利用軟體定義網路在資料中心的路線權重計算負載平衡演算法

鄭明華[1], 王建明[2], 黃文祥[3,*], 吳妍靚[4], 林政翰[5]
國立高雄應用科技大學[1,2,3],慈惠醫護管理專科學校[1],實踐大學高雄校區[4],輔英科技大學[5]
1100404107@cc.kuas.edu.tw, xeriod049136@hotmail.com, wshwang@cc.kuas.edu.tw*
yanjing@mail.kh.usc.edu.tw, ft065@fy.edu.tw

摘要

近年來隨著資料中心(Data Center)的相關領域興起，與其息息相關的雲端運算(Cloud Computing)技術發展議題隨之熱門了起來。對於資料中心來說，如何有效率的處理其商業和運作組織的資料是一項重要的指標，對應於此項需求衍伸出的即是相關伺服器網路環境優化問題。軟體定義網路(Software-defined networking)也是近年來炙手可熱的技術之一，此架構能在不更動硬體設備的前提之下，配合網路管理者需求進行動態的網路規畫，為優化網路環境提供了新的方法，也提供了核心網路及應用創新的良好平台。由於在稍具規模的資料中心中，雲端伺服器的數量往往十分可觀。此種情形會造成整體網路環境的相對複雜化，倘若沒有一個良好的路由演算法來規劃資料轉送情形，則容易造成整體網路的壅塞(Network congestion)且無法達到負載平衡(Load balancing)的目的，從而造成整體網路效能的損失。為了解決這樣的問題，本論文提出一個針對資料中心網路進行負載平衡的演算法，利用 SDN 架構的特性，配合路線權重計算路由演算法進行負載平衡的路由動作，藉此優化整體網路環境的表現。

關鍵詞：資料中心、雲端運算、軟體定義網路、動態路由演算法

Abstract

Nowadays, with the rise of the area about Data Center (DC), the issue of Cloud Computing is growed up. How to deal business and operational data effectively for Data Center is an important thing. Thus, the network performance issue is accompanied. Software-defined networking (SDN) is the hot technology, too. This architecture allows dynamic network planning to meet the needs of network administrators without the need for hardware devices, providing new ways to optimize the network environment and providing a good idea for core network and application platform. Because the great number of normal Data Center server, is easy to make network environment more complexly. If it is lack a suitable routing algorithm to planned routing path, it may cause network congestion and load balancing issues, moreover make the performance of network get poor. In this paper, we proposed a load balancing algorithm based on SDN, with Path Weight Computing to improve load balancing issue and get more performance.

Keywords: Data Center, Cloud Computing, SDN, Dynamic Routing Algorithm

1. 緒論

1.1 研究背景與目的

隨著近年來雲端領域相關技術的逐漸成熟，使得商用及相關運作組織逐漸紛紛計畫架設屬於內部的資料中心，如 Google 企業的 Google Modular Data Center、昇陽電腦的 Sun Modular Datacenter、IBM 的 Portable Modular Data Center……等，越來越多的企業公司規畫著自己的資料中心[1]。傳統的資料中心網路(Data Center Netwrok)[2]架構可分為三層，分別為：接取層(Access Layer)、Aggregate 聚合層(Aggregate Layer)和核心層(Core Layer)，如圖1所示，接取層的組成為網路邊緣(Edge layer)的網路交換器。聚合層的網路交換器則負責將接取層的網路交換器互相連接在一起。所有聚合層網路交換器通過核心層網路交換器相互連接。此外，核心層的交換機還負責將資料中心連接到互聯網。然而，三層式架構的資料中心型態無法應對日益增長的雲端計算需求，擴展性是這個架構的主要問題之一，並且在容錯能力、能源效能等相關問題上也存在部分問題導致本架構無法有效的因應現下的資料中心需求。

