# RSVP 計價系統之設計：分帳處理模型[1]

# The Design of RSVP Billing Syste ：Cost Allocation Mode

陳郁堂 黃日興
台灣科技大學電子系
ytchen@et.ntust.edu.tw rishing@ms3.hinet.net

## 摘要

*本文主旨在探討 RSVP 計價系統之分帳處理，主要針對 Multicast 環境下，用戶端提出不同的 QoS Level（如頻寬）與不同的 Reservation Style，發展計價模型(Cost Model)。依據 RSVP 分帳處理的原則，我們設計兩個計價系統架構：集中式計價系統與分散式計價系統。集中式計價系統由 Server 集中做分帳處理。分散式計價系統則將分帳處理交給沿路的 Router。*

## Abstract

*The objective of this paper is to design a RSVP billing system. First, we develop a cost allocation model that is simple and scalable architecture for allocating multicast costs to receivers. Especially, each receiver of the multicast group can request a different QoS level service (e.g., Bandwidth, Delay, Reservation Style). Based on the proposed Cost Allocation Model, we design two billing architectures: a Centralized Billing System and a Distributed Billing System.*

## 1、緒論

多媒體時代的來臨，在網路上傳送多媒體資料已成為一個重要趨勢，然而現有網路上並無法對多媒體資料提供保證品質的服務。透過 Admission Control、Policy Control 及 Scheduling 的機制，採用 RSVP 資源預留協定來控制網路資源的使用，使用戶在傳送多媒體資料時獲得良好的服務品質。然而 RSVP 架構下，計價是一個重要議題，關係 RSVP 是否為市場接受。

目前網路公司的計價方式是根據使用者租用的專線及使用的時間來計價，此種方式並不能實際反應用戶使用多少資源，從經濟角度來看，Usage-based 計價系統較適合多媒體傳輸，在另一方面，傳統電信業計價方式僅考慮點對點通訊，未將 Multicas 環境列入考慮。

一個良好的計價系統應具備以下三點：

(1) 合理價格：能針對用戶提出的服務與實際使用資源的情形，算出合理的付費價格。
(2) 避免系統負載增加：避免系統因加入計價功能，而導致負載增加。
(3) 正確性與可靠性：計價系統和公司的利益有密切關係，假若計價系統因用戶過多而發生當機，將造成公司莫大的損失，因此計價系統必須保有正確性與可靠性。

本文將探討 RSVP 環境下計價系統採用 Usage-based 的計價方式，主要根據使用者傳送的資料量及要求的 QoS Level 來計價，因此能實際反應用戶使用的資源，並且能針對 Multicas 環境下做分帳處理，使用戶獲得公平合理的費用。另外，在架構設計上，我們直接將計價資料記錄在 RSVP 本身已有的 Message，因此減少額外 Message 的傳送與處理，對系統而言，將不會因為加入計價功能而降低效能。

以往有關計價的研究[14~22]，往往以經濟上的效益，對計價系統做最佳化的數值分析，使系統獲得最大的利益，但並未針對 Differential Services，提出計價處理的方法及步驟，來解決 QoS 實際所帶來的分帳問題。由 Shai Herzog 所提出的 Cost Allocation Model [1]，主要解決在 Multicas 環境下多人共享資源時，需將共享路徑應付價格分配給用戶，其處理原則為每人平均分攤共享路徑之價格，然而 RSVP 的預留環境下，允許用戶提出不同的 QoS Level 與不同的 Reservation Style，若使用 Shai Herzog 所提出的分帳處理模型，將原本以人數做為分配比例，改成以頻寬做為分配的比例，將其運用在 RSVP 環境下，用戶仍無法獲得公平的分配。例如圖 1，由 S1 經由 Router R1 傳送資料到用戶 v1 及 v2，在表 1.1 範例一中，v1 預留 1b，v2 預留 10b 給 S1，在 R1 做 Merge 後，從 S1 至 R1 之路徑會預留 10b 頻寬給 v1 及 v2 共享，由於費用與頻寬成正比，因此假設 S1 至 R1 應付價格為 10k，依頻寬比例來分配價格，在 S1 至 R1 路徑上 v1 須付 10k*{1/(1+10)}=0.909k，v2 須付 10k*(10/11)=9.09k。

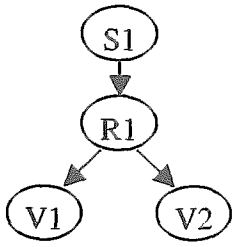


圖 1、頻寬比例分配之範例

表 1.1、頻寬比例分配之說明

| 範例 | 用戶預留頻寬 | | S1 至 R1 路徑 | | 分配結果 | |
|---|---|---|---|---|---|---|
| | V1 | V2 | 預留頻寬 | 應付費用 | V1 | V2 |
| 1 | 1b | 10b | 10b | 10k | 0.909k | 9.09k |
| 2 | 1b | 100b | 100b | 100k | 0.990k | 99.0k |
| 3 | 1b | 1000b | 1000b | 1000k | 0.999k | 999k |

[1] 本研究由國科會 NSC 88-2213-E-011-04 補助支持

要求頻寬不變，v2 要求頻寬愈多，v1 所應付之價格亦增加，如此造成用戶 v1 要求頻寬不變，卻得付較多費用的不公平現象。

針對前述幾篇計價研究的缺失，我們發展出 RSVP 分帳模型，其針對不同 Reservation Style 與 QoS Level，提出分帳演算法來計算用戶應付費用，使用戶獲得最公平及合理的價格。

依據我們所發展的 RSVP Cost Allocation Model，提出兩個計價系統架構：集中式計價系統與分散式計價系統。集中式計價系統主要以 Bernard Aboba 所提出的 Accounting Management Architecture[3]為設計藍圖，將其修改為適合 RSVP 的計價架構，此架構為集中式的分帳處理，主要經由統計資料的收集，利用 Cost Allocation Model 集中計算出用戶端應付費用。分散式計價系統將分帳處理交給沿路的 Router，主要利用 Two-Pass Accounting Mechanis 的執行，讓沿路的 Router 來做分帳處理，以獲得用戶端應付價格。

