第四章

核心網路之路徑規劃

封包所行經一連串的節點和鏈結，就是所謂的路徑(path)。在傳統的封包網路(datagram)中，封包是以 hop 的方式傳送，每個封包之間是彼此獨立的，並不像 circuit-switching 中資料的傳送需要建立連線，因此在封包網路中，並無路徑的概念。

不過，為了某些目的，網路營運者可能會為一個封包指定一條路徑，此條路徑可能是預先靜態指定或是動態指派。為一個封包找一條路徑的程序稱為路由。某些路由執行方法是由網路營運者依據路由的結果為所有的路由器設定路由表。封包的行進路徑可能根據由路由模組為路由器設計的靜態路徑，也可能根據由路由模組為封包規劃的動態路徑。如果封包能依規劃的路徑行進，確保服務品質之保證將更容易。

但在傳統的封包網路中，對於服務品質的保證並沒有太多的機制來確保。封包在傳送的過程中，在 hop 之中都需要經過檢驗封包的動作，查看其傳送目的地的動作，這處理封包的動作，在傳送經過的每一個節點都需要進行一次，而同一 flow 中的封包，傳送到達目的地的時間不定，另外也可因為網路擁擠或其他因素導致封包遺失的現象，網路只能盡力(Best Effort)將封包送達。此種 Best Effort 的網路在傳統的應用中(如 email)，只需負責靜態資料傳遞的封包網路上是足以應付所需，但若要以 packet-switching 來傳送即時性資料，則會出現嚴重的問題。例如一個即時的 VoIP 應用，若傳遞語音資料的封包到達時間不一，甚至資料封包遺失，則會造成語音斷斷續續，無法讓人舒服的進行一個交談的過程。

如前所述，關於服務品質保證的研究中，多以傳送延遲時間、傳輸延遲抖動和封包遺失率作為服務品質評量的標準，而許多服務品質保證的相關研究被提出以解決上

述的問題，如 IntServ、DiffServ 和 MPLS。 IntServ 使用 RSVP 來即時保留資源給訊務，給予訊務 per-flow 的保證；DiffServ 將訊務在 Ingress 預先分類，而在其內的 core router 則會依據所每一個分類的 PHB(Per-Hop Behavior)來處理封包；而近來十分熱門的 MPLS 技術則是使用 label 將封包在 Ingress 標記，然後在 MPLS Domain 中，各個節點只需要根據封包標記加以轉送，大幅降低封包處理的時間，進而改進封包傳送延遲過長的問題。

可是只提高封包傳送的速度，並不足以保證傳送品質。以常見的服務品質指標 – 傳送延遲來說，在 DiffServ 中，每一個相同分類的封包，在每個 Router 中的轉送服務時，雖然受到某種程度的資源保證(PHB)，但是封包經過幾個節點的轉送後，通過此一 DiffServ Domain 後的總傳送延遲(Total Delay)，卻可能遠超過訊務要求的傳送延遲，對於即時性應用之品質而言，將造成嚴重傷害。且在 DiffServ 的服務中，服務品質的保證是以 Class 為基礎，一旦網路的負擔較重時，在同一 Class 會被丟棄的封包並不知道是屬於何個 flow，無法對 flow 提供絕對的 QoS 保證。因此，如欲提供核心網路內的 End-to-End QoS 予某一應用，應在 DiffServ 機制上加上路徑規劃，引導封包循特定路徑轉送，才能提供 End-to-End QoS。但路徑規劃耗時較久，使用簡單的即時路徑規劃容易導致資源浪費。

反之預先的路徑規劃可以有充裕的時間作較佳的規劃，提高網路效率，加速封包處理速率。本研究以歷史資料預測將來可能發生的訊務，然後預先規劃出符合訊務頻寬和傳輸品質要求的路徑組。在系統運行時透過 source routing 的方法，讓封包在一進入網路的時候，就在指定的路徑上傳送，由於每個訊務所傳送的路徑經過適當的規劃，不但可以將流量分散，也由於封包所經過的路徑已知，所以封包經過整個網域時的傳輸品質是可以獲得控制。

4.1 BBQ 架構中各層級之路徑 (Path Definition)

在 BBQ 的架構中我們以服務品質保證的路徑，來達到服務品質保證，而透過分層

分權的方式，各層元件各自規劃該層級的資源組成所需之路徑，提供給上層元件規劃之用。Short Path 為一條穿越核心網路且提供服務品質保證之路徑，由 Ingress PPA 負責規劃；Long Path 為一穿越 Backbone 且可提供服務品質保證之路徑，由 LPPA 負責規劃。End-to-End Path 則為 end user 到 end user 且實際提供端對端服務品質保證之路徑，由 GACA 負責取得。 表 4.1列出各種在 BBQ 中路徑的名詞定義，及其負責規劃之元件。

