一個 OVSF 多碼延時分配機制與重配置效能之研究

周恭立
中華大學資訊工程系
m9002009@chu.edu.tw

嚴力行
中華大學資訊工程系
lhyen@chu.edu.tw

摘要

在 IMT-2000 的 WCDMA 行動通訊網路中，以 OVSF (Orthogonal Variable Spreading Factor) 碼作為通道碼 (Channelization Code)。OVSF 碼的管理分為三個議題：分配 (Assignment)、配置 (Placement) 與重配置 (Replacement)。其主要目標皆在於減少使用者頻寬要求被拒絕的次數。在本篇論文中，我們假設使用者可以同時使用多個通道碼，提出通道碼延時分配機制，透過向前觀察 (Look-Forward) 的策略，選擇阻斷率較低的碼分配結果。我們使用模擬的方式來驗證此方法能夠有效地降低阻斷率。另外，在多碼環境下碼重配置所需的換碼次數，我們也透過實驗進行了效能分析。關於通道碼的處理順序與其換碼次數，實驗結果顯示必要時才作重配置之換碼次數少於重配置優先處理及重配置最後處理的策略；對於有多個候選通道碼情形 [16]，實驗結果顯示可以選擇阻斷率不會改變但換碼次數最少的通道碼。

關鍵詞：OVSF、分配、配置、重配置、延時分配

一、簡介

回顧行動通訊網路的發展進程，從服務型態來看，在傳統語音服務需求之外，數據服務也日趨重要。從數據服務的進展來看，傳輸速度由第二代 GSM 9.6Kbps、2.5 代 GPRS 171.2Kbps，進展到第三代 UMTS 2Mbps，使得數據服務內容可以愈多元化，且讓需要高頻寬的多媒體應用更加成熟。實體層多重存取技術 (Multiple Access) 也從第一代 AMPS 網路之 FDMA (Frequency Division Multiple Access) 、第二代 GSM/GPRS 網路之 TDMA (Time Division Multiple Access) 與 cdmaOne(IS-95) 之 CDMA (Code Division Multiple Access)，進展到第三代 WCDMA、CDMA2000、TD-SCDMA。目前研發中的行動通訊網路幾乎全是 CDMA 的天下。

第三代行動通訊網路可以支援高傳輸速率 (High Data Rate) 與可變傳輸速率 (Variable Data Rate) 服務。當使用者處在較高的移動環境，數據傳輸速率在 144Kbps 至 384Kbps 之間；反之，則速率可達 2Mbps。在 IMT-2000 (International Mobile Telecommunications in year 2000) 多項草案規格中，以 IMT-DS 最被看好能夠成為未來的主流，即 WCDMA [1]。

WCDMA 採用了 DS-CDMA (Direct Spread Code Division Multiple Access) 無線傳輸技術，每個傳輸通道都使用了彼此間具有正交 (Orthogonal) 的通道碼 (Channelization Codes)，使得不同實體通道間可以使用相同的頻率而不互相干擾。根據 3GPP (3rd Generation Partnership Project)規格書 [2] 所描述，通道碼是根據 OVSF (Orthogonal Variable Spreading Factor) Code Tree，以不同的展頻因子 (Spreading Factor; SF) 所產生的。不同展頻因子的通道碼提供不同傳輸速率的通道，以滿足需要不同資源的應用。3GPP 規格書定義三種 DS-CDMA 傳送機制 (Transmission Schemes) [4]：多碼 (Multi-Code) [5][6]、單碼 (Single-Code) [7] 及混合型 (hybrid) [8][9]。單碼傳送機制是指使用者只能用一個通道碼來傳送資料，而多碼就是數個通道碼的組合。多碼的使用者設備必須配備多個 RAKE Combiner，方可同時使用多個通道碼。至於混合型即結合上面兩種機制，對於較低頻寬需求使用單碼傳送機制，對於較高頻寬需求則使用多碼傳送機制。

關於 OVSF Code 的管理，一般可分為三個議題：分配 (Assignment) [5][13][16]、配置 (Placement) [14] 與重配置 (Replacement) [14]。分配決定滿足使用者速率需求之通道碼。配置指決定通道碼後，該如何於 OVSF Code Tree 中配置可用的碼。此兩種機制的主要目標皆在於減少使用者需求被拒絕的次數（即阻斷）。一般而言，阻斷發生的原因有二：系統頻寬不足與碼碎裂 (Code Fragmentation)。所謂碼碎裂是指現有系統頻寬可滿足使用者的請求，但由於可用的碼過於分散以致於無法分配給使用者。重配置即透過換碼的動作來解決碼碎裂的問題。所謂換碼是指回收原來已分配出去的碼，代以相同頻寬的另一個碼，目的在使系統未分配出去的碼與回收的碼能