本論文共分五節。第二節我們提出 Cost Allocation Model。第三節及第四節，則提出實現 Cost Allocation Model 的兩種 RSVP 計價系統架構。第五節為結論。

## 2、Cost Allocation Model

分帳模型主要解決在共享的路徑上，該如何將費用分配給提出不同 QoS Level 的用戶，使用戶獲得合理且公平的費用。本節首先說明為何要分帳，進而依照不同的 Reservation Styl 提出分帳處理演算法，來計算出用戶應付費用。

### 2.1、基本原則

為了能說明本計價系統的處理方法，我們必須定出其處理的基本原則。

(1) 本計價系統只討論 Accounti n 及 Billing 的處理方式，不討論 Security 的方法。
(2) 設計計價模型：解決在 Unicast 或 Multicast 環境下，不同的 QoS Level 與 Reservation Style 所帶來的分帳問題。
(3) 本計價系統，除了計數 Flow 資料量是在資料傳送時 On-Line 處理，其餘的處理，如分帳處理，皆是在資料傳送完畢後 Off-Line 才處理。
(4) 『Usage Fees』：在需付費的 Link，其費用主要為『Usage fees』，也就是根據使用者傳送的資料量來計算費用。

### 2.2、應付費對象

在做計價時，可能因為應用服務類型的不同，而有不一樣的應付費用戶，相對其統計資料量的方法也就不同，因此必須各別分析，以下是一般三種應付費用戶類型。

(1) 『Sender Oriented』:Flow 使用資源的費用由傳送端付費。例如有一演講向 MBONE 做 Broadcast，使在 MBON 上的用戶皆能聽到演講，這種情況下，本計價系統是向傳送端的用戶收費。
(2) 『Receiver Oriented』：Flow 使用資源的費用由接收端付費，如 VOD(Video on Demand)，接收端用戶要求觀賞影片的服務，因此由接收端用戶付費。
(3) 『Group Oriented』：Flow 使用資源的費用由 Group 來付費，如視訊會議 (Video Conferencing)，一般由 Group 提出視訊會議的要求，所以費用是向 Group 要求付費。

#### 2.2.1、三種應付費用戶類型的分析

我們用範例說明三種應付費用戶類型、如圖 2，S1 傳送資料給 v1 及 v2：

(1) 『Sender or Group Oriented』：系統需統計 Link a、b、c 上的價格，因為由 Sender 或 Group 來付費，所以『Link a』『Link b』『Link c』的 Cost 全由 Sende 或 Group 付費。
(2) 『Receiver Oriented』：第一種處理方式是計價系統只計數 v1 及 v2 所收到資料量的多寡，但此種方法只能考慮到 QoS Level 和資料量，但未考慮『距離』的因素。第二種方法是計價系統需統計 Link a、b、c 的 Cost，其中 Link a 的 Cost 需分配給 v1、v2，而 Link b 的 Cost 屬於 v1，而 Link c 的 Cost 屬於 v2。此種方法的優點是除了考慮到 QoS Level 和資料量，也考慮到『距離』的因素。

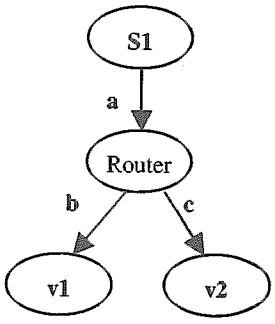


圖 2、v1 及 v2 為用戶端，S1 為 Source 端

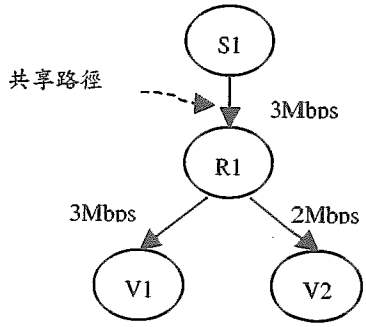


圖 3、共享路徑之說明

Multicas 和 Unicast 在不同的應付費用戶時，其分帳的情形：

(1) Multicast：當『Sender』和『Group』付費時，因為付費對象只有一個，所以將其所要求的 Multicast Grou 使用資源費用算一起；若為『Receiver』付費時，由於其付費對象為數個，當有 Link 有共享的情形時，則必須做分帳處理。
(2) Unicast：Unicast 只需討論『Sender』或『Receiver』付費，不會有『Group』付費的情

形。由於付費的對象只有一個，所以不會有分帳的問題。

2.2.2、發生分帳需求的必要條件

根據上述的說明可知，當下列三項同時發生時，即發生分帳問題。

(1) 由 Receiver 付費：付費對象有多人，因此有分帳問題的發生。

(2) Multicas 環境下，單一 Data Flow 傳送給多個用戶端。

(3) 預留頻寬給同一 Source，使某路徑的頻寬為多人共享。

如圖 3，S1 傳送單一 Data Flow 給 v1、v2，圖中由 S1 至 R1 之路徑預留 3Mbps 給 S1，由 V1 及 V2 兩用戶共享，因此為需分帳路徑，我們將此路徑稱為『共享路徑』。

2.3、基本的分帳處理原則

當一個 Data Flow 送給多個接收者時，則該如何分配傳送 Flow 應付的價格給多個提出不同 QoS level 的接收者呢？我們利用以下的原則來處理：

(1) Data Flow 經過一 Link 的價格只分配給需經由此 Link 的用戶。

(2) 分帳的方式主要依據提出之頻寬來分配，其處理方式是在共享路徑上，對於各用戶預留的頻寬，將其頻寬重疊使用的部份所應付之費用以使用此部份資源之用戶平均分攤費用為原則。