表 4. 1：BBQ 架構中各種路徑之定義

規劃路徑	管理元件	路徑能力
端對端路徑 (End-to-End)	接取網路之 Global ACA	End-to-End path 由通過接取網路之路徑和 long path 相連而成，同樣可提供服務品質保證。
長路徑 (Long Path)	各核心網路之 LPPA	Long path 為可通過骨幹網路之路徑，並且可提供服務品質保證。
短路徑 (Short Path)	各核心網路之 SPPA	Short path 為通過核心網路之路徑，並且可提供服務品質保證。

4.2 在 BBQ 架構中的路徑規劃元件 (Path Planning Agent in BBQ Architecture)

在 BBQ 的架構中，負責規劃某個核心網路的路徑之元件稱為 Path Planning Agent (PPA)，每個核心網路中總共有三種 QoS 路徑規劃元件，一為 LPPA，主要是負責 end-to-end 的路徑安排和協調，而在核心網路內的路徑規劃，分別由兩個元件負責，一個為位於 Core Network Coordinator 中的 Short Path Planning Agent(SPPA)，另一為在每一個

Ingress Router 中的 Local SPPA(表 4.1)。

SPPA 位於 Core Network Coordinator 之中，主要的功能是依據各個 Ingress 過去統計的訊務歷史資料計算出一組最能符合未來可能出現訊務需求的路徑，此一組路徑係針對網域內所有資源考量而提供給所有可能進入該核心網路的訊務使用，其次將此一組路徑依其所屬的 Ingress 分割，在據以計算出每一個 Ingress 在每一個鏈結上的頻寬需求。最後將此資訊將交給位於每一個 Ingress 內的 BOA，作為向 BB 批購資源時的依據。

而每個 Ingress 之中另外還有 Ingress PPA 的元件，其作用則是把 BOA 經過協調後所購得位於每一條鏈結上的頻寬資源，組成一組一組的具有服務品質的路徑，儲存在 Ingress 之中的路徑資料庫之中，供負責 end-to-end 路徑規劃的 LPPA 作為規劃 end-to-end 路徑的參考和核心網路內之訊務支援。在系統運作的時候，允入控制元件(ACA)也會根據此一路徑的資料表來為每一個欲進入網域中的訊務挑選符合其頻寬和傳輸服務品質的路徑並指派給該訊務。

因此 PPA 的目標就是依據預計會發生的需求，找出一組可以滿足最多需求的路徑組合，而且此一組路徑必須可以滿足訊務所要求的頻寬和傳輸品質上的要求。在傳輸品質上，我們將以本研究所提出的 Quality Entropy 作為品質評定的標準。

表 4. 2：BBQ 架構中各種 PPA 的比較

名稱	位置	功能
Long PPA(LPPA)	位於 Core Network Coordinator 中	協調具有 End-to-End 的服務品質保證的 Path。
SPPA	位於 Core Network Coordinator 中	負責將規劃出來的路徑頻寬需求，轉換成鏈結上的頻寬需求。
Local SPPA	位於每一個 Ingress Router	主要的功能在於將所擁有的頻寬資源，規劃成不同服務等級的路徑，供應 Run-time 時的 Request 需求。

4.3 路徑規劃的環境假設

在 BBQ 中我們以一個單一的 Quality Entropy 來評量傳送品質的好壞，此一參數是由系統業者依其所需制訂，現階段此一參數的設定並非本研究所著重的部分，為了簡化起見，我們假設 Quality Entropy 具有可加性(additive)，如常見的服務品質保證參數－傳輸延遲(Delay)而言，即為一種可加性的服務品質參數，若一條由 A 至 C 傳送路徑經過 ABC 三個點，則封包從 A 傳送到 C 的傳輸延遲可由 A 至 B 的傳輸延遲加上 B 至 C 的傳送延遲而得。

而在每一個鏈結上，我們假設其可以達到所設定的 Quality Entropy，如果在此鏈結上的所使用的頻寬沒有超過鏈結上的頻寬容量[22]。

4.4 端對端規劃路徑元件與核心網路內路徑規劃元件之互動

在 BBQ 架構中，核心網路只是整個 End-to-End 的一個部分，因此核心網路內的資

源規劃也需要考慮到整個 End-to-End 的需求。在 BBQ 中，負責 End-to-End 路徑規劃的元件為 Long Path Planning Agent(LPPA)，此元件位於每一個核心網路的 CNC 內，是核心網路的對外協調窗口。