夠集中成為頻寬較大且使用者能夠使用到的新碼，降低使用者被阻斷的機會。在滿足降低阻斷率的主要目標前提下，這些機制也設法來提高碼的使用率。

我們在先前的研究中提出一個多碼分配演算法 [16]。實驗顯示此方法的阻斷率及換碼次數皆很不錯，不過我們發現這個議題仍有一些待研究之處。首先，一次處理一個請求的方式有時並不能得到最低的阻斷率。例如現在依序有三個請求 Req1、Req2、Req3 分別要求頻寬 100、40、60。如系統目前剩餘頻寬為 120 且以循序處理方式處理，則 Req1 可以成功被分配，而 Req2 和 Req3 會因系統頻寬不足而被拒絕。如果我們可以延後處理 Req1 的請求，那麼我們可能就可以有足夠的資訊來降低阻斷率。以此例而言，如果我們阻斷 Req1 使得 Req2 和 Req3 可以成功被分配，顯然能夠得到較低的阻斷率。為此我們在本論文中提出一個能夠有效降低阻斷率的通道碼 “延時分配” 機制，以不改變要求順序 (Request Order) 為前提，透過向前觀察 (Look-Forward) 的策略，選擇阻斷率較低的碼分配結果。實驗顯示此機制配合任何多碼分配演算法，均可再降低阻斷率約 0.5% 至 1.5%。

此外，於配置分配到的多碼時，可能會因碼碎裂而導致其中某些通道碼需進行重配置動作。先前的作法並不考慮此種狀況，而直接依通道碼的展頻因子由小到大進行配置。此篇論文中，我們考慮了其他的可能性，包括優先處理需要進行重配置動作的通道碼及最後才處理需要進行重配置動作的通道碼。我們針對這幾種可能性進行了實驗分析，結果顯示還是原先的作法其換碼次數為最少。

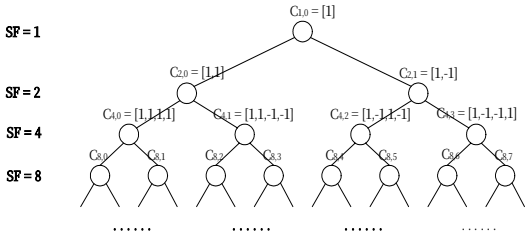
最後，之前的作法在有多個多碼候選者時，會選擇造成系統最小碼碎裂的候選者。此種作法經實驗證明可有效減少系統碼碎裂情形，進而降低阻斷率。不過當配合重配置機制一同運作時，因重配置機制可完全解決碼碎裂問題，挑選造成系統最小碼碎裂的多碼候選者即變得多此一舉。在本研究中，我們嘗試在此情況下直接挑選換碼次數最少的多碼候選者。實驗結果證實此種策略可將換碼次數減少 2% 至 5%。

本文架構如下：第二節描述 OVSF Code Tree 的背景知識與通道碼相關議題研究。第三節則詳述本篇所提之通道碼延後分配機制。第四節我們透過模擬來分析改進效能及重配置分析。最後是本文的結論。

二、背景知識與相關研究

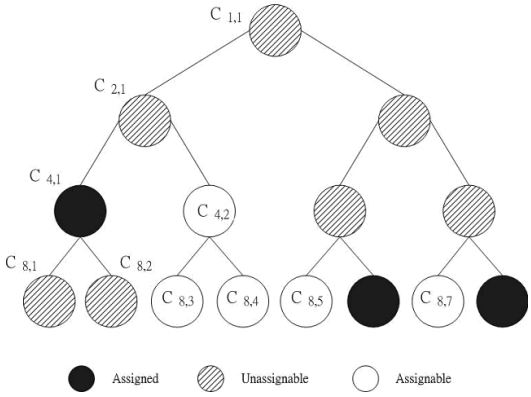
2.1 OVFS Code Tree 及其管理

在 CDMA 系統中，傳輸通道碼彼此之間具有正交 (Orthogonal) 性質。這種性質保證了通道之間的傳輸不會互相干擾。依 3GPP 規格書 [1] 所定義，通道碼是以 OVSF Code 方式來實現，圖一為一 OVSF Code Tree [16]。圖中每個節點 (Node) 代表一特定通道碼。，每一階層 (Level) 分別表示不同的展頻因子 (Spreading Factor; SF)。同一階層中所有的碼具有相同的展頻因子(即有相同頻寬)。每一階層碼的頻寬大小皆為下一層節點的 2 倍。



圖一：OVSF Code Tree

為了保持每個碼在不同階層間的正交關係 [13]，OVSF Code Tree 中的一個碼 *c* 在滿足下面兩個條件時才可以分配給使用者 (Assignable)：(1) *c* 到根節點 (Root) 之間沒有別的碼已被分配 (Assigned) ；(2) *c* 的所有子樹 (Sub-tree) 中沒有任何已被分配的碼。因為有這項特性，在一個 OVSF Code Tree 中，一條路徑 (Path) 上只有一個碼可以被分配給使用者。一旦這個碼被使用了，路徑上的其他的碼即無法再被使用 (Unassignable)。此項特性降低了系統頻寬的使用率。

