二、研究計畫內容（以 10 頁為限）：

(一)摘要

在傳統的網路中，動態路由演算法會根據特定的條件去找尋最佳路徑。然而，隨著網路流量的增加，鏈路負載狀態的改變會造成當前的路由並非最佳路徑。本研究是要利用軟體定義網路(Software Defined Networking、SDN)的特性，由控制器記錄路由資訊並監控鏈路負載狀況，當鏈路負載的狀況有改變時，控制器可以根據當前的負載重新計算，找到另一條最佳的路由路徑，並以此路由路徑取代原先負載過重之路徑。藉由動態即時改變路徑之後，網路的整體效能將會大幅度的提升。

(二)研究動機與研究問題

路由器的封包轉送，是根據路由演算法，計算出最佳路徑後，再將這些路由資訊放入路由表中。當有封包經過路由器時，則會去比對路由表中的目的位址，一但比對匹配後，則會往適當的介面送出。目前較常見的路由演算法有 RIP(Routing Information Protocol)、OSPF(Open Shortest Path First)、以及 EIGRP(Enhanced Interior Gateway Routing Protocol)等。

RIP 是在 1969 年所提出的路由演算法，其決定最佳路徑的度量為跳點(hop count)，也就是根據經過的路由器的數量來判斷路徑的優劣。由於這是屬於早期的演算法，因此無法根據鏈路頻寬或是壅塞狀況來決定最佳路徑。OSPF 路由演算法，雖然改採用成本的概念來當成度量，其成本的計算是根據頻寬的倒數，因此鏈路頻寬愈大，其成本愈低。利用 OSPF 演算法，相對於 RIP 而言，可以得到較精準的最佳路徑。

不論是 RIP 或是 OSPF 演算法，其用來決定最佳路徑的跳點或頻寬，都是屬於靜態的資訊，也就是一開始就是固定的數值，無法根據當時的壅塞情況自動做調整。在 2013 年思科公司公開的 EIGRP 演算法中，採用了混合的度量值，同時考慮了頻寬、延遲、可靠度以及負載等四個因素。其中的頻寬以及延遲，是屬於靜態的參數，其值與鏈路的頻寬有直接的相關。可靠度以及負載，則是以動態的方式隨時計算更新。可靠度是定義為正確地傳送的封包數；負載是定義為鏈路上目前的負載狀況。在 EIGRP 的路由演算法中，雖然可以同時考慮這四個因素，然而其預設的情況，在計算 EIGRP 度量時，僅僅只有採用頻寬及延遲，也就是並沒有採用動態計算的參數，這是因為如果要一直計算可靠度及負載值的話，則勢必會造成路由器的負擔，造成路由器效能下降。因此，EIGRP 路由演算法，預設的情況並沒有辦法根據鏈路的負載即時變更路由路徑。

在傳統的路由器運作下，由於控制平面(control plane)和資料平面(data plane)是在同一個硬體設備，因此要同時計算動態的負載資訊，並同時找出最佳路由，是一件非常消耗資源的工作，因此大部分的路由協定都是採

用靜態的資訊，來找出最佳路由。然而，SDN 的網路架構，就是將控制平面和資料平面分離，也就是將需要消耗硬體資源來運算的部分獨立出來。有了這樣的特點後，路由協定要動態計算負載資訊，並即時找出最佳路徑，變成是一件容易完成的事情。

因此，本研究就是利用 SDN 的特性，設計出一個以動態負載資訊為基礎的路由演算法，也就是當鏈路發生壅塞的情況時，控制器可以重新計算，找出目前最佳的路由路徑，使封包可以更快速地到達目的地。

(三)文獻回顧與探討

本節將介紹目前常用的路由演算法，包括 RIP、OSPF、以及 EIGRP。

3.1 RIP

RIP(Routing Information Protocol)又稱路由訊息協定，採用距離向量路由協定(Distance Vector Routing)，可以透過不斷地交換訊息讓路由器動態地適應網路連接的變化，適用於小型網路架構。

RIP 的演算方式主要計算設備數目(hop count)，計算來源端到目的端會經過幾個路由器，經過的點愈多則數值愈大，RIP 就會將它視為較差的路徑。因此 RIP 路由協定具備以下特性：

- 採用 Distance Vector Routing 演算法。
- 根據網路路徑所經過的設備數目來決定最佳網路路徑。
- 一條網路路徑最多只允許經過 15 個路由器設備。
- 路由器預設每隔 30 秒互相傳遞網路路由資訊的更新。
- RIP 路由協定支援 Load Balancing 功能，最多支援六條路徑。

RIP 使用 Bellman-ford 演算法，說明如下：

$$d_x(y) = \min_v\{c(x,v) + d_v(y)\}$$

演算法中的參數定義如下：

- $\min_v\{\ \}$：取得 x 的所有鄰居 v。
- $c(x,v)$：鄰居的成本。
- $d_v(y)$：鄰居 v 到目的 y 的成本。

3.2 OSPF

OSPF(Open Shortest Path First)又稱開放式最短路徑優先，是由 IETF 開發，採用鏈路狀態(Link-State)方式收集網路架構，並使用 SPF 演算產生最佳路由資訊記錄到路由表中，適用於較大型網路架構。而 OSPF 有以下特性：