在 End-to-End 路徑規劃階段時，各個核心網路之 LPPA 會先以每一個核心網路所可能提供的路徑選擇來規劃 End-to-End 本身核心網路負責處理之訊務的最佳路徑，此預訂的頻寬會成為 Short Path Planning Agent 計算所需頻寬的另一個參考依據(主要的依據是核心網路內的歷史統計資料)，當核心網路內的完成資源規劃的程序後，每一個 Ingress 會擁有真正可以運用的資源，並以路徑的方式儲存在一個路徑的資料庫中，此路徑的資料庫將是 LPPA 在 Runtime 時可以運用來規劃 End-to-End 路徑的資源，圖 4.1 為此一個過程的流程圖：

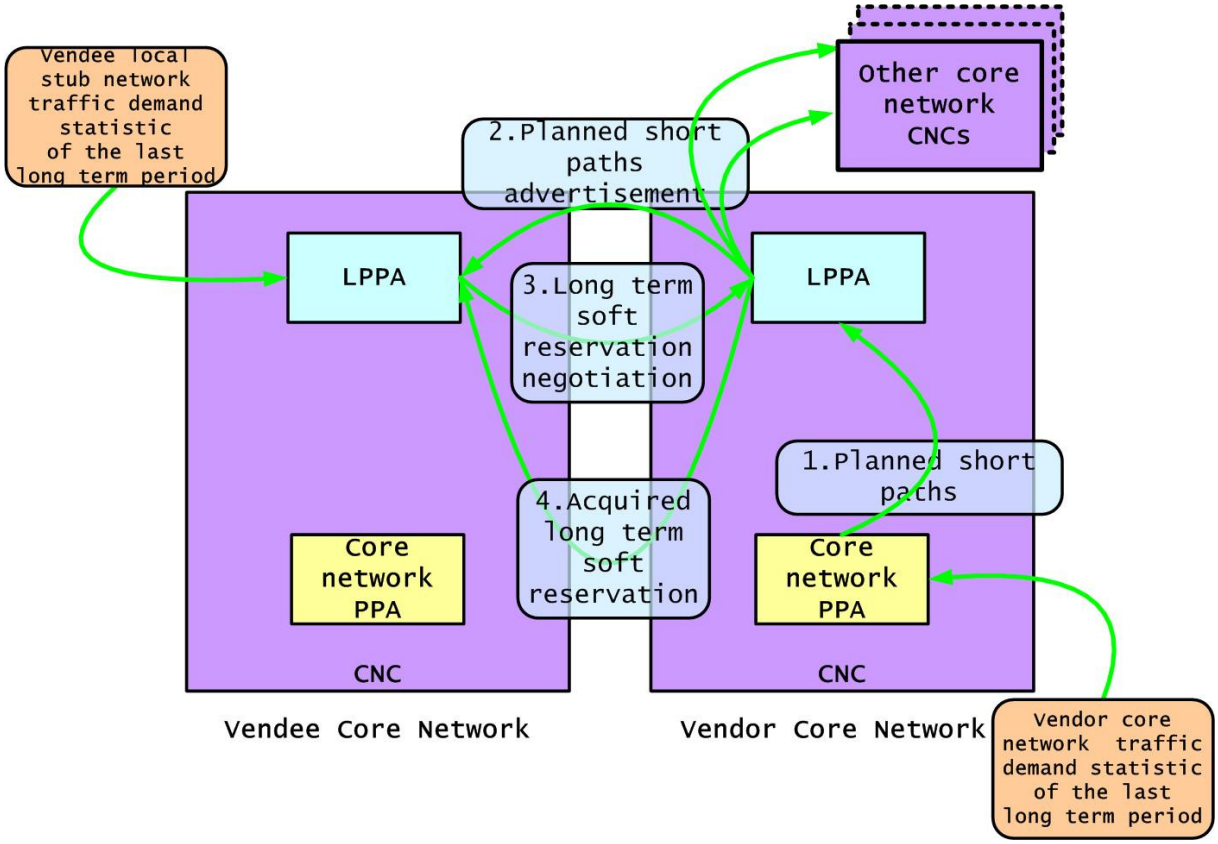

圖 4. 1：路徑規劃元件之互動程序

4.5 核心網路內路徑規劃運作流程 (Path Planning Procedure)