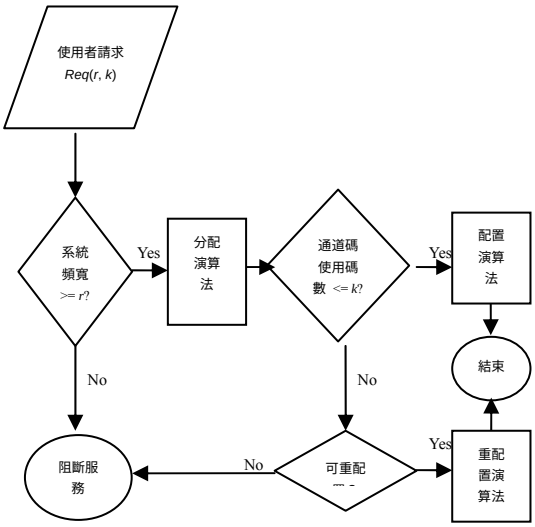


圖二：OVSF Code Tree 三種屬性節點

圖二顯示了 OVSF Code Tree 三種屬性節點 [15]。圖中 $C_{4,1}$ 是一個已被分配 (Assigned) 的節點，為了維持正交特性使得 $C_{1,1}$、$C_{2,1}$、$C_{8,1}$、$C_{8,2}$ 均成為不可分配 (Unassignable) 的

節點。而 $C_{4,2}$、$C_{8,3}$、$C_{8,4}$、$C_{8,5}$、$C_{8,6}$ 是可分配 (Assignable) 的節點 [16]。

通道碼分配的方法可分為單碼 (Single-Code) 分配及多碼 (Multi-Code) 分配兩種。當使用者設備只有一個 Rake Combiner 時，只能使用一個通道碼來傳輸資料，因此只要在 Code Tree 上找出一個能夠滿足頻寬需求的節點配置即可。在多碼分配方面，由於使用者分配了兩個以上的碼數，可以使用兩個以上的通道碼進行資料的傳輸，因此在進行通道碼分配時就較單碼分配有彈性，但過程也較為複雜。



圖三：多碼頻寬&碼數請求之流程

圖三顯示了多碼請求之處理流程。使用者請求 Req(r, k)表示需求頻寬 r 而最多可同時使用 k 個碼。系統首先會檢查剩餘頻寬是否足夠分配。若剩餘頻寬小於 r 時，此請求即會被拒絕。否則呼叫分配演算法，決定欲分配給此要求的頻寬組合 (Codeword)。由於系統擁有的頻寬單位均為 2 的次方數，若使用者要求之頻寬不為 2 的次方數之組合時，將不可避免地會分配多於 r 的頻寬給使用者，造成頻寬的浪費。

當決定好要分配的頻寬組合後 (組合數目可能多於一組以上)，即檢查這些可分配組合所用到的碼數是否有不大於 k 者。若有，則選取最佳者呼叫配置演算法將其組合中所用到的碼自 OVSF Code Tree 中一一配置出來。若可分配組合所用到的碼數均大於 k，則視系統是否有重配置機制將此請求阻斷 (Block) 或呼叫重配置演算法。

由上述流程可知使用者所提出的頻寬請求被拒絕的原因有二：頻寬不足及碼碎裂 (Code Fragmentation)。當使用者的頻寬要求多於系統剩餘頻寬時，即為頻寬不足；當現有系統頻寬組合可滿足使用者的請求，但因組合所用到的碼數大於使用者可同時使用者，即為碼碎裂。

2.2 相關研究

目前已有多個多碼分配演算法被提出。Cheng and Lin [12] 提出的多碼分配演算法中，偏好能夠留下最多大頻寬碼的 Codeword。若多於一個以上的 Codeword 可造成系統擁有最多大頻寬碼，則選擇使用碼數較少的 Codeword。此方法簡單但執行速度相當快。在[13]中，作者首先將使用者請求之頻寬 r 化作二進位數字當作欲選的 Codeword，再根據使用者可用的碼數目 k 進行 Codeword 的調整，以確保所選的 Codeword 所需碼數目不會超過 k。這個方法可能配置過多的大頻寬碼給使用者，造成頻寬浪費。我們在之前的研究中曾提出一個多碼的分配演算法 [16]。透過 Dynamic Programming 的技巧，我們可以很有效地於事前找出滿足特定請求 Req(r, k) 的所有頻寬組合。處理請求時，再從這些頻寬組合中挑出造成系統最小碼碎裂者進行配置。若有兩個以上頻寬組合均造成系統最小碼碎裂，則選擇組合中使用碼數較少者。實驗顯示這個方法的碼阻斷率是上述方法中最小的。配合重配置演算法運作時，所需換碼的次數也是最少的。

經過多碼分配演算法挑選出 Codeword 後，此 Codeword 中所含的通道碼該在 OVSF Code Tree 中的哪個節點中配置，也關係著系統使用率的問題。一個最直接的作法，就是套用現有的單碼配置演算法，逐一配置選取 Codeword 中所含的通道碼。[14]中，作者提出三種單碼配置策略：

- 隨機配置 (Random)：從 OVSF Code Tree 相符 SF 中任選一可分配節點來配置。
- 最左優先配置 (Leftmost)：配置 OVSF Code Tree 相符 SF 中最左邊的可分配節點。
- 擁擠優先配置 (Crowded-first)：從 OVSF Code Tree 相符 SF 中，挑選祖先節點的子樹所擁有的可分配節點數目最少的可分配節點。