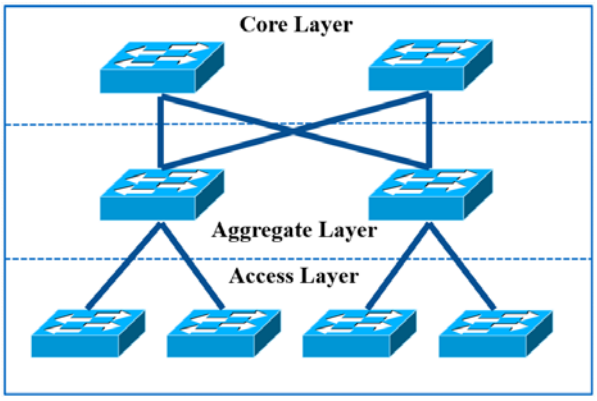

圖 1 傳統三層式架構資料中心網路

因為上述種種傳統架構存在的問題，胖樹[3]DCN 架構(Fat tree DCN architecture)即被提出來解決傳統式三層 DCN 架構。胖樹 DCN 架構基於傳統式架構的分層定義之下再進行擴充，此架構下的網路交換器透過採用 Clos 拓樸[4]進行基本部屬，因此其網路交換器數量遠遠大於三層式 DCN 架構，並且擴充性及管理性都遠優於傳統式架構。再利用特殊規則及定義來有效的擴充網路交換器的數量，並藉此規範進行相關資訊(線路情形、網路交換器分布)的統整，也因此而給予網路管理策略更加靈活的調度。

1.2 研究目的

由於資料中心的特性[2][9]，營運方必須考量相關成本進行安置相關系統及部件的措施，如電信和儲存系統。如何在成本與效能之間取得平衡一直以來都是營運方需要考量的重要關鍵，在如此數量龐大的運算服務器中該如何有效地進行網路環境的佈署更是十分重要的一大問題。結合SDN[5][6][7][8]架構進行資料中心網路的佈署是一項有效率解決此問題的方法，透過 SDN 特性，營運方不需要對現有的硬體設備進行大規模的汰換及更新，就能達到優化現有配置的目的。然而，在如此繁複的網路環境之中，倘若缺少一個有效的路由演算法，很容易造成整體網路的壅塞，使得效能下降，或是無法達到線路負載平衡所造成的部分線路過度利用，其他的網路線路過度閒置的情況，則使用 SDN 所得到的效益也就被侷限在一定的程度了。

另一方面，由於 DCN 使用網路拓樸大多是所謂的多重樹狀網路拓樸(multi-rooted tree network topology)，此種網路拓樸相對於其他拓樸更加的繁複，所使用的路由演算法複雜度也因此相對應的提高，這將會造成運算資源上一定的成本消耗，倘若運算時間過長，也可能因為網路環境動態的變更而產生運算後的最適合路徑與當下網路環境不符情況，而舊有的路由方式又多是採取單路線路由[10](Single Path Routing)的方式進行運算。在可用線路相對較多的多重樹狀網路拓樸中，很容易產生部分線路利用率過高，但其餘可用線路卻處於閒置狀態的情形發生，更有可能在進行多次單路線路由的情況之下選用到重複的路線配置，使最終選出的路徑造成重疊(overlapping)的情況，故在此種網路拓樸之下大多選用的路由方式為多路徑(Multipath routing)[11]的方式，利用多路徑方式進行路由可以有效地降低路徑重疊的現象，並使用多個替代路徑進行路由動作，此種方式擁有增加容錯、帶寬以及安全性等各種優點。

2. 相關研究

SDN 架構與 OpenFlow 協議在近年來被提及的使用層面越來越廣，由於其可程式化的特性，對不同的領域進行配合的程度上擁有較高延展性及可擴充性，透過分離雙平面的特性[12][13]，SDN 可達成網路基礎設施的虛擬化，相對容易建立客製化的功能及服務。在本章節將說明 SDN 的概念與架構，接下來來針對 OpenFlow 協定進行介紹並談論到胖樹拓樸，最後會提及現行三種的路由策略。