接下來我們將說明在三種不同的 Reservation Style 下的分帳處理。

2.4、參數的定義

為了使我們能清楚的說明 Cost Allocation Model，因此本節定義參數。在我們的 Model 中，我們主要包含三個元件：Network Topology 、Group Membership 和 Routing Function。

(1) N=(V,L,T) 描述 Network N，節點 vi∈V，Directed links(vi,vj) ∈L。

(2) T(N,R,vi) 描述一 Routing Function 在 Network N，其 Source 為 R，Destinati o 為 vi。

(3) $C_{Si}(vi,vj)$ 表示在(vi,vj)的 Directed link 上，由 Si 所傳送 Data Flow 所應付的價格；其中 c：L→ $R_+$。

(4) Cost(vi) 描述節點 vi 應付價格。

目前的 RSVP 支援 Best Effort、Controlled load 和 Guaranteed Service 三種 QoS 的 Request，對於此三種 Service，我們定義其單位價格：

(1) $U_B$：描述在 Best Effort Service 下，每 Byte 的單價。

(2) $U_C(Bw)$：描述在 Controlled Load Service 下，預留頻寬 Bw 所應付的每 Byte 單價。

(3) $U_G(Bw, D)$：描述在 Guaranteed Servic 下，預留頻寬 Bw 且要求 Delay bound D 所應付的每 Byte 單價。

(4) $Amt_K(vi,vj)$：描述 Flow K 在 vi,vj Link 上傳送 Packet 的資料量；單位為 Byte。

在(vi,vj)的 Link 上，flow k 傳送的資料量，乘上單價後，此 Flow 在 Link 所應付的價格。以下為其公式表示：

(1) Best Effort：$C_K(vi,vj) = Amt_k(vi,vj) * U_B$

(2) Controlled Loa ：$C_K(vi,vj) = Amt_k(vi,vj) * U_C(Bw)$

(3) Guaranteed：$C_K(vi,vj) = Amt_k(vi,vj) * U_G(Bw, D)$

利用上述的公式套用，便可得到三種 Service 在 Link 上的費用。

**2.5、Reservation Style**

RSVP 提出三種 Reservation Style，我們首先以基本分帳處理原則設計出 FF Style 分帳演算法。接下來，利用『SE 轉換 FF Style，再以 FF 分帳演算法計價』和『直接分帳』的兩個概念，設計出 SE 與 WF 兩種預留方式的分帳演算法。

2.5.1、Fixed-Filter

Fixed-Filter 為用戶預留一個頻寬給 Source；處理方法是在共享路徑上，對共用此路徑之用戶提出的頻寬，將重疊使用的頻寬值在最大用戶預留頻寬所占的比例乘上共享路徑應付價格，再將此重疊使用頻寬的費用平均分給有使用此部份資源之用戶。

如圖 4 所示，假設有三個用戶，v1 預留 1b、v2 預留 3b、v3 預留 6b，在共享路徑上預留 6b 頻寬，圖 5，v1,v2 及 v3 有 1b 重疊的頻寬，因此有 1/6 的應付費用由 v1、v2 及 v3 平均分攤，v2 及 v3 有 2b 重疊頻寬，因此有 2/6 的應付費用由 v2 及 v3 平均分攤，相同地，有 3/6 的應付費用由 v3 付費，經由此分配算出用戶應付費用。接下來提出分帳演算法，其說明在共享路徑上如何分配的價格。

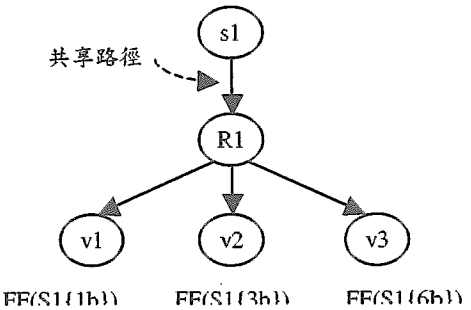


圖 4 FF Sty l 之分帳範例

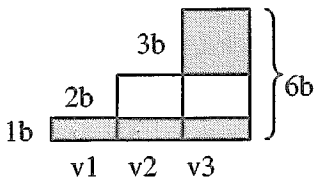


圖 5、分帳示意圖

接下來我們利用上述的處理原則，導出用戶應分配價格的公式。如圖 6，假設 $V_1$ 預留 $Bw_1$、$V_2$ 預留 $Bw_2$、$V_3$ 預留 $Bw_3$...$V_N$ 預留 $Bw_N$ 給 S1。經由 R1 做 Merge 後，S1 至 R1 的 Link 會預留 $Max(Bw_1, Bw_2, ..., Bw_N)$給 S1，因此 S1 至 R1 之路徑為共享路徑，需做分帳。其分帳演算法如下：

Step1：將用戶預留之頻寬值做Sort，由大至小。經排列後，獲得下列：

$B_1 \geqq B_2 \geqq \ldots \geqq B_i \geqq \ldots \geqq B_N$

Step2：產生頻寬對應用戶之 Function User()

$User(B_i)=V_j$ 表用戶 Vj 預留頻寬 Bi 給 S1。

Step3：算出用戶在此共享路徑所分配之價格。

$$Cost\,(User\,(B_N)) = \frac{B_N}{N} \times \frac{C(S1,R1)}{B_1}$$

$$Cost\,(User\,(B_{N-1})) = \left(\frac{B_N}{N} + \frac{B_{N-1}-B_N}{N-1}\right) \times \frac{C(S1,R1)}{B_1}$$

$$Cost(User(B_{N-2})) = \left(\frac{B_N}{N} + \frac{B_{N-1}-B_N}{N-1} + \frac{B_{N-2}-B_{N-1}}{N-2}\right) \times \frac{C(S1,R1)}{B_1}$$

經由上列公式歸納後，可得用戶端應分配之價格：

$$Cost(User(B_i)) = \begin{cases} \left(\frac{B_N}{N} + \sum_{k=i}^{N-1} \frac{B_K - B_{K+1}}{K}\right) \times \frac{C(S1,R1)}{B_1} & \text{If i=1,2,...N-1} \\ \frac{B_N}{N} \times \frac{C(S1,R1)}{B_1} & \text{If i=N} \end{cases}$$