由 [14] 的模擬實驗結果可知此三種單碼配置策略的效能優劣依序為：擁擠優先、最左優先、及隨機配置。

當碼碎裂問題發生時，透過重配置策略可降低因碼碎裂而造成阻斷發生。在[10]中，作者根據重配置執行時機將重配置應用分成主動 (Proactive) 與被動 (Reactive) 兩種。所謂主動是指於下一個請求進來之前或某個碼被釋放後，系統就進行碼重配置動作；所謂被動是指當系統無法直接配置某請求所需的碼時，才進行碼重配置動作。在[14]中，作者指出一個好的重配置策略所花費代價應最少，即換碼次數應愈少愈好。目前公認最佳重配置策略演算法為 DCA [9]。

三、延時分配機制

本節主要詳述我們所提出的兩個通道碼延時分配機制。我們的機制要求系統有一個 Queue 以暫存使用者的請求。

3.1 方法一 (M1)

當系統收到某 Request A，根據系統是否可分配足夠頻寬給 A 分為兩個狀況處理：

狀況 1. 有足夠頻寬與碼數可分配給 A。此時將系統頻寬減去欲分配給 A 的頻寬，並依 Queue 目前狀態決定處理方式，如表一。

表一：M1 有足夠頻寬可分配於 Request 之處理原則

Queue 狀態	處理方式
空	將此可分配的 Request A 暫存於 Queue 中
已有可分配的 Request B	配置 Request B 所需之碼。並將此可分配的 Request A 暫存於 Queue 中。
有無法分配的 Request B	嘗試新排程的可能性（後述）

狀況 2. 因頻寬不足或碼碎裂而無法分配給 A，依表二處理。

表二：M1 頻寬不足或碼碎裂之處理原則

Queue 狀態	處理方式
空	阻斷服務
有可分配的 Request B	將此無法分配的 Request A 暫存於 Queue 中
有無法分配的 Request B	將此無法分配的 Request A 暫存於 Queue 中

表一中嘗試新排程的方式，乃嘗試強制阻斷 Queue 中第一個可分配的 Request，然後重新計算新排程的阻斷次數、重配置次數及剩餘頻寬。重新計算完後，無論新排程是否較佳，均將 Queue 完全清空。判斷排程是否較佳的原則如下：

(1) 阻斷次數較小為佳。
(2) 若阻斷次數一樣，換碼次數較小為佳。
(3) 若換碼次數一樣，剩餘頻寬較多為佳。
(4) 若剩餘頻寬一樣，碎裂碼個數較小為佳。
(5) 若上述皆相同，則原排程較佳。

圖四執行範例中的 ok、er1、er2、er3 分別代表可分配、頻寬不足、碼碎裂、強制阻斷。假設目前系統剩餘頻寬 256。事件編號 120 要求頻寬 126，結果為可分配（分配頻寬 128）。A 暫存於 Queue 中，系統剩餘頻寬 128。事件編號 121 要求頻寬 127，結果為可分配（分配頻寬 128）。配置先前事件編號 120 所需之碼，並將事件編號 121 請求暫存於 Queue 中。系統剩餘頻寬 0。事件編號 122 要求頻寬 72。因系統頻寬不足，此請求暫存於 Queue 中。接下來的事件 123、124、1、2 皆因頻寬不足暫存於 Queue 中。接下來因釋放先前配置給事件編號 120 的頻寬 128，使得系統剩餘頻寬增為 128。事件編號 3 要求頻寬 112 且系統剩餘頻寬 128，結果為可分配，此時嘗試新排程的可能性。新排程將暫存之事件編號 121 作強制阻斷，然後重新計算之後的配置情形。結果阻斷次數由原先的 5 次降為 4 次，因此我們取新排程結果，將事件 121-124 與 1-3 的結果輸出，清空 Queue。

事件編號	要求/釋放頻寬	要求碼數	系統剩餘頻寬	結果	新排程剩餘頻寬	新結果
120 Req	126	2	256	ok		
*121 Req	127	2	128	ok	128	er3
122 Req	72	4	0	er1	128	ok
123 Req	62	2	0	er1	56	er1
124 Req	19	3	0	er1	56	ok
1 Req	19	1	0	er1	37	ok
2 Req	48	2	0	er1	5	er1
120 Release	128		0		5	
*3 Req	112	2	128	ok	5	er1

圖四：方法一執行範例

3.2 方法二 (M2)

方法一的缺點為 Queue 的長度不固定。方法二限制 Queue 中最多只有一個暫存請求。當系統收到某 Request A，則根據系統是否可分配足夠頻寬給 A 分為兩個狀況處理：

狀況 1. 有足夠頻寬與碼數可分配給 A。此時將系統頻寬減去欲分配給 A 的頻寬，並依 Queue 目前狀態決定處理方式，如表三。

表三：M2 有足夠頻寬可分配於 Request 之處理原則

Queue 狀態	處理方式
空	將此 Request A 暫存於 Queue 中
有可分配的 Request B	配置 Request B 所需之碼，並將此 Request A 暫存於 Queue 中
有無法分配的 Request B	系統錯誤