2.1 SDN 概念與架構

在網路環境中，資料的傳輸由：客戶端對客戶端、客戶端對伺服器端與客戶端與基礎建設(Infrastraction)端之間的連線所構成，在連線上的封包傳輸是由路由器與交換器來轉送。而這些路由器與交換器通常都是只擁有區域性(Local)資訊，屬於相對「封閉」的系統，僅由網管人員藉由特定的控制介面來操作，而這些硬體設備一但面臨維護及設定上的變更，在不影響使用者正常使用的前提之下是一件很困難的事情。SDN 概念的出現，讓網管人員在不更動硬體設備的前提之下以中央共管的方式重新規畫網路，因此，SDN 架構是一項解決上述問題的有效方法。

SDN 的架構主要分為應用層、控制層及資料層(圖2)，中央控制層主要透過南北向不同類型的ＡＰＩ協定與上下層溝通，並且將資料層與控制層進行分離，SDN 將網路架構簡化為封包轉送與網路決策控制器(decision-making network controller)二個部分，其中封包轉送由 FlowTable 所負責，與 controller 的部分則用 OpenFlow Channel 所負責，如圖3所示。

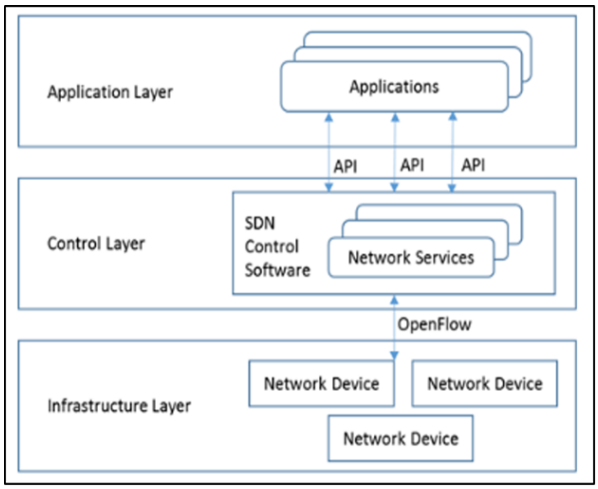

圖 2 SDN 整體架構圖
資料來源：[13]

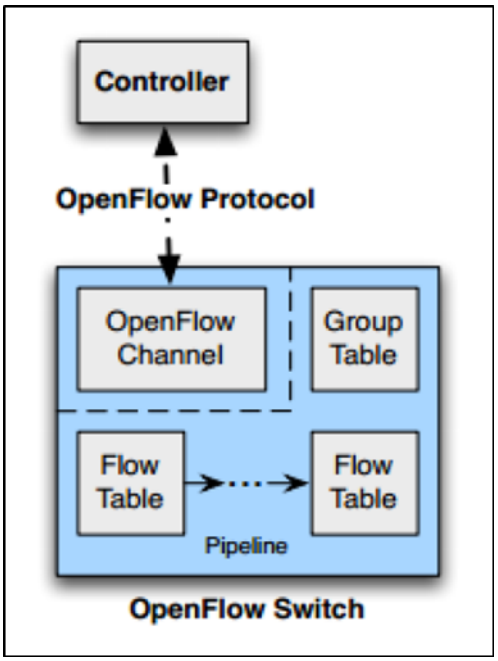

圖 3 OF Switch 分層示意
資料來源：[12]

2.2 OpenFlow 協定

OpenFlow[5][6][7][8]是一種網路通信協定，透過網路交換器或是路由器的轉發層(Forwarding Plan)改變封包所走的路徑，在 OpenFlow 協定裡提到，Flow 的轉發由 Flow Table 裡面的 flow entry 所決定，每張 Flow Table 是由複數條 flow entries 所組成，其中每一條 flow table entry 分別擁有以下欄位：Match Field、Priority、Counters、Instructions、Timeouts、Cookie。

Match Field 的內容包含輸入的端口(Ingress Port)、封包的標頭(Header)以及數個選用的匹配字段。Priority 內主要存放的是該條 entry 的匹配優先度。Counter 內包含數個統計資料，在該條 entry 被匹配成功的時候更新。Instructions 內包含修改動作的指令或是需要處理的流程。Timeouts 內存放的是該條 entry 的最大存活時間或是閒置時間 Cookie 存放的是不可見的 Controller 用數據，被用來過濾統計、修改及刪除。