在核心網路內的路徑規劃，分為兩個不同的階段，並由不同的元件所執行，第一個階段是由 SPPA 以預測的資料來做路徑規劃，而第二階段則由位於每一個 Ingress Router 中的 Ingress PPA 負責，依據實際所獲得的資源來做規劃，兩個階段的詳細流程如下。

在第一階段中，SPPA 會依據兩個輸入來做路徑的規劃，一為過去的整個核心網路中同一參考時段的所出現的訊務流量與型態。此統計資料是以過去此網域內，同一個參考時段中所出現過的訊務統計之後所得的結果，會記錄每一個 flow 的資訊，其中包含四個項目，訊務的來源、訊務的目的地、訊務所要求的頻寬和此一訊務所要求的服務品質參數(s, d, b, q)。第二個輸入是整個網域之中的拓樸結構和組成網域中每個鏈結上的資源資訊，鏈結上的資源除了其上可以使用的頻寬以外，還要知道此一個鏈結所可提供的服務品質熵數為何，以本研究目前所著重的傳輸延遲而言，在 SPPA 規劃路徑的時候，需要知道通過每一個鏈結時，所可能遭遇的傳輸延遲為何，這樣規劃出的路徑才可以準確的提供訊務所要求的品質。有了以上兩項參數，SPPA 會以本研究所提的路徑規劃方法，為整個核心網路下一個即將來臨的時段中可能出現的訊務流量規劃出一組路徑(Planned Path Set)，此一組路徑預計可以符合未來運作時段中的訊務需求。

4.5.1 路徑資源需求與鏈結頻寬需求之轉換

歷史資料中記錄流量需求的進入點、離開點、頻寬及品質需求，但資源規劃是以鏈結上的頻寬為單位，由 BOA 向 Bandwidth Broker 批購鏈結頻寬資源，所以統計資料中以進入點、離開點和服務品質需求為分類的流量需求必須轉換成鏈結為基礎之流量需求。因此 SPPA 的主要任務即為規劃出一組路徑，將原始需求依照路徑分解成每一個鏈結上的頻寬需求，交給 BOA 作為批購頻寬的依據。

而 BOA 在批購頻寬的時候，不能只參考一個時段的流量，必須參考多次訊務的流量情形，以需求量的統計分佈作預測才會較為精確，我們從所有 Reference Time Period

中挑出最近一次的訊務流量統計，讓 SPPA 執行計算，求得一組路徑作為基礎，然後我們會輸入以往相同 Reference Time Period 中不同的訊務資源需求量經過這些路徑，得到一個鏈結上的資源需求量之統計，此即為 BOA 計算最佳的頻寬訂購量的基礎，以上為 Path Planning 第一階段的任務。