狀況 2. 因頻寬不足或碼碎裂而無法分配給 A，依表四處理。

表四：M2 頻寬不足或碼碎裂之處理原則

Queue 狀態	處理方式
空	阻斷服務
有可分配的 Request B	嘗試將 B 強制阻斷的新排程，若 A 變成可分配，則取新排程結果輸出。
有無法分配的 Request B	系統錯誤

圖五的例子中，假設系統剩餘頻寬 256。事件編號 120 要求頻寬 126，結果為可分配（分配頻寬 128）並暫存於 Queue 中。系統剩餘頻寬 128。事件編號 121 要求頻寬 127，結果為可分配（分配頻寬 128），先配置事件編號 120 所需之碼，並將事件編號 121 暫存於 Queue 中。系統剩餘頻寬 0。事件編號 122 要求頻寬 72，因頻寬不足無法分配。此時嘗試新排程的可能性：將事件編號 121 作強制阻斷，然後重新計算事件編號 122 的頻寬要求，新結果變成可以被分配。因此我們取新排程結果。

使用者編號	要求/釋放頻寬	要求碼數	系統剩餘頻寬	結果	新排程剩餘頻寬	新結果
120 Req	126	2	256	ok		
121 Req	127	2	128	ok	128	er3
122 Req	72	4	0	er1	128	o

圖五：方法二執行範例

四、模擬實驗與結果

本節分為四小節。4.1 節定義實驗環境。4.2 節模擬第 3 節所提方法的實驗結果。4.3 節進行重配置分析。4.4 節討論有效延遲之效能分析。

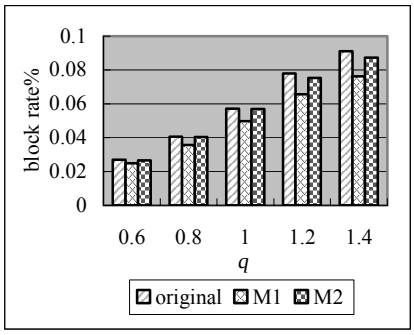
4.1 實驗環境

我們透過模擬實驗的方式來評估本文所提演算法之效能，並將我們的方法 (M1 與 M2) 與 TY [16] 、SLC [13] 及 CL [5] 三種分配演算法做比較分析。這五種方法皆須單碼配置演算法之協助，我們採用兩種單碼配置演算法：擁擠優先 (CF) 與最左優先 (LM)。重配置演算法則採用 DCA [9]。我們預設 OVSF Code Tree 的深度為 9，使用者的請求 (Call Request) 以蒲松程序(Poisson Process)產生，其平均產生率(Mean Arrival Rate)為 λ。Codeword 每次被使用的時間則假設為指數分佈(Exponential Distribution)的隨機值且其平均值為 $1/\mu$。q 值定義為 λ/μ。每個請求 Req (r, k)中的 r 是一個隨機變數，且其值均勻分佈(Uniformly Distribution)於 1 至 128 間的整數。k 的值則與 r 值相關。當 $1 \le r \le 64$，k 值均勻分佈於 1 至 4 間；當 $64 \le r \le 128$，k 值均勻分佈在 2 至 4 間。在實驗中，我們每次將產生 50000 次請求 (Requests)，分別對所有的演算法進行測試。我們量測的數據有：

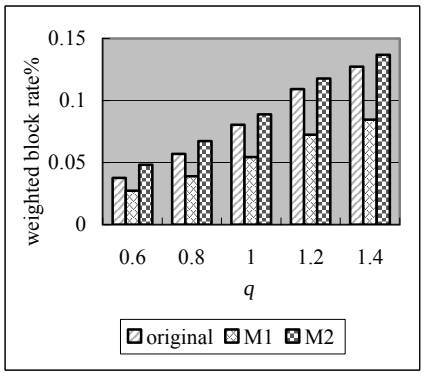
- 阻斷次數百分比 (Blocking Rate %) = 阻斷次數 / 頻寬要求次數 * 100% ，即阻斷率。其中阻斷次數為頻寬不足、碼碎裂與強制阻斷次數的總合。
- 阻斷頻寬百分比 (Weighted Blocking Rate %) = 阻斷總頻寬 / 要求總頻寬 * 100%。其中阻斷頻寬為頻寬不足、碼碎裂與強制阻斷頻寬的總合。
- 重配置所需的換碼次數 (Number of code replacements)