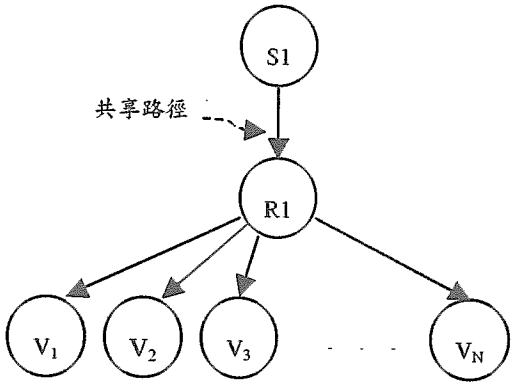


圖 6、推導分帳公式之範例

2.5.2、Shared-Explicit

Shared-Explici 為 Receiver 預留一個頻寬給指定的 Sender，因此在共享路徑上會有不同的 Source 共用頻寬，若用戶們預留頻寬給多個Source 共用，其用戶指定的 Source 皆不同時，造成共享路徑應付的費用分配相當複雜。為解決這問題，我們提出兩種處理原則，第一種將 SE 的預留方法轉換成 FF，再用 FF 的分帳處理算出應付費用；第二種採用直接分帳。接下來根據這二種原則，提出三種處理方法：

(1) Virtual FF Style (VFFS)：將用戶端所要求的 SE 預留類別，利用平均分配的原則，將其轉成 Fixed-Filter 預留類別，再利用 FF 的分帳方法，便可算出價格。其演算方法如下：

Step1：在有 N 個 Source 傳送 Data Flow 的 Multicast 路徑圖，將其分成 N 個單一 Source 傳送 Data Flow 的路徑圖。

Step2：用戶預留 $SE(S_1,S_2,\ldots,S_N\{Bw\})$，即預留一頻寬 Bw 給 N 個 Source，我們將其視為預留 $FF(S_1\{Bw/N\})$ $FF(S_2\{Bw/N\})\ldots FF(S_N\{Bw/N\})$

Step3：利用 FF 的分帳演算法算出價格。

(2) Virtual Differential FF Style (VDFFS)：與 VFFS 相似，不同在於：雖然用戶端使用 Shared Style 並指定 Source，但某些路徑並未有其指定的 Source 經過，因此對於用戶所提之頻寬，在不同共享路有不同的預留頻寬值，相對價格分配的比例亦不同。其演算法如下：

Step1：在有 N 個 Source 傳送 Data Flow 的 Multicast 路徑圖，將其分成 N 個單一 Source 傳送 Data Flow 的路徑圖。

Step2：用戶預留 $SE(S_1,S_2,\ldots,S_N\{Bw\})$，即預留一頻寬 Bw 給 N 個 Source，在不同的共享路徑，我們將其視為預留 $Bw/N_{Shared}$ 。

其中：$BW_{SE}$：為用戶端所提出的 SE 預留頻寬；

$N_{Shared}$：為在 Link 上用戶端有幾個 Source 做 Shared；

Step3：在不同的共享路徑，依照 Step2 所視的預留頻寬，使用分帳演算法，算出分配價格。將不需分帳路徑的價格加在唯一使用此頻寬之用戶的費用上。

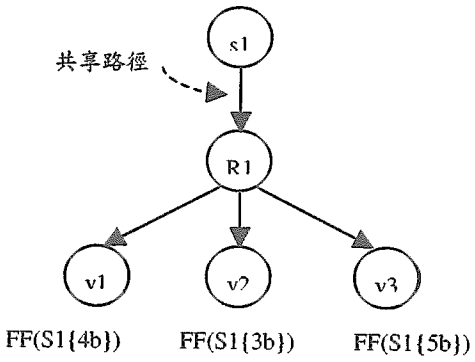


圖 7 FF Sty1 之分帳處理

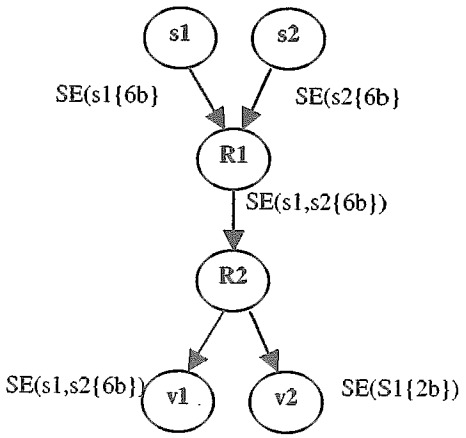


圖 8 用戶端使用 SE 的 Reservation Style

(3) Direct Cost Allocation (DCA)：不需要轉送成 FF Style 的模式，直接分帳。其演算法如下：

Step1：將分帳路徑價格依分帳演算法算出分配價格。

Step2：將不需分帳路徑的價格加在唯一使用此頻寬之用戶的費用上。

範例一：如圖 8 中，為 s1 送 Data Flow 給 v1、v2，而 s2 送 Data Flow 給 v2，我們使用 VFF 的處理方法。假設 v1 預留 6b 的頻寬給 s1 和 s2,v2 預留 2b 的頻寬給 S1。其演算步驟如下：