當一個 packet flow 進入到一個 OpenFlow Switch 後，會在該 Switch 的 Table 0 進行查詢轉送，並到達對應的 flow table n，再根據匹配到的 instructions 欄位執行應用動作同時進行更新，重複此流程直到最終執行所有的 Action Set 將此 flow 轉發出去為止。

2.3 胖樹(Fat Tree)拓樸

Fat Tree DCN 架構基於傳統式架構的分層定義之下再進行擴充，此架構下的網路交換器透過採用 Clos 拓樸[4]進行基本部屬，因此其網路交換器數量遠遠大於三層式 DCN 架構，根據[3]內容的定義，一個 Fat Tree topology 裝置數量需要遵循以下方式佈署——定義 Pod 數量為 k 的一個 DCN 拓樸中每個 Pod 的內容包含了〖(k/2)〗^2個 host 以及在 Aggregation 層與 Edge 層各 k/2個 k-port 的 Switches，每個 Edge 層的 Switch 連接 k/2個 host 與 k/2 個 Aggregation 層 Switches，每個 Aggregation 層的 Switch 連接 k/2個 Edge 層與 Core 層的 Switches，共會有〖(k/2)〗^2數量的 Core 層 Switch 並且連接到 k 個 pods 上，如圖4。

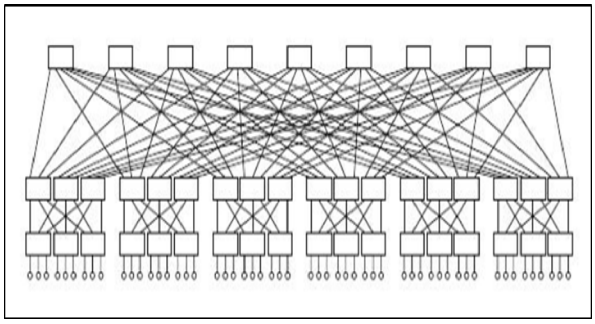

圖 4 K=6的 Fat Tree Topology
資料來源：[3]

2.4 路徑路由

由於針對不同類型規模拓樸的影響，所選用的路由策略也不盡相同，最基本會使用的策略即是單一路徑路由方法[10][14]。然而，在多重樹狀結構拓樸的網路環境中，由於線路狀況相對以往複雜許多，單路徑路由方式無法滿足需求。因此大多使用多重路徑路由策略[11]進行路由，多重路徑路由較單一路徑路由不同的是其路由路徑使用多個替代路徑的路由技術，此種做法可有效的包含增加線路容錯率並分散重載流量。

等價多路徑路由(Equal Cost Multiple Path Routing, ECMP)[15] 承襲多重路徑路由的特點，擁有多個替代路徑同時的進行路由行為，並以隨機分割 flow 的方式進行等價多路徑路由動作。非等價多路徑路由(Non–Equal Cost Multiple Path Routing,Non-ECMP) 在 ECMP 的基礎上再考慮其他權重的路線算法，如 flow 的狀態、線路的利用率，或是其他網路參數。藉由考量更多的環境參數加入權重計算，為解決壅塞及網路利用率不均問題提供一種新的解決辦法。

3. 路線權重計算負載平衡演算法

在2-4所談論的兩種多路徑方法中，ECMP 路由方法較為簡易，但若將所有路線進行同等的優先度區分，不考慮其他因素的話，則在網路運行過程中很有可能會產生多路徑路由卻又重疊的情況發生，如圖5所示，拓樸圖為一個典型的4-8-8-16 Fat Tree 網路拓樸，每條線路的容量大小均為1Gbps，其中有兩條 Flow 分別為 Flow A 與 Flow B。Flow A 的線路流量為500Mbps，Flow B 的線路