4.2 延後分配效能分析

我們以 TY 的演算法為基礎，加入延後分配機制而成為 M1 與 M2。圖六與圖七顯示 TY、M1 與 M2 的阻斷次數百分比與阻斷頻寬百分比的測量結果。其中圖六採用擁擠優先單碼配置演算法，而圖七採最左優先單碼配置。圖中右邊 Y 軸為阻斷次數百分比，以長條圖表示其數值，左邊 Y 軸為阻斷頻寬百分比，以折線圖表示其數值。我們可以看到 M1 在阻斷次數與阻斷頻寬百分比都有最好的效果；M2 在阻斷率雖較低於 TY，但其阻斷頻寬多於 TY。另外，擁擠優先配置演算法所得之阻斷次數與阻斷頻寬百分比皆低於最左優先配置演算法。圖八顯示 TY、M1 與 M2 重配置所需的換碼次數，我們可以看到 M1 搬動碼的數目最多。另外，擁擠優先配置演算法所得之換碼次數也低於最左優先配置演算法。我們也試著修改 SLC 及 CL，以加入延後分配機制。實驗結果與修改 TY 所得類似，在此我們就不重複其結果。

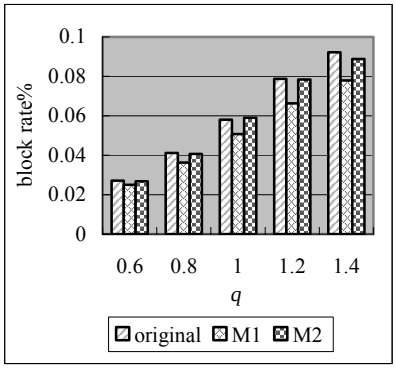


(a)

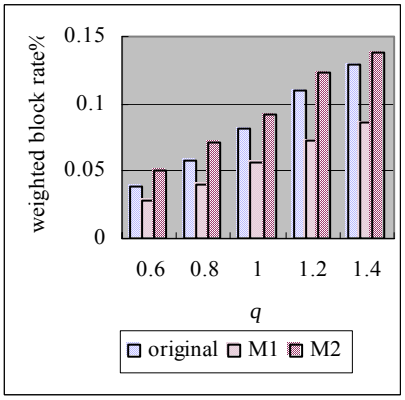


(b)

圖六：TY 使用擁擠優先配置演算法所得之(a)阻斷次數百分比與(b)阻斷頻寬百分比

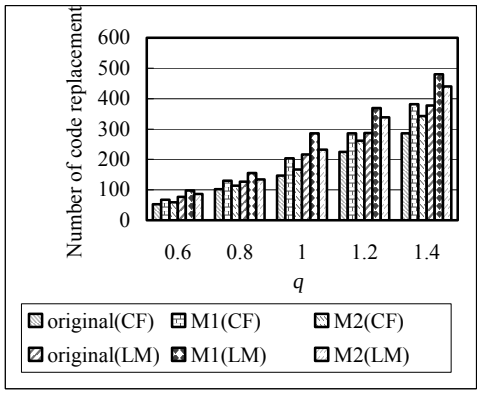


(a)



(b)

圖七：TY 使用最左優先配置演算法所得之(a)阻斷次數百分比與(b)阻斷頻寬百分比



圖八：TY 所需搬動碼的數目

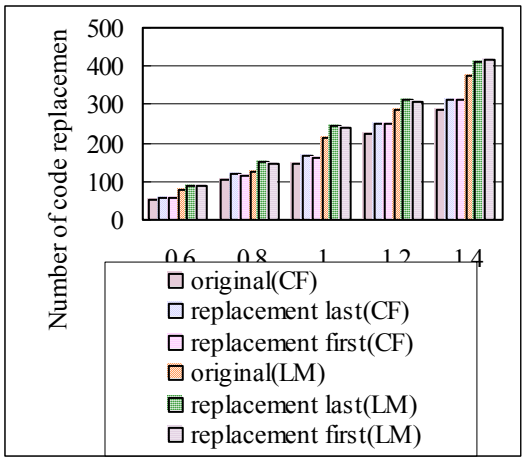
4.3 重配置處理效能分析

通道碼配置順序與換碼次數之關係

使用單碼配置演算法配置多碼中的通道碼時，可能會因碼碎裂現象而導致其中某些通道碼需進行重配置動作。TY 並不考慮此種狀況，直接依通道碼的展頻因子由小到大依序進行配置。在此我們考慮其他兩種可能性，包括優先處理需要進行重配置動作的通道碼 (Replacement First) 及最後才處理需要進行重配置動作的通道碼 (Replacement Last)。舉例來說，假設現有一待配置多碼，包含 1 個 SF=64，2 個 SF=128，與 1 個 SF=256 共四個通道碼，且系統現有 1 個 SF=64，1 個 SF=128，與 3 個 SF=256 的未配置通道碼。TY 的處理順序為先配置一個 SF=64 的通道碼，系統剩下 1 個 SF=128 與 3 個 SF=256 的通道碼，接下來欲配置兩個 SF=128 的通道碼時，因系統只剩下 1 個 SF=128 的通道碼，勢必要進行重配置動作。TY 的作法是展頻因子由小排到大，依照展頻因子由小到大的慢慢配置，如碰到需要重配置的就直接重配置，重配置完以後再處理接下來較大的展頻因子。Replacement First 的作法是先嘗試計算出哪些等下將需要重配置通道碼，則我們先將這些進行重配置，之後再依展頻因子由小到大依序處理剩餘的通道碼，因此 SF=128 的通道碼會被優先處理。Replacement Last 的作法則是依展頻因子由小到大依序處理，遇到需重配置的通道碼時（此例為 SF=128 的通道碼）先跳過，最後再處理。