Step1：將 Multicas 圖拆開成 s1 與 s2 兩個傳送路徑圖，如圖 2.7

Step2：(1) v1：SE(s1,s2 {6b}) 視為 FF(s1{3b}) 及 FF(s2{3b}) 預留。

(3) v2：SE(s1{2b}) 可視為 FF(s1{2b})。

Step3：利用分帳演算法分別算出每部份用戶應付價格，再將兩部份之用戶應付價格相加，即獲得用戶應付價格。

(1) 左圖由 S1 傳送 Data Flow 給 v1、v2。

$Cost_{S1}(v1) = 2/3*\{ C_{S1}(s1,R1)+ C_{S1}(R1,R2)\} + C_{S1}(R2,v1)$

$Cost_{S1}(v2) = 1/3*\{ C_{S1}(s1,R1)+ C_{S1}(R1,R2)\} + C_{S1}(R2,v2)$

(2) 左圖由 S2 傳送 Data Flow 給 v1。

$Cost_{S2}(v1) = C_{S2}(s2,R1) + C_{S2}(R1,R2) + C_{S2}(R2,v1)$

(3) 各用戶應付價格如下：

$Cost(v1) = Cost_{S1}(v1) + Cost_{S2}(v1)$ ；

$Cost(v2) = Cost_{S1}(v2)$

範例二：如圖 8 中，為 S1 送 Data Flow 給 v1、v2，而 s2 送 Data Flow 給 v1，由於 v1 與 v2 的預留，實際上會在 Router 上做 Merge 動作：

a. R2 將 v1 及 v2 提出的 Request 做 Merge 後，會在 Link(R1,R2)預留 SE(s1,s2{6b})，

b. R1 會在 Link(S1,R1)預留 SE(s1{6b})，在 Link(S2,R1)預留 SE(s2{6b})，

我們使用 VDFF 的處理方法。其分析如下：

Step1：將 Multicas 圖拆開成 s1 與 s2 兩個傳送路徑圖，如圖 9。

Step2：(1) 在 Link(s1,R1)的價格會分配給 v1 與 v2，其中將 v1 視為預留 6b，v2 預留 2b 給 s1。

(2) 在 Link(R1,R2)的價格會分配給 v1 與 v2，其中將 v1 視為預留 $BW_{SE} \div N_{Shared} = (6b)/(2)=3b$，v2 預留 2b 給 s1。

Step3：利用分帳演算法將各共享路徑之分配價格算出

(1) 左圖由 S1 傳送 Data Flow 給 v1、v2。

$Cost_{S1}(v1) = 5/6*C_{S1}(s1,R1) + 2/3*C_{S1}(R1,R2)+C_{S1}(R2,v1)$

$Cost_{S1}(v2) = 1/6*C_{S1}(s1,R1) + 1/3*C_{S1}(R1,R2)+C_{S1}(R2,v1)$

(2) 左圖由 S2 傳送 Data Flow 給 v1。

$Cost_{S2}(v1) = C_{S2}(s2,R1) + C_{S2}(R1,R2) + C_{S2}(R2,v1)$

(3) 各用戶應付價格如下：

$Cost(v1) = Cost_{S1}(v1) + Cost_{S2}(v1)$

$Cost(v2) = Cost_{S1}(v2)$

範例三：如圖 7 中，為 s1 傳送 Data Flow 給 v1、v2，而 s2 送 Data Flow 給 v1，我們使用 DCA 的處理方法。其分析如下：

(1) 需分帳路徑為 Link(R1,R2)與 Link(s1,R1)，將其價格的總和依分帳演算法分帳。

(2) $C(R1,R2) = C_{S1}(R1,R2) + C_{S2}(R1,R2$ ；將 Link 上的 Cost 相加；

(3) $Cost(v1) = 5/6 * [C_{S1}(s1,R1)+C(R1,R2)] + C_{S2}(s2,R1) + C(R2,v1)$

(4) $Cost(v2) = 1/6 * [C_{S1}(s1,R1) + C(R1,R2)] + C_{S1}(R2,v2)$

接下來，我們來比較三種分帳方法：

(1) 公平性：VDFF ＞VFFS＞DCA，原因是 VDFF 在不同 Billing Link 能實際反應用戶端使用的資源，而給予價格分配的比例，最能表現出公平性。

(2) 實作複雜度：VDFF ＞VFF ＞DCA，由於 VDFF 在不同 Billing Link 可能會有不同分配的比例，當 Flow 所經過的路徑越長，複雜度也就越大。

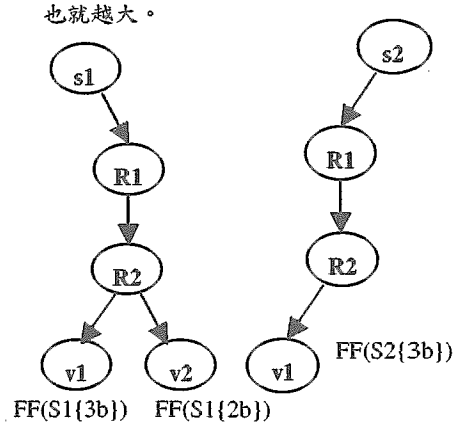


圖 9 經由 VDFF 轉換後的兩個部份

2.5.3、Wildcard-Filter

Wildcard-Filter 為 Receive 預留一個頻寬給所有要送資料給他的 Source，基本上 Wildcard-Filter 的分帳問題亦可利用之前 SE 所提出的三種方法來處理。

2.6、動態加入和離開

在 Multicast 環境下，若有用戶端動態的加入和離開，在共享路徑上應付的價格 該如何公平的分配呢？以下我們針對不同的付費對象，分析動態加入或離開對分帳處理的影響，並提出解決方法。

2.6.1、『Sender or Group Oriented』之分析

對單一付費對象的 Sender 或 Group Oriented 而言，由於沒有分帳的問題，因此動態的加入和離開不會增加系統的困擾。如圖 9 用戶端 v3 後來加入，計價系統去統計 Link d 的 Cost，再將此 Cost 加在應付的價格中，便可獲得應付價格。

2.6.2、『Receiver Oriented』之分析

在 Multicas 的環境下由 Receiver 來付費，會有共享路徑分帳的問題，因此當用戶動態的加入和離開，其分帳的複雜度相對增加。接下來我們提出兩種方法來解決分帳問題。