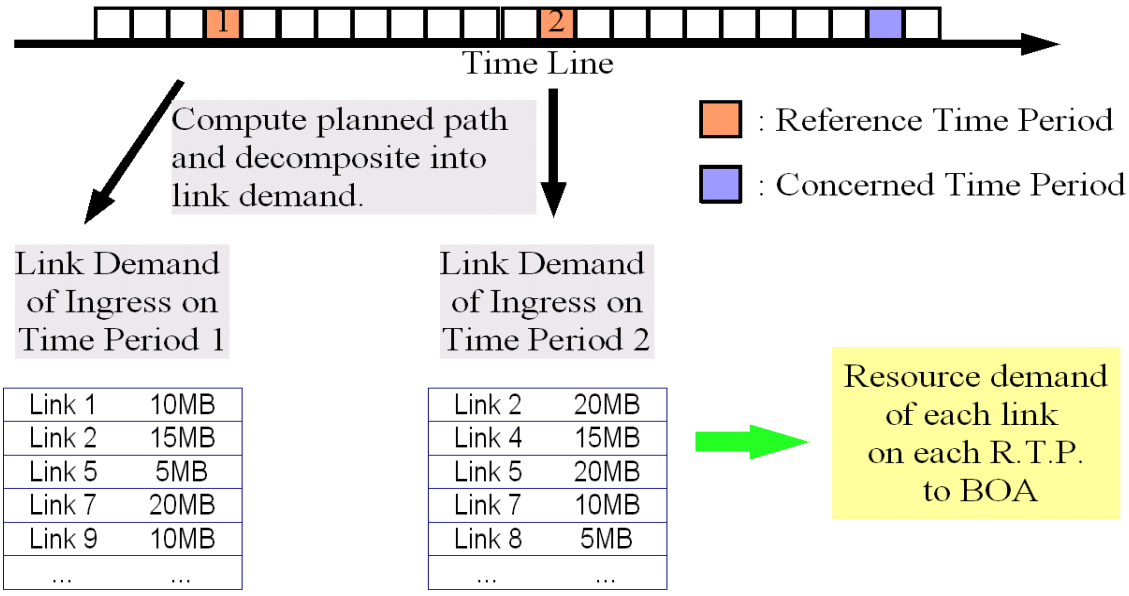

圖 4.2：路徑規劃轉換鏈結頻寬需求之說明

當 BOA 將真正購得的頻寬交給在每一個 Ingress 中的 LSPPA 元件後，則進入 Path Planning 的第二階段，由於所批購得到的資源可能不如預期，因此位於各個 Ingress 中的 LSPPA 元件必須利用所購得的資源來作第二次的路徑規劃，計算時仍然需要參考此一 Ingress 過去的訊務需求來規劃此一 Ingress 在系統運行時所真正可以利用的路徑組，不同於第一階段的路經規劃所參考的是最近一次 Reference Time Period 的統計資料，在

LSPPA 中採用的是過去幾次 Reference Time Period 的平均值，做為路徑規劃的依據，所規劃得出的此組路徑具有頻寬和傳輸品質保證，然後將儲存在各 Ingress 中的路徑 Database 中，供 LPPA 作為規劃 End-to-End 路徑時選擇和系統運作時允入控制元件以此路徑資訊來對欲進入網域中的訊務做允入控制。

表 4. 3：各階段路徑規劃所需之輸入與輸出參數

路徑規劃階段	輸入資訊	輸出資訊
第一階段：SPPA 執行	1. 核心網路之拓樸和每個鏈結上的頻寬和傳輸服務品質參數 2. 整個核心網路內的歷史訊務統計	一組符合最符合網路狀態的路徑組，並會依路徑的出發點分類，然後分解成每一個 Ingress 在每一個鏈結上所需要的頻寬。
第二階段：由每一個 Ingress 中的 Local SPPA 元件所執行	1. 在核心網路中每一個鏈結上所可以運用的頻寬資源和鏈結的傳輸服務品質參數。 2. 此一個 Ingress 在過去 Reference Time Period 的統計訊務量	一組路徑組由許多不同目的地和傳輸服務品質參數的路徑所組成。

4.6 路徑規劃最佳化

路徑規劃的目的在於提供訊務一定的服務品質保證，充分利用網路的頻寬並追求營運者所定義之最大利益，在資源保留路徑規劃方法中，規劃所得之路徑將附有頻寬限制，換言之各鏈結之頻寬將分配並保留予規劃所得路徑。在系統運作時可以保證路徑所擁有頻寬，不會與其他經過相同鏈結的訊務競爭，一條鏈結上所有的訊務流量皆經過一定的規劃控制，而不會超出其上的傳輸能力。

但是要如何評量我們所規劃出來的路徑之好壞是一個重要的議題，因此在本研究中以網路營運者的角度來思考，為我們所規劃出的路徑來作一個衡量，一般說來一個網路營運者莫不希望在既有的網路設備上提供更多傳輸服務，因此若我們可能以一定的網路資源來服務更多的使用者，即表示網路營運者的獲益將會越高，但是每一個訊務所要求的頻寬和傳輸服務品質並不同，越高的傳輸品質代表的是系統保留更多的頻寬來服務訊務的需求。

因此，我們可以為要求進入核心網路中的訊務，依其資源需求定出一個值，此值越高則代表系統需要以更多的資源來滿足訊務的需求，但是同時也是表示若可以滿足的此訊務的需求，我們將可向此訊務收取較高的費用，因此當我們依整個核心網路中過去的訊務流量歷史統計資料規劃出一組路徑後，我們可以依此路經組可以容許通過之所有訊務的價值加總，來衡量此組規劃路徑是否有充分的使用到網路中資源，而整個絕對資源保留路徑規劃的最佳化目標，就是要盡量提高此值。

4.7 最佳化模型 (System Model)

鑒於上述的因素，我們對於 Path Planning 的問題以 Integer Programming 方式呈現，並求最佳解，表 4.4列出路徑規劃相關之符號表說明。