我們針對這幾種可能性進行了實驗分析，結果顯示還是 TY 原先的作法換碼次數為最少，其次為 Replacement First，最後才是 Replacement Last，我們發現無論是先重配置或是最後才配置都沒有直接重配置來的好。Replacement First 的做法可能會因為要重配置時會動到其它通道碼的空間，所以可能造成其

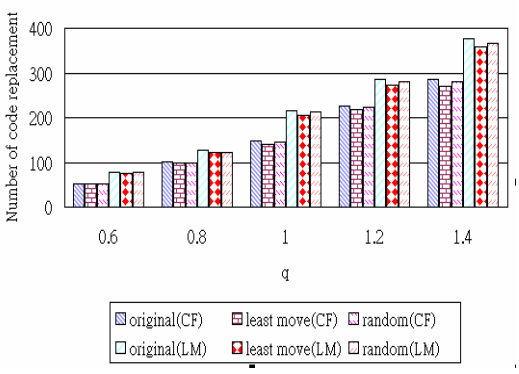
它原本不需要重配置的最後也要重配置，而 Replacement Last 的做法可能會因為有很多要重配置的通道碼，先重配置了最小的 SF 而換了一些碼，會因為換了這個碼讓第二小的 SF 會要換更多的碼，也就是之前換過的碼可能又再被換了一次，這樣連鎖下去會產生更多的換碼動作，故先前 TY 做法乃是取中庸作法，不先處理也不最後處理，而在重配置時考慮了會造成系統最小碼碎裂情況，也比較不會發生現在被重配置的碼等下還要再被重配置的可能。圖九為 TY 的實驗數據，在圖九中我們可以發現 TY 的作法是找出一個。以 SLC 與 LC 所做的實驗，也得到與 TY 相同的結論。



圖九：TY 的碼重配置處理順序與換碼次數的關係

多個候選通道碼與其換碼次數之分析

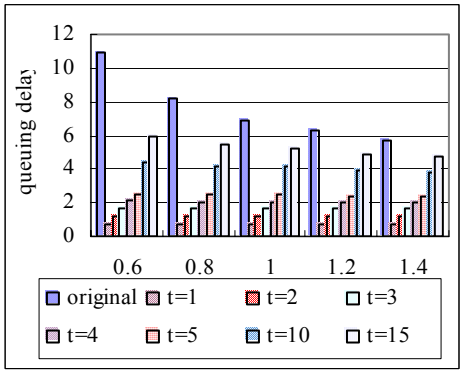
TY 的作法在有多個多碼候選者時，會選擇造成系統最小碼碎裂的候選者。此種作法經實驗證明可有效減少系統碼碎裂情形，進而降低阻斷率。不過當配合重配置機制一同運作時，因重配置機制可完全解決碼碎裂問題，挑選造成系統最小碼碎裂的多碼候選者即變得多此一舉。我們在此嘗試其他兩種可能作法，一是直接選取換碼次數 (Least Move) 最少的候選者，二是隨機選取任一候選者。換碼次數最少的乃是我們找出滿足特定請求 Req(r, k) 的所有頻寬組合，再從這些頻寬組合中挑出換碼次數最少者。而隨機選取是在候選者中任意挑出一個來做分配，並無其它挑選依據。圖十顯示實驗所得這三種作法的換碼次數結果，因為先前 (TY) 的作法並沒有考慮將來換碼是否有最少的換碼次數，而是考慮造成最小碼碎裂結果，故在換碼次數上不如直接選取換碼次數較少的。實驗結果證實 Least Move 策略比起原先作法換碼次數減少 2% 至 5%。



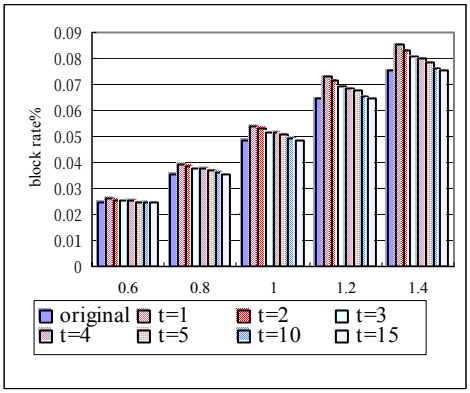
圖十：TY 的換碼次數選擇

4.4 限制延遲之效能分析

方法一的缺點為請求暫存在 Queue 中的等待時間並無限制，結果可能造成等待時間過長。在此我們以實驗方式探討限制等待機制的效能表現。此機制限制每個請求之最長等待時間為 t，當等待結果超過 t 時，即輸出目前暫存之結果。圖十一顯示實驗結果，我們可以看到原先方法一的 Queuing delay 介於 11 至 5。隨著 t 值遞減，Queuing delay 也愈來愈小。不過相對的，阻斷率也會隨之上升 (圖十二) 。