流量為300Mbps，經由 ECMP 方式算出來的路徑會在聚合層至核心層之間產生重疊現象，則該重疊線路將會產生800Mbps 的流量，雖然還未達到線路容量上限，但其實若走聚合層第二台網路交換器利用該交換器之線路是可以將流量進行有效的分流而不會造成重疊線路的重載情形，這是因為 ECMP 只考慮到距離而未將路徑上線路的使用狀況考慮進去而發生的情況。

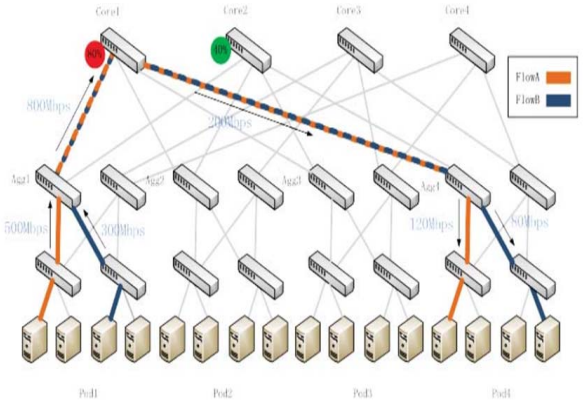

圖 5等價多路徑路由(ECMP)

倘若是採用 Non-ECMP 進行路由，則在同樣情況之下計算出來的路線會變為如圖6所示，將兩條 Flow 考量線路狀況後進行分流，可以很大幅度地減少計算出重複路徑並造成線路重載的情況發生。

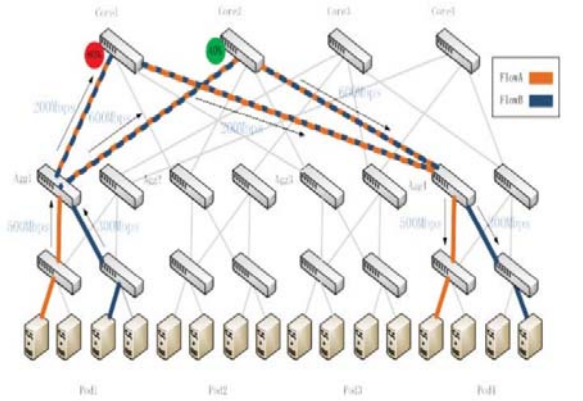

圖 6 非等價多路徑路由(Non-ECMP)
資料來源:[20]

本章將依序說明──3-1節說明路線權重計算負載平衡演算法(Path Weight Computing Algorithm)設計概念、3-2節針對所提的 PWC 演算法以及 PWC 流程圖詳細敘述，最後3-3節使用 PWC 演算法舉一個簡單例子加以說明 PWC 在線路重載狀態時如何運算路徑的過程。

3.1 PWC 演算法設計概念

本論文所設計之路線權重計算負載平衡演算法是以 Non-ECMP 的路由方式為基礎進行改良，在眾多可用路線之中選擇線路負載較低的路線作為傳輸路線，由於結合在 SDN 架構之中，中央控制器(Controller)可以擁有整體網路的所有資訊，再更進一步的計算權重來配合路由方式來制定策略。由於在 Non-ECMP 的路由策略每條路徑的權重值是一項重要的參考數據，因此如何參考整體網路環境狀況計算所有可用路徑的權重值即是整體路由策略的重點。PWC 演算法最開始會考慮所有可用線路的負載程度(線路利用率)，並且由於所謂的瓶頸(bottle neck)及重疊(over lapping)問題，單純考量線路利用率依然無法滿足所有的情況，因此 PWC 演算法除了考量線路利用率以外會再計算每條路徑的利用率之後進行該路徑的標準差(Standard Deviation)的計算，由於標準差的大小可以判斷該路徑所有內含線路的利用率離散程度，因此可以在相同平均值的情況下協助選擇路徑的判別，並更進一步利用標準差值進行 T 檢定(T-test)的動作，藉由此動作來判斷該路徑與平均值的差異顯著性。