表 4. 4：資源分配路徑規劃符號表

Symbol	Definition
G(V, E)	A directed graph, ***G***, containing |V| nodes and |E| directed links; ***V*** denotes the set of all nodes, and ***E*** denotes the set of all links
v	A node; $v \in \boldsymbol{V}$
e_k	A directed link $e_k = (v_x, v_y) \in E$, v_x is the start node, v_y is the end node of link e_k; also denoted as e_{xy}

Symbol	Definition
$\boldsymbol{R}$	Incoming traffic set.
r_i	A request i, consist of (s_i, d_i, p_i, q_i); $r_i \in \boldsymbol{R}$
s_i	Source node of a request i
d_i	Destinaiton node of a request i
b_i	Bandwidth requirement of a request i
q_i	Quality entropy of a request i
m_i	Profit earn of a admitted request i
P_i	All possible path satisfied constraints of request i
p_{ij}	A path; $p_{ij} \in P_i$
ω_i	The path set satisfied constraints of request i and selected by our algorithm
Ω	The final path set selected by our algorithm
$B_e(p_{ij})$	$= b_e(p_{ij})$, if links of p_{ij} contain link e $= 0$, otherwise
$L(p_{ij})$	The satisfied ratio of request i with p_{ij}
$q(e)$	Quality entropy of link e

絕對資源保留路徑規劃問題描述如下：給定一組網路的拓樸 G，由 node(V)和 link (E)所組成，對於某一個 link e_i，會帶有兩個屬性，一個為此鏈結所能負擔的最大頻寬 b_i，一個為此 link 所能提供的服務熵數 ε_i；另外有一組 Requests (R)，由許多的 request

所組成，對於每一個 r_i 以下列四個項來描述，起點(s)、終點(d)、所需頻寬(b)和我們定義的服務品質熵數(q)。我們定義 Profit m_i 是允許某一個 request i 進入核心網路後所獲得的利益，Profit m_i 是由 b_i 除以 q_i 所得(公式 3.1)，其中 b_i 為 request i 所要求之頻寬，而 qi 為 request i 所要求之服務品質熵數。若一個 Request 所要求的品質(即頻寬 b_i 或服務品質熵數 q_i)越嚴苛，則其所獲得的 Profit 是越大的，所以 Profit m_i 可以說是和要求的頻寬成正比，和服務品質參數 ε 成反比。

$$m_i = \frac{b_i}{q_i} \tag{4.1}$$

當某一個訊務通過允入控制元件進入核心網路，即表示我們可以滿足其對於頻寬和服務品質的要求。所以路徑規劃的最佳化目標即是求得一組可以使我們獲得最大利益的路徑組，我們利用 Ω 來表示此一個路徑組，則路徑規劃的最佳化問題描述如下：對於一組給定的拓樸 G，G 是由一組節點 V 和一組鏈結 E 所組成，另外給定一組 Request R，R 是由 n 個 request 所組成，每一個 request 由四個項所表示起點 s、終點 d、此 request 的頻寬需求量 b 和此 request 的服務品質熵數 q。以數學模型表示，路徑規劃的最佳化目標即是找一組路徑Ω，此組路徑可以使得滿足 request 之 profit 總量為最大(公式 3.2)：

$$max \sum_{i=n} m_i \sum_{p_{ij} \in \omega} L(p_{ij}) = max \sum_{i=n} \frac{b_i}{q_i} \sum_{p_{ij} \in \omega_i} L(p_{ij}) \tag{4.2}$$

此一最佳化的函式有兩個限制式：

一、在所求出的路徑組中，每條路徑所使用的頻寬，不能超過組成網路的每一個 edge(e)所提供的總頻寬，即不等式 3.3：

$$\sum_{e \in \Omega} q(e) \le q_i, \; for\, all\; p_{ij} \in \Omega \tag{4.3}$$

二、而對於某一個被滿足的 request r_i，所挑選出的路徑 P_i，其所帶 QoS 熵數，至少必須滿足此一個 request 的服務品質熵數的限制，如不等式 3.4 所示：

$$\sum_{p_{ij}\in\Omega} b_e(p_{ij}) \le B(e), \; for\, all\, e \in E \tag{4.4}$$

4.8 Greedy Algorithm for Path Planning(G.P.P.A.)

對於上述的問題，可得知路徑規劃的主要目的是找出一組最符合利益的路徑，而且組成此路徑組中的每一條路徑皆需要符合頻寬和品質兩項限制式。如第二章所提及的，在繞徑研究中，在一個網路拓樸中，找出一組符合一個以上限制式的問題，被稱為是 Multi-Constraints Path(MCP)的問題，而一般皆認為此問題是一個 NP-Complete 的問題，所以對於路徑規劃的問題，我們也假設為 NP-Complete。而對於 NP-Complete 的問題我們可知將無法找到最佳解，因此我們在此提出一個 Heuristic 的演算法來解決路徑規劃的問題。此種方法是 greedy method，以下是演算法的步驟：