2.6.2.1、精確計價

此方法處理原則是當用戶動態加入或離開時，便做一次分帳處理。其演算法如下：

第一部份：當某用戶動態加入或離開時，其處理步驟如下：

Step1：將目前共享路徑之價格記錄下來。

Step2：在用戶加入或離開之前，依原用戶所提出的頻寬，利用分帳演算法算出分配比例。

Step3：依分配比例算出原先用戶應付費用。

Step4：從新計算在用戶加入或離開之後的共享路徑價格。

第二部份：當 Multicas 結束後，將每次用戶加入或離開時所分配之價格相加，即可獲得用戶應付價格。

經由演算法的計價，便可解決動態加入或離開的問題。

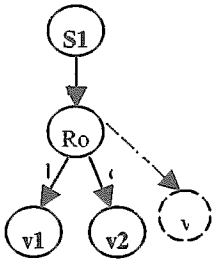


圖 10、v1 及 v2 為用戶端，S1 為 Source 端，v3 為動態來加入

範例說明：如圖 10，S1 傳送資料給 v1 及 v2，在 Link a 的 Cost 需由 v1 及 v2 分帳，圖 11 為用戶動態加入或離開之說明，圖中所示，若以用戶加入或離開為時間點，其有三個不同的時段，處理方式如下：

(1) 從開始到用戶 v3 加入的時段：此時段 Link a 的價格需分配給 v1 及 v2。

(2) 從 v3 加入到 v2 離開的時段：此時段 Link a 的價格需分配給 v1、v2 及 v3。

(3) 從 v2 離開到結束：此時段 Link a 的價格需分配給 v1 及 v3。

從範例中可知，當有 N 個用戶動態加入或離開時，則有 N+1 個時段需做分帳處理，因此當用戶動態加入或離開愈多，則不同分帳對象的時段增加，對系統的 Overhea 也增加。

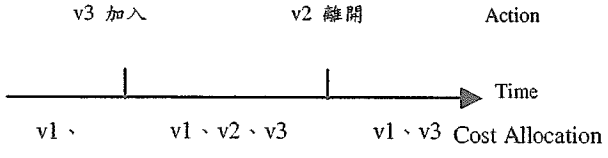


圖 11、在 Link a，用戶端在時間區間對 Cost 分帳的情形

2.6.2.2、忽略動態加入或離開之計價

此方法處理原則是不管用戶何時加入或離開，都將其視為一開始加入及最後離開，不考慮動態加入與離開的時間點。其演算法如下：

第一部份：當某用戶動態加入或離開時，其處理步驟如下：

Step1：若為新加入之用戶，記錄其要求的頻寬以供 Multica s 結束後計算分配比例。

Step2：若為離開之用戶，則不做處理。

第二部份：當 Multica s 結束後，其處理如下：

將共享路徑的價格利用分帳演算法分配給所有共享過此路徑的用戶，如此便可算出每個用戶應付價格。

經由演算法的計價，便可解決動態加入或離開的問題。

範例說明：不管圖 10 中的 v3 何時加入或 v2 何時離開，我們視 v2 及 v3 與 v1 一樣一開始加入及最後離開，這樣在做分帳時，Link a 全部的價格會分配給 v1、v2 和 v3。當 Actio n增加，並不會增加系統的 Overhead。

## 3、集中式計價系統

接下來我們將於第 3 及 4 節分別提出兩個計價系統架構，本節描述集中式計價系統，所謂的集中式計價系統是集中式的分帳處理，由 Accounting Route 及 Accounting Server 收集用戶端傳送資料的資料量，再由 Billing Server 依 Cost Allocation Mod e 算出用戶端應付的費用。

集中式計價系統架構包括了『Router』、『Accounting Server』、『Billing Server』三個部份，如圖 12 所示。以下為其工作的項目：

(1) 『Accounting Router』：除了有一般 Router 的功能外，主要負責在需付費的 Link 上，收集每個 Flow 傳送資料的資料量，在 Flow Teardwo n後，則將此收集的資訊（即 Accounting Dat ）利用 Batch 方式傳送給 Accounting Server。

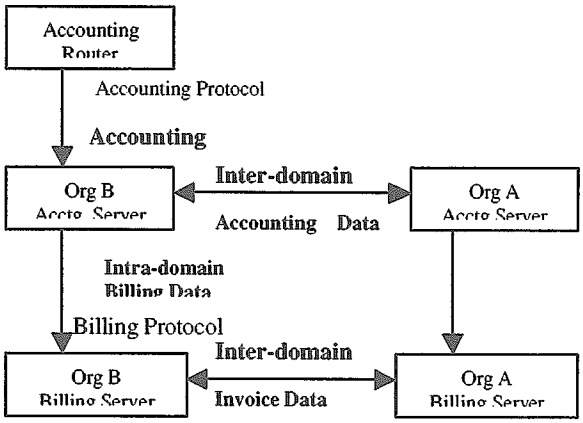


圖 12、集中式計價系統架構

(2) 『Accounting Server』：主要處理由 Accounting Router 送來的 Accounting Data，來獲得 Flow 在 Link 上傳送資料量，並判斷 Accounting Data 是屬於那一個 Domain，將 Accounting Data 送給所屬的 Domain 做計價。另外 Accounting Server 可從 Router 獲得『Routing Tree』，再將此圖以 Batch 的方式傳送給 Billing

Server。

(3) 『Billing Server』：定出各 Service 單位價格；根據 Accounting Serv e 所送來的『計價路徑圖』，利用我們所提出的 Cost Allocation Model，算出各用戶應付的費用。此外 還需與其它 Domain 的 Billing Serv e 做分帳。

## 4、分散式計價系統

分散式計價系統是屬於分散式的分帳處理，主要利用 Two-Pass Accounting Mechanis ，經由兩次 Message 的傳送，將分帳處理的工作交給資料傳送路徑中的 Router 來處理。