3.2 PWC 演算法

本研究提出之機制設計於網路環境運作，期間週期性的令 SDN controller 藉由定期由 OpenFlow Switch 回傳的整體網路線路負載情形進行統計分析，根據各線路利用率來計算所有可用路徑的利用率平均值，再計算出所有的路徑標準差值做為第二個判斷最佳路徑的條件，並在最佳路徑擁有複數解的情況之下則將所有的最佳路徑進行 T 檢定，篩選出最符合負載平衡目的的路徑。

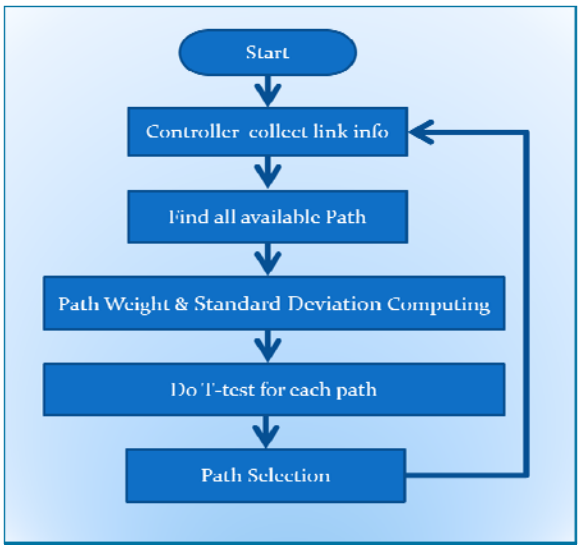

圖 7 PWC 演算法流程圖

3.3 PWC 演算法範例

本章節將舉例 PWC 演算法在特殊狀況例如：

由三條路線所組成的兩條路徑：

路徑 A 利用率平均值為0.5，其路線利用率分別為0.4、0.5、0.5、0.6。

路徑 B 利用率平均值為0.4，其路線利用率分別為0.1、0.2、0.8、0.9。

依照路由規則路徑 B 的優先度會高於路徑 A，然而路徑 B 的第三、四條路線當下利用率已經高於0.8，對於該路徑來說線路處於重載狀況，不適合再給予該條路徑給予負載，即產生瓶頸問題。另一方面，路徑 A 與路徑 B 的路徑平均值均為0.5，其路線利用率也在正常範圍之內，則此時原機制亦無法判斷出哪條路徑為優先度最高的路徑，因此透過標準差(σ)的大小來決定最終路徑。

4. 模擬實驗與效能分析

為了驗證本論文提出的 PWC 演算法確實能夠改善在資料中心上的負載平衡問題，我們將使用 Mininet[16]來創建一個虛擬的網路拓樸平台藉此創造出虛擬的 host 並使其如同真實環境一般的發送封包。在 Mininet 的平台之下建立相對應的 Fat Tree 拓樸並透過 host 的動作來模擬資料中心的運作情形。

4.1 PWC 演算法範例

在 Mininet 的設定上除了提供基本網路相關的協定外，因為使用 ubuntu 系統的關係，可以透過不同的擴充插件進行流量發送模擬、監控及追蹤等功能，並更進一步的配合不同框架的 Controller 框架(本次模擬採用 POX Controller[17][18][19] [21])進行不同的處理。詳細設定如圖8、表1。