- 輸入：一個網路拓樸 G(V, E)，一個拓樸是由一組節點 V 和一組鏈結 E 所組成，和一組 Request Set R，R 為 n 個 request 所組成，每一個 request 有四個部分，起點 s、終點 d、此 request 的頻寬需求量 b 和此 request 的服務品質熵數 q。
- 輸出：一組符合最大利益的路徑組Ω。Ω由一組帶有頻寬限制和服務熵數的路徑所組成，此組路徑的 Profit 加總為最大。

1. 從 Request Set 中，選出 Profit 最大的一個 request i ，profit 由 b_i / q_i 計算而得。
2. 利用 Dijkstra Algorithm 為 request i 的 s_i, d_i，找出一條路徑，若我們無法找出一條路徑，則將此 request i 標示為已處理。
3. 檢查由步驟二所找出的路徑中，其組成路徑的所有鏈結是否符合該 request i 之頻

寬 b_i 和 qulity entropy q_i 之限制。

4. 若有鏈結 m 不符 request i 之 b_i 和 q_i 的限制，則暫時拿掉鏈結 m，得到一個暫時的網路拓樸 G'(V', E')，回到步驟二；若此 path 合乎 request i 之限制，則將此 path 加入 W 中，並將 path 標示為已處理。
5. 若還有未處理的 request 在 request set 中，則回到步驟一，直到所有 request 皆為已處理。

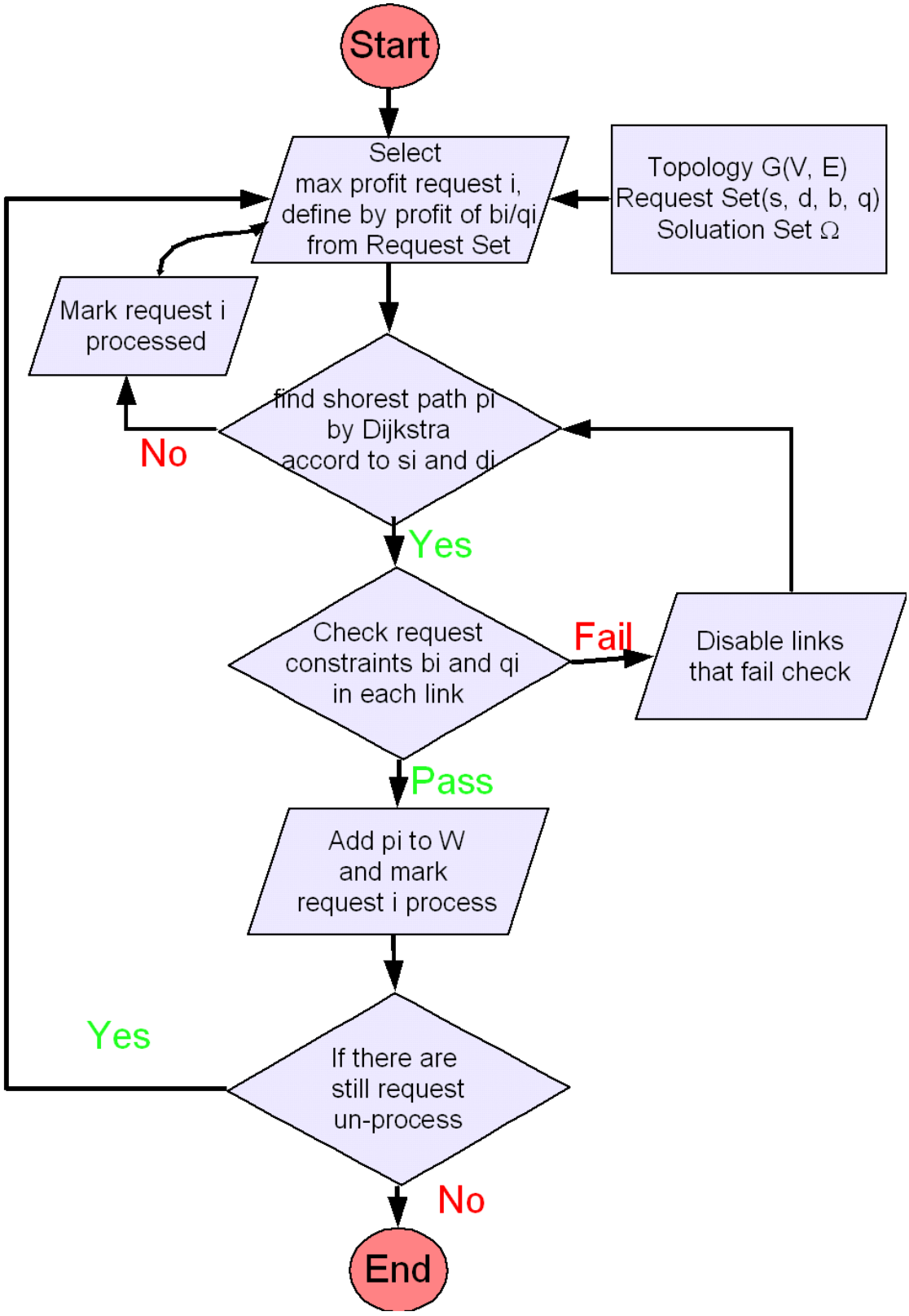
Start
Select
max profit request i,
define by profit of bi/qi
from Request Set
Topology G(V, E)
Request Set(s, d, b, q)
Soluation Set Ω
Mark request i
processed
find shorest path pi
by Dijkstra
accord to si and di
No
Yes
Check request
constraints bi and qi
in each link
Fail
Disable links
that fail check
Pass
Add pi to W
and mark
request i process
Yes
If there are
still request
un-process
No
End