- 能夠快速收斂，在拓撲結構發生改變後立即發送更新訊息。
- 不會造成迴圈。
- 有區域劃分，減少占用的網路頻寬。
- 分四個路由等級，分別為內部路由器(Internal Router)、骨幹路由器(Backbone Router)、區域邊界路由器(Area Border Router)及自治系統邊界路由器(Autonomous System Boundary Router)。

OSPF 的演算法如下：

```
Initialization:
   N' = {u}
   for all nodes v
      if v adjacent to u
            then D(v) = c(u,v)
      else D(v) = ∞

Loop
   find w not in N' such that D(w) is a minimum
   add w to N'
   update D(v) for all v adjacent to w and not in N' :
      D(v) = min( D(v), D(w) + c(w,v) )
   /* new cost to v is either old cost to v or known
    shortest path cost to w plus cost from w to v */
until all nodes in N'
```

演算法中的參數定義如下：

- c(x,y)：從節點 x 到 y 的鏈路成本，若是等於無限大，則不是直接相連。
- D(v)：從來源端到目的端的路徑成本。
- p(v)：來源端到 v 的路徑節點。
- N'：最小成本路徑已知的節點集合。

3.3 EIGRP

EIGRP(Enhanced Interior Gateway Routing Protocol)又稱增強內部閘道路由協定，為 Cisco 公司開發的路由協定，屬於 Cisco 專利，需使用 Cisco 路由器。

EIGRP 主要使用三種表，分別為鄰居表(Neighbor Table)、拓撲表(Topology Table)及路由表(Routing Table)，說明如下：

- 鄰居表：用來記錄直接相連的路由器相關數據。
- 拓撲表：記錄到目的網段的所有路徑。最佳路徑連接的路由器稱為 successor 。
- 路由表：拓撲表中的 successor 會被放入路由表，路由器根據路由表來轉發數據包。

EIGRP 使用四個參數計算成本，分別為頻寬(Bandwidth)、延遲(Delay)、可靠度(Reliability)及負載(Load)，說明如下：

- 頻寬：選取來源端到目的端，所有路由器的出口網路介面最小參考頻寬的值，單位為 kilobits。
- 延遲：來源端到目的端中，所有路由器的出口網路介面上參考延遲的值加總，單位為 microsecond。
- 可靠度：從來源端到目的端中最低的可靠性，使用 keepalives 封包計算。
- 負載：從來源端到目的端中傳送封包的負載度。

上述四個參數帶入以下公式計算成本：

$$\left[\left(K_1 \cdot Bandwidth + \frac{K_2 \cdot Bandwidth}{256 - Load} + K_3 \cdot Delay\right) \cdot \frac{K_5}{K_4 + Reliability}\right] \cdot 256$$

K_1 到 K_5 為用來控制四個參數的計算，而預設情況下 K_1 和 K_3 為 1，其他 K 值為 0，公式可以簡化為(Bandwidth + Delay) * 256。

(四)研究方法及步驟

4.1 研究方法

本研究是要根據鏈路的狀況，進行動態路由協定成本的判斷。因此，必須先將影響鏈路傳輸效能的因素定義出來。在本研究中，定義了五個影響因素，分別為延遲(Delay)、遺失率(Loss)、抖動(Jitter)、往返時間(Round Trip Time，RTT)、以及傳輸量(Throughput)。在本研究中，這五個參數定義如下：

- 延遲(P_1)：封包從鏈路的一端傳送到另一端的時間。
- 遺失率(P_2)：從鏈路一端送出的封包數與鏈路另一端收到的封包數相減後除以總封包數。
- 抖動(P_3)：封包延遲時間的變化量。
- 往返時間(P_4)：封包從鏈路的一端送到另一端後，再送回原發送

端的時間。

- 傳輸量(P_5)：鏈路每秒鐘可以通過多少個 bit 數量。

在 SDN 控制器上，動態地計算每條鏈路的這五個參數，一但計算出來後，我們可以重新定義該條鏈路的成本值。在本研究中，將以下列的公式來計算鏈路的成本：

$$\text{成本 } cost=\sum w_i * P_i$$

在此公式中，w_i為每個參數的權重值。根據此公式，所有鏈路的成本可以依據即時的鏈路狀況重新定義，也就是說，鏈路愈壅塞，其成本值愈高。有了這些最即時的成本值之後，再利用 Dijkstra 演算法找出最佳路徑。最後，控制器新增 flow 到所有的交換器上，更新 flow table 的內容，使封包的傳送路徑可以根據當時鏈路的狀況，經過最佳的路徑來傳送。