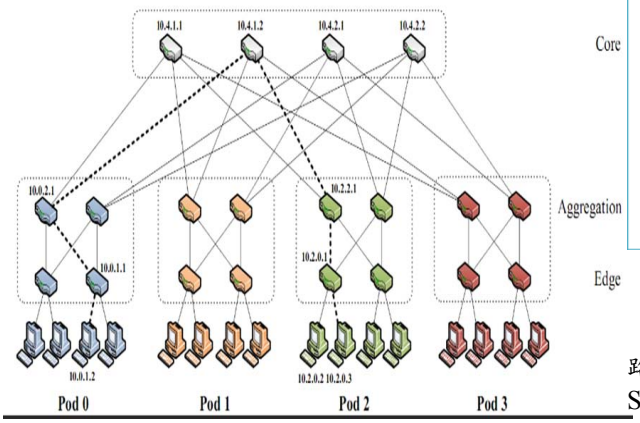

圖 8 拓樸環境

4.2 模擬結果與效能分析

本次模擬採用 Dijkstra's Algorithm (Single path, ECMP 策略)，一般考量利用率(Weight)以及 PWC 演算法進行效能評估，觀察在整體網路 host 在不同負載程度的情況下對於線路利用率以及整體吞吐量的表現。由於單路徑路由在線路狀況及整體附載具有一定水準的情況下容易產生壅塞問題，透過圖9可觀察出在整體負載約70%的時候 Single Path 策略表現與其他 Mutiple path 的策略(ECMP, Weight, PWC)的表現相比開始出現落差，在整體負載80%的時候所有線路成本被視為相等的 ECMP 策略開始出現利用率上升幅度遭遇瓶頸的狀況，而在90%以上的負載情形時，PWC 相較於只考慮線路利用率的 Weight 策略擁有較高的利用率，代表 PWC 比 Weight 策略還要能夠避免瓶頸及重疊問題。

表 1 Mininet 模擬參數

Parameter	Value
k pod of Fat tree	4
Number of Core Switchs	4
Number of Aggregation Switchs	8
Number of Edge Switchs	8
Number of Servers	16
Number of links	48
Each link capacity	1000 Mbps
Routing Policy	Single path, ECMP, Weight, PWC
Routing Alogrithm	Dijkstra's Algorithm, PWC Algorithm

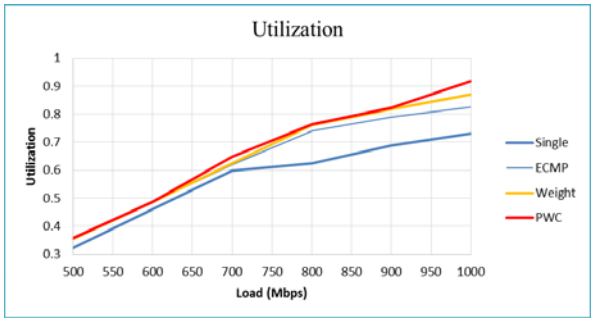

圖 9 Link Utilization

圖10可明顯看出來單路徑路由相較於多路徑路由的吞吐量表現明顯的差距，單路徑路由的 Single path 策略無法有效的提升效能。在多路徑路由的部份，ECMP 與 Weight 兩種策略無法有效的避開壅塞及高負載情況，因此在整體表現上較 PWC 策略差，PWC 在重載的情形下依然可以有優於所有其他策略的表現。

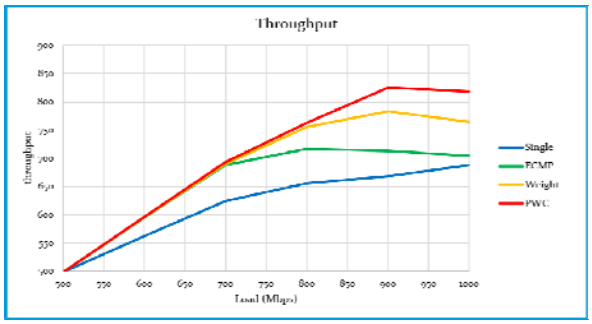

圖 10 Throughput

圖11比對一個 Packet flow 從 Source 端到 Destination 端的延遲時間也可發現 Single Path 策略的表現為最差的，ECMP 策略在重載情形下無法有效的改善延遲時間，而 Weight 與 PWC 兩種考慮權重的策略則擁有較好的表現，並且 PWC 的表現優於單純考慮路徑權重的 Weight 策略。