本節首先說明 Two-Pass Accounting Mechanism，再描述分散式計價系統架構之動作流程。

### 4.1、Two-Pass Accounting Mechanism

Two–Pass Accounting Scheme 在 Router 收到 Reservation Messa g 後，需將用戶端預留之 QoS Level 記錄，並使用分帳演算法，將分支點的分帳比例算出。當 Flow 結束時，則利用 Accounting Messag 沿著資料路徑傳送，此 Message 記錄了一筆預留 Flow 沿路應付的價格，從 Source 傳送到所有的 Receivers，沿路若遇到分支點時，便在分支點解決價格分配的問題。

#### 4.1.1、參數定義

Out(vi,vj)：表示在 vi 將價格做分配，應該送給 vj 的 Cost。其記錄在 Accounting Message。

#### 4.1.2、Two-Pass Accounting Mechanism 演算法

第一部份：Reservation Messa g 的傳送

Step1：用戶端發出 Reservation Messa g 給 Router，Router 需記錄以本身為 Sub-tree root 的以下所有用戶預留的頻寬值。此項步驟需將 Reservation Messa g 做修改(如附錄)，經修改後的 Messag e需額外記錄數個未 Merge 之前的 FlowSpec (Type 為 254)，如此將使 Router 能獲得用戶原本預留的頻寬值，以提供共享路徑分帳比例之計算。

Step2：當 Router 記錄完以本身為 Sub-tree root 的以下所有用戶預留的頻寬值，便可利用分帳演算法，算出共享路徑費用之分帳比例。

第二部份：Flow 傳送結束，Accounting Messag 的傳送

Step1：Accounting Mess a g 由 Sende 端所發出。

Step2：每當 Message 到達分支點，則利用第一部份已算出的分配比例，將分配價格算出，再產生 Accounting Messag 將其分配的價格各別送給各分支。

Step3：接收端收到記錄用戶應付費用的 Accounting Message。

經由兩個部份的處理完成 Two-Pass Accounting Mechanism。

### 4.2、分散式計價架構

分散式計價系統提出的計價架構裏包括了『Router』、『Accounting Router』及『Billing Server』三個部份，如圖 13 所示。以下為其工作的項目：

(1) 『Router』：除了要有 Routing 的功能，還要利用本論文所提出的 Cost Allocation Model，對計數的資料量做分帳計算。

(2) 『Accounting Router』：除了有 Routin 及 Cost Allocati o 的功能外，主要負責在需付費的 Billing Link 上，收集每個 Flow 傳送的資料量。特別重要的是分散式計價系統是規定資料量的統計是由資料經過 Billing Link 之後的 Accounting Route 來計價。當 Flow 資料傳送完後，表示 Router 將此 Flow 傳送資料量收集完成。

(3) 『Billing Server』：主要在計算及記錄各用戶端應付的價格。將獲得用戶端所應付的價格，直接記錄至 Billing Datab a s 中。

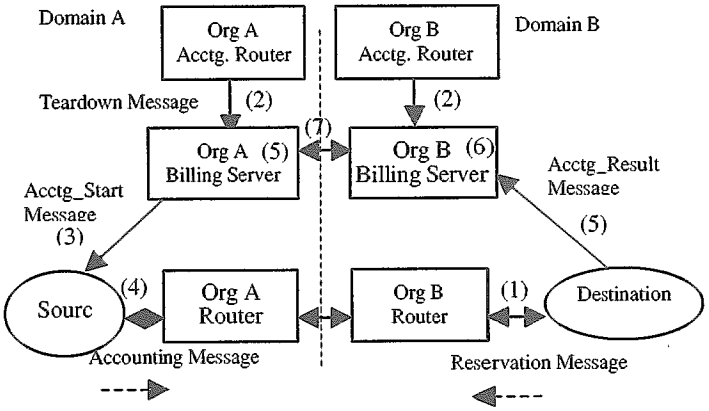


圖 13、分散式計價系統架構

分散式計價系統架構之動作流程如下：

第一步驟、用戶端提出 QoS Level，發出 Reservation Message，開始執行 Two-Pass Accounting Mechanism 的第一部份，使用分帳演算法，將分帳比例算出，並計錄 Router 中。

第二步驟、當 Accounting Route 收到 Path Teardown Message, 表示 Flow 傳送資料結束，其統計資料量已完成。因此 Accounting Router 將 Path Teardown Messag 轉送給 Billing Server，告知此訊息。

第三步驟、Billing Server 傳送 Accounting_Start Message 要求 Source 端傳送 Accounting Message，開始處理 Two-Pass Accounting Mechanis 的第二部份工作。

第四步驟、由 Source 送出的 Accounting Messag 會依照之前資料傳送的路徑傳送，利用 Two-Pass Accounting Mechanism 第二部份的執行算出用戶端應付費的價格。

第五步驟、當 Destinatio 端接收到 Accounting Message 時，即表示已算出 Destination 應付費的價格，Destination 將此結果利用 Accounting Result Messa g 送給 Billing Server。

第六步驟、Billing Server 收到的資料即為用戶端應付的價格

第七步驟、利用 Inter-Domain Invoice Message 來傳送 Domain 與 Domain 之間的分帳費用。

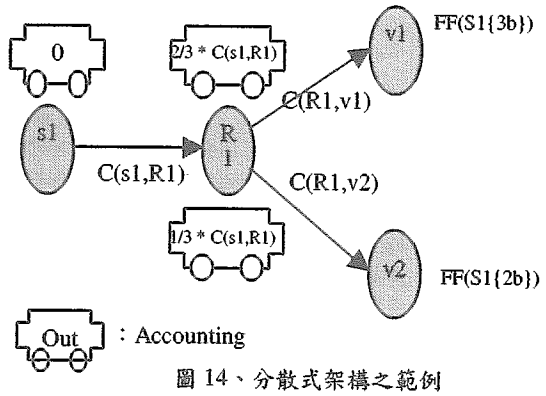


圖 14、分散式架構之範例

4.3、範例說明

如圖 14，由 S1 送資料至 v1、v2。根據上述步驟，我們來說明此範例的動作流程：

(1) v1 及 v2 發出 Reservation Message，v1 預留 3b、v2 預留 2b 給 S1。當 R1 收到 Reservation Message，將預留資料記錄下來，並利用分帳演算法算出分帳比例。