圖十一：方法一限制延遲之效能



圖十二：方法一限制延遲之阻斷次數百分比

五、結論

於 OVSF 碼的管理議題中，最主要的目標

皆在於減少使用者需求被阻斷的次數。阻斷發生的原因可能是系統頻寬不足，也可能是碼碎裂。為了降低系統頻寬不足所造成的阻斷次數，我們提出了通道碼延時分配機制，以不改變要求順序為前提，透過向前觀察的策略，選擇阻斷率較低的碼分配結果。實驗顯示此機制配合任何多碼分配演算法，均可再降低阻斷率約 0.5% 至 1.5%。不過我們的策略也犧牲掉了高傳輸率請求的服務機會。為了降低碼碎裂所造成的阻斷次數，我們可以採用重配置機制。先前的作法在有多個多碼候選者時，會選擇造成系統最小碼碎裂的候選者。我們提出配合重配置機制一同運作時，直接挑選換碼次數最少的多碼候選者。實驗結果證實此種策略可將換碼次數減少 2% 至 5%。另外，於配置分配到的多碼時，可能會因碼碎裂而導致其中某些通道碼需進行重配置動作。先前的作法並不考慮此種狀況，而直接依通道碼的展頻因子由小到大進行配置。此篇論文中，我們考慮了其他的可能性，包括優先處理需要進行重配置動作的通道碼及最後才處理需要進行重配置動作的通道碼。我們針對這幾種可能性進行了實驗分析，結果顯示還是原先的作法其換碼次數為最少。

參考文獻

[1] 3GPP TS 25.213, v4.2.0, Spreading and modulation (FDD), Dec. 2001.

[2] F. Acachi, K. Ohno, A. Higashi, T. Dohi and Y. Okumura. “Coherent Multicode DS-CDMA Mobile Radio Access,” *IEICE Trans. Commun.*, vol. E79-B, pp.1316-1325, Sept. 1996.

[3] F. Adachi, M. Sawahashi and K. Okawa.“Tree-structured Generation of Orthogonal Spreading Codes with Different Length for Forward Link of DS-CDMA Mobile Radio,” *Electronic Letter*, vol.. 22, pp. 27-28, Jan. 1997.

[4] Wen-Tsuen Chen, Hung-Chang Hsiao and Ya-Ping Wu. “A novel code assignment scheme for W-CDMA systems,” *IEEE VTC 2001*, vol. 2, pp. 1182-1186, 2001.

[5] Ray-Guang Cheng and Phone Lin. “OVSF Code Channel Assignment for IMT-2000,” in *Prof. of IEEE VTC*, vol. 3, pp. 2188-2192, May 2000.

[6] E.Dahlman, B.Gudmundson, M.Nisson, and J.Skold. “UMTS/IMT2000 based on wideband CDMA,” *IEEE Communications Magazine*, vol. 36, pp.70-80, Sep. 1998.

[7] Erik Dahlman and Karim Jamal.“Wide-band Services in DS-CDMA Based FPLMTS System,” in *Prof. of IEEE Vehicular Technology Conference*, vol. 3, pp. 1656-1660, 1996.

[8] C. L. I and R. D. Gitlin. “Multi-code CDMA Wireless Personal Communications Networks,” in *Proc. of IEEE International Conference on Communications*, vol. 2, pp. 1060-1064, 1999.

[9] T. Minn and K.-Y. Siu. “Dynamic Assignment of Orthogonal Variable-Spreading-Factor Codes in W-CDMA,” *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 8, pp.1429-1440, August 2000.

[10] Angelos N. Rouskas, and Dimitrios N. Skoutas. “OVSF codes assignment and reassignment at the forward link of W-CDMA 3G systems,” *Proc. of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 5, Sep. 15-18, 2002.

[11] H. K. Seok, S.H. Shin and K.S. Kwak. “A Hybrid Multi-rate Scheme for WCDMA,” in *Proceedings of TENCON’99*, vol. 2, pp. 1224-1227, Dec. 1999.

[12] Fenfen Shueh, and Wen-Shyen Chen. “Code assignment for IMT-2000 on forward radio link,” in *Proc. of. IEEE VTC*, vol. 2, May 2001.

[13] Fenfen Suneh, Zu-Eu Purple Liu and Wen-Shyen Chen. “A Fair, Efficient, and Exchangeable Channelization Code Assignment Scheme for IMT-2000,” in *Proc. of IEEE International Conference on Personal Wireless Communications*, pp. 429-433, Dec. 2000.

[14] Yu-Chee Tseng and Chih-Min Chao. “Code Placement and Replacement Strategies for Wideband CDMA OVSF Code Tree Management,” *IEEE GlobeCom*, 2001.

[15] Yang Yang and Tak-Shing P. Yum. “Nonrearrangeable compact assignment of orthogonal variable spreading factor codes for multi-rate traffic，” in *Prof. of IEEE VTC*, vol. 2, Oct. 2001.

[16] Li-Hsing Yen and Ming-Chun Tsou. “An OVSF Code Assignment Scheme Utilizing Multiple Rake Combiners for W-CDMA,” in *Proc. of IEEE ICC*, vol. 5, pp. 3312-3316, May 2003.