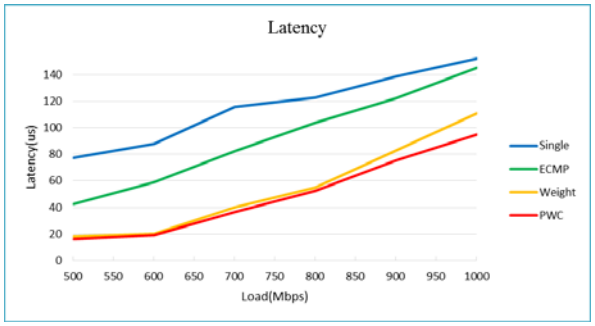

圖 11 Latency

5. 結論與未來工作

在未來會有更多的雲端計算相關應用產生，其運作模式與網路相關技術可以說是日新月異，隨之而來的流量爆炸性增長情形是可預期的。而網路功能虛擬化以及資料中心概在近年來也逐漸成為相關業界的發展趨勢，資料中心的網路管理者如何對於日漸龐大的網路管理方式進行有效率的優化是十分重要的課題之一。多重樹狀結構特性帶來相對複雜的連線佈署環境，考慮其每條路線上的使用狀態並據此對應不同的策略是非常有其必要性的，結合SDN進行決策的相關議題在近年來也逐漸上升。本論文為了解決多重樹狀結構特性可能帶來的網路壅塞及負載平衡議題提出了 PWC 演算法，依據網路負載的使用程度計算相對應的權重值，並利用標準差值作為進階決策的參考數據之一，在網路運行情況下不僅遠勝傳統單路由路徑的表現，對於同為多重路由路徑的其他兩項策略也在高負載時擁有較高的利用率及吞吐量，對於解決壅塞情形及瓶頸問題提供了一項可行的方法，用以應對在資料中心上相對複雜線路的負載平衡問題。

致謝

本文感謝科技部計畫經費補助，計畫編號：MOST 105-2221-E-151-037-MY3。

參考文獻

[1] Cisco Global Cloud Index: Forecast and Methodology, 2015–2020

[2] Data center - Wikipedia

[3] https://www.cs.cornell.edu/courses/cs5413/2014fa/lectures/08-fattree.pdf

[4] Clos network – Wikipedia

[5] OpenFlow – Wikipedia

[6] OpenFlow – ONF, https://www.opennetworking.org/sdn-resources/openflow

[7] OpenFlow Switch Specification Version 1.0.0 (Wire Protocol 0x01), December 31, 2009

[8] OpenFlow Switch Specification Version 1.3.1 (Wire Protocol 0x04), September 6, 2012, ONF, TS-007

[9] http://searchdatacenter.techtarget.com/definition/data-center

[10] Routing – Wikipedia

[11] Multipath routing – Wikipedia

[12] ONF, “Software-Defined Networking: The New Norm for Networks,”*ONF White Paper*, 2012.

[13] B.A.A. NUNES, M. MENDONCA, XUAN-NAM NGUYEN, K.OBRACZKA, T. TURLETTI, “A Survey of Software-Defined Networking:Past, Present, and Future of Programmable Networks,” *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1617-1634,2014.

[14] http://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/7039-1.html

[15] C. E. Hopps, Analysis of an Equal-Cost Multi-Path Algorithm, RFC2992 [Online]. Available: http://tools.ietf.org /html/rfc2992

[16] B. LANTZ, B. HELLER, N. MCKEOWN, “A Network in a Laptop: RapidPrototyping for Software-Defined Networks,” *Hotnets-IX Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 2010.

[17] POX, homepage, http://www.noxrepo.org/pox/about-pox/

[18] POX,openflow.discovery,https://github.com/noxrepo/pox/blob/betta/pox/openflowdiscovery.py

[19] IPERF, homepage, http://iperf.fr/.

[20] Jing Liu, Jie Li, Guochu Shou, Yihong Hu, Zhigang Guo, Wei Dai, “SDN based load balancing mechanism for elephant flow in data center networks,” *Wireless Personal Multimedia Communications (WPMC)*, 2014

[21] K. Srikanth, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, “The Nature of Data Center Traffic: Measurements & Analysis,” *in Proc.ACM SIGCOMM on Internet Measurement Conference (IMC)*,November 2009.