Hu, Dayou Qian, Ting Wang, “Wireless Intermediate Frequency Signal over Passive Optical Networks: Architecture and Experimental Performance Evaluation,” OFC/NFOEC 2008.

[10] Qian Dayou, Hu Junqiang, Ji Philip Nan, Wang Ting, Cvijetic Milorad, “10-Gb/s OFDMA-PON for Delivery of Heterogeneous Services,” Optical Fiber communication/National Fiber Optic Engineers Conference 2008, 24-28 Feb. 2008, pp.1-3.

[11] S.-1. Choi and J,-D. Huh, “Dynamic Bandwidth Allocation Algorithm for Multimedia Services over Ethernet PONs,” ETRI J., vol. 24, no. 6, Dec. 2002, pp. 465-68.

[12] G. Kramer, B. Mukherjee, and G. Perwento, “IPACT: A Dynamic Protocol for an Ethernet PON (EPON),” IEEF Commun. Mag., vol. 40. no. 2, Feb. 2002, pp. 74-80.

[13] H.-J. Byun, I.-M. Nho, and J.-T. Lim, “Dynamic Bandwidth Allocation Alaorithm in Ethernet Parrive Ootical Networks,” Elect. Lett.. vol. 39. no. 13, June 2003. pp. 1001-02.

[14] Hassan Naser, Hussein T. Mouftah, “A joint-ONU interval-based dynamic scheduling algorithm for ethernet passive optical networks,” IEEE/ACM Transactions on Networking, vol. 14, Issue 4 , Aug. 2006, pp 889-899.

[15] Ghani N., Shami A., Assi C., Raja M.Y.A., “Intra-ONU bandwidth scheduling

in Ethernet passive optical networks," IEEE Communications Letters, vol. 8, Issue 11, Nov. 2004, pp 683-685.

[16] Biao Chen, Jiajia Chen and Sailing He, "Efficient and fine scheduling Algorithm for Bandwidth Allocation in Ethernet Passive Optical Network" IEEE Journal of selected topics in Quantum electronics July/August 2006.

[17] Holland, John H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press. ISBN 0262581116.

[18] Steuer, R.E. (1986). Multiple Criteria Optimization: Theory, Computations, and Application. New York: John Wiley & Sons, Inc. ISBN 047188846X.

[19] ^ Sawaragi, Y.; Nakayama, H. and Tanino, T. (1985). Theory of Multiobjective Optimization (vol. 176 of Mathematics in Science and Engineering). Orlando, FL: Academic Press Inc. ISBN 0126203709.

[20] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. 2000, A Fast and Elitist Multi-Objective Genetic Algorithm-NSGA-II.

[21] Deb, K. Multi-Objective Optimization using Evolutionary Algorithms John Wiley & Sons, 2001.

[22] MPI-2: Extensions to the Message-Passing Interface: http://www.mcs.anl.gov/research/projects/mpi/mpi-standard/mpi-report-2.0/mpi2-report.htm

[23] Message Passing Interface Forum: http://www.mpi-forum.org/

[24] MPICH2 homepage: http://www.mcs.anl.gov/research/projects/mpich2/

[25] Deb and R. B. Agrawal. Simulated binary crossover for continuous search space. Complex Systems, 9(2):115--148, 1995.