圖 4. 3：演算法流程

4.8.1 演算法虛擬碼 (Pseudo Code)

```
// Check Path Constraints
CheckPathConstraints( path p_i , request r_i)
{
    BWCheck <- TRUE;
    DECheck <- FALSE;

    TOTAL_DELAY <- 0;
    // Check link capacity
    for all e_k in p_i {
      if B(e_k) < B(r_i)
            BWCheck <- FALSE;

      TOTAL_DELAY <- TOTAL_DELAY + D(e_k);
    }
    // Check Total Delay
    if TOTAL_DELAY < D(r_i)
        DELAYCheck <- TRUE;

    if BWCheck and DELAYCheck
        return TRUE;
    else
        return FALSE;
}
```

```
PlanPlanning(G, R)
{
    for k <- 1 to |R|
    {
        // Process Request k
        P(k) <- NIL;
        process <- false;
        While (!processed){
            // Find shortest path for request k
            S(k) <- Dijkstra.shortest.path(G, rk);

            // Check Constraints
            if (P(k)i != NUL){
                Granted <- CheckPathConstraints(S(k)i, );
                if Granted
                    P(k) <- P(k) ∪ S(k)i;
                else
                    DisableUnqualifiedLink(G, S(k)i);
            }
            else
                processed <- TRUE;
        }
    }
}
```

4.8.2 演算法複雜度分析

在路徑規劃的演算法之中，我們根據歷史的訊務資料，對可能會進入網域內之訊務尋找一個符合其要求的路徑，因此需要對每一個訊務都進行一次最短路徑演算法，在本研究中，我們採用的最短路徑演算法為 Dijkstra's shortest path 演算法，因此對一組需有 M 個預測訊務的輸入和 N 個節點之拓樸的路徑規劃而言，其時間的複雜度為 O($M * N^2 \log N$)。而在處理每一個訊務的過程中，我們以 shortest path 演算法所找出的路徑不一定可以符合訊務的需求，因此在每一個尋找的過程之中，可能需要執行超過一次以上的 shortest path 演算法，假設一次的處理訊務過程最多需要額外執行 k 次的 shortest path，則我們的時間複雜在最糟糕的情形下，將會被限制在$(M+k)* N^2 \log N$之下，一般

說來本演算法的時間複雜度為 O($M * N^2 \log N$)。

4.9 小結(Summary)

本章 4.1 節和 4.2 節介紹了核心網路中路徑規劃的問題和路徑規劃的元件，4.3 節中說明路徑規劃的環境假設，4.4 節和 4.5 節中，介紹核心網路中路徑規劃的程序，在 4.6 節中，我們將探討核心網路中路徑規劃的問題，本研究提出一個路徑規劃的最佳化模型則在 4.7 節和中說明，並在 4.8 節提出一個 Heuristic 的方法來解決路徑規劃的問題。接下來在第五章中，我們將以模擬的方式來測試此路徑規劃演算法和 OSPF 的效能比較。