(2) 當 Flow 傳送資料結束，Accounting Route 告知 Billing Server，其統計的資料量已完成。

(3) Billing Server 要求 Source『S1』端傳送 Accounting Message。

A. S1 送 Accounting Messag 給 R1，Out 值為 0。
Out(s1,R1)=0

B. 當 R1 收到 Accounting Message，則依先前計算的分配比例算出用戶應付價格。
Out(R1,v1) = 2/3*C(s1,R1) 記錄在新產生的 Accounting Mess a g 給 v1
Out(R1,v2) = 1/3*C(s1,R1) 記錄在新產生的 Accounting Mess a g 給 v2

C. 當 Accounting Messag 從 R1→v1 時，v1 可算出其應付的資料量：
Cost(v1) = Out(R1,v1) + C(R1,v1) = 2/3 * C(s1,R1) + C(R1,v1)

D. 當 Accounting Messag 從 R1→v2 時，v2 可算出其應付的資料量：
Cost(v2) = Out(R1,v2) + C(R1,v1) = 1/3 * C(s1,R1) + C(R1,v1)

(4) 當 Destination『v1』『v2』端接收到 Accounting Message 時，並算出 Destinatio 應付費的費用，Destinati o 將此結果送給 Billing Server。

(5) Billing Serv e 獲得用戶端所應付的價格。

## 5、結論

本論文針對 RSVP 環境下提出 Cost Allocation Model，主要在群組環境下，依據不同的 Reservation Style 與用戶端提出 QoS Level，發展不同的分帳處理方法。在 FF Styl 下，以頻寬比例來分配共享路徑之價格。在 SE Style 下，提出兩種原則，第一種原則將 SE 轉換成 FF Style，再以 FF Styl 的處理原則算出用戶應付費用；第二種原則為直接分帳不做轉換。在 WF Style 下，則可直接使用 SE Style 的方法來分帳。

依據我們所設計的 RSVP Cost Allocation Mod e 提出兩個計價系統架構：集中式計價系統與分散式計價系統。集中式計價系統為集中做分帳處理，在分帳處理上，其複雜度高，而分散式計價系統由沿路的 Router 做分帳處理，其可直接利用 RSVP Message 的傳送來做分帳處理，大大簡化分帳處理之複雜度，因此分散式計價系統非常適合 RSVP 環境。

## 參考文獻


[1] Shai Herzog, Scott Shenker and Deborah Estrin. "Sharing the Cost of Multicast Trees: An Axiomatic Analysis" To appear in ACM SIGCOMM'95 Conference, August 1995, Cambridge, Massachusetts, USA.

[2] Shai Herzog. "Accounting and Access Control for Multicast Distributions: Models and Mechanisms" August 1996.

[3] Bernard Aboba and Jari Arkko. "Introduction to Accounting Management" Internet-Draft, File: Draft-Aboba-acct-00.txt, 6 August 1998.

[4] C. Mills. "Internet Accounting: Background" RFC: 1272, Novermber 1991.

[5] Richard J. Edell, Nick Mckeown and Pravin P. Varaiya. "Billing Users and Pricing for TCP" IEEE JOURNAL on selected areas in Communicati ons, VOL. 13, NO. 7 September 1995, pp. 1162-1175.

[6] K. Ravindran and Ting-Jian Gong. "Cost Analysis of Multicast Transport Architectures in Multiservice Networks" IEEE/ACM Transactions on Networking, VOL. 6, NO. 1, February 1998, pp. 94-109.

[7] Ron Cocchi, Scott Shenker, Deborah Estrin and Lixia Zhang. "Pricing in Computer Networks: Motivation, Formulation, and Example" IEEE/ACM Transactions o Networking, VOL. 1, NO.6, December 1993, pp. 614-627.

[8] R. Braden, Ed, L. Zhang, S. Berson, S. Herzog and S. Jamin. "Resource ReSerVation Protocol (RSVP) – Version 1 Funcational Specification" RFC 2205, September 1997.

[9] R. Braden, D.Clark and S. Shenker. "Integrated Services in the Internet Architecture: an Overview" RFC: 1633, Jun 1994.

[10] J. Wroclawski "The Use of RSVP wi th IETF Integrated Services" RFC: 2210, September 1997.

[11] S. Shenker "Specification of Guaranteed Quality of Service" RFC: 2212, September 1997.

[12] J. Wroclawski "Specification of the Controlled-Load Network Element Service" RFC: 2211, September 1997.

[13] G. Apostolopoulos, R. Guerin, S. Kamat, A.Orda, T. Przygienda, D.Williams "QoS Routing Mechanisms and OSPF Extensions" In ternet Draft File: draft-querin-qos-ospf-04.txt, December 23, 1998.

[14] P.B. Key and D.R. McaAuley "Differential QoS and Pricing in netwroks: Where flow control meets game theory" IEE Proc-Softw., Vol. 146, No. 1, February 1999.

[15] Andrew Oldlyzko, AT&T Labs "Paris Metro Pricing: The minimalist differentiated services solution" IEEE 1999.

[16] Leanne P. Breker Carey L. Williamson "A Simulatio Study of Usage-Based Pricing Strategies for Packet-Switched Networks" IEEE 1996.

[17] N. G. Duffield, S.H. Low "The Cost of Quality i Networks of Aggregate Traffic" IEEE 1998.

[18] Errub W, Fulp, Maximilian Ott, Daniel Reininger, Douglas S. Reeves "Paying for QoS: An Optimal Distributed Algorithm for Pricing Network Resources" IEEE 1998.

[19] Martin Karsten, Jens Schmitt, Lars Wolf, and Ralf Steinmetz "An Embedded Charging Approach for RSVP" IEEE 1998