4.2 研究步驟

本研究的步驟流程如圖 1 所示，詳細說明如下。

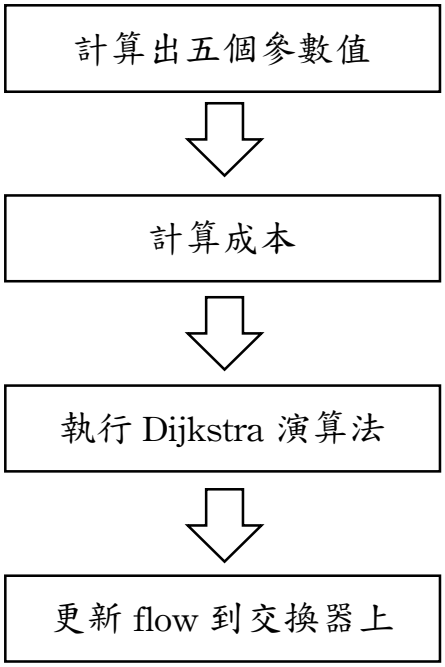

圖 1、研究流程圖

1. 計算出五個參數值：使用 Python 語言，在 Ryu 控制器上撰寫程式，分別把這五個參數值計算出來。
2. 計算成本：利用成本計算公式，將成本計算出來。參數權重值的設定，會經由多次的實驗找出最佳的權重值。

3. 執行 Dijkstra 演算法：利用現有的 Dijkstra 演算法，計算最佳路徑。
4. 更新 flow 到交換器上：控制器將計算出來的最佳路徑，新增到交換器的 flow table 中。

(五)預期結果

本研究將會在 SDN 控制器中同時實現 RIP、OSPF、EIGRP 以及本研究所提的方法。當演算法達到收斂後，開始發送封包測試點對點間封包傳輸的時間。接下來在模擬環境中增加網路流量，利用 iperf 隨機產生大量的交通量，使鏈路逐漸開始壅塞。隨著時間的增加，我們觀察點對點的傳輸時間，預期的結果如圖 2 所示，在鏈路逐漸壅塞之下，本研究所提出的方法，可以得到較佳的傳輸時間。

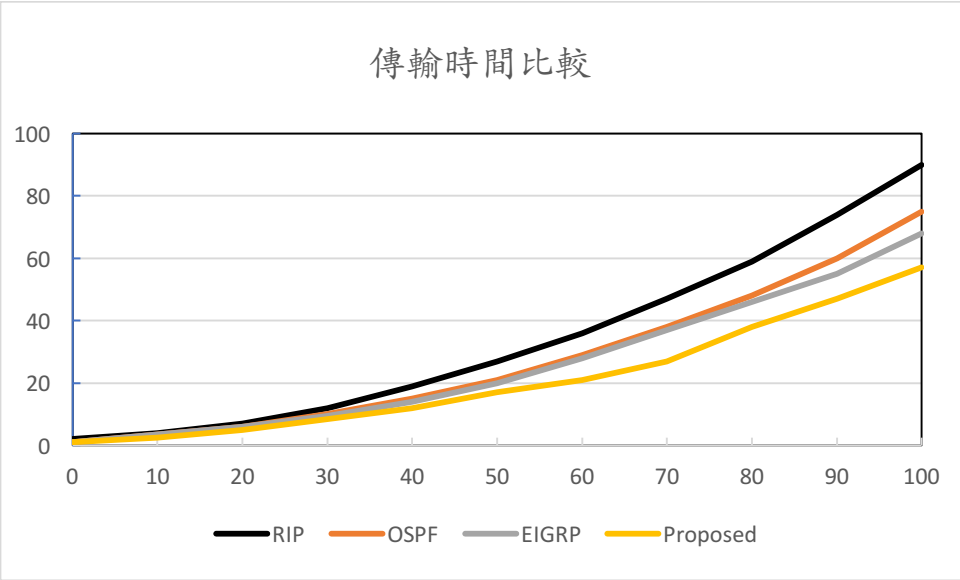

圖 2 傳輸時間比較

(六)參考文獻

[1] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, Vol. 1, No.1, 1959, pp. 269-271.

[2] A. Furculita, M. Ulinic, A. Rus, and V. Dobrota, "Implementation Issues for Modified Dijkstra's and Floyd-Warshall Algorithms in OpenFlow," *Proc. Networking in Education and Research*, 2013, pp. 141-146

[3] B. Lantz, B. Heller, and N. McKeown, "A Network in a Laptop: Rapid Prototyping for Software-Defined Networks," *Proc. ACM Hotnets*, 2010.

[4] Jochen W. Guck and Wolfgang Kellerer, "Achieving End-to-End Real-time Quality of Service with Software Defined Networking," *Proc. International Conference on Cloud Networking*, p.p. 70-76, 2014.

[5] Mininet, http://mininet.org/

[6] Open Network Foundation, https://www.opennetworking.org/

[7] Ryu SDN Framework，http://osrg.github.io/ryu/
[8] Cisco System, http://www.cisco.com
[9] Iperf, http://iperf.fr/

(七)需要指導教授指導內容

1. 專題研究相關方向、內容、問題。
2. 程式寫作相關想法、問題、演算法。
3. 介面架設相關觀念、指令操作、問題。
4. 實驗環境的架設。
5. 正式文件撰寫。