有效率的多維度分類規則衝突偵測演算法

林冠宇[a*] 李春良[b] 陳耀宗[a]
[a]國立交通大學資工程學系
[b]長庚大學資訊工程學系
[*]guanyu@cs.nctu.edu.tw

摘要

為滿足進階網路服務的需求，網路中的路由器必須具備封包分類(Packet Classification)的功能。相關的研究議題包含兩個；加速封包分類的速度以及快速偵測分類規則的衝突。若封包同時符合兩個以上分類規則(Filter)，且分類規則所設定的動作不同時，我們稱之為分類規則衝突(Filter Conflict)。分類規則衝突可能造成封包錯誤的分類，進而影響網路安全、資料傳輸的正確性等問題。近年來許多文獻所提出衝突偵測演算法，大多只針對分類規則前兩維度(來源端以及目的地端位址)的內容來進行衝突偵測，因為不同欄位含有不同的資料特性，現有演算法不是無法直接應用於多維度分類規則，就是無法有效率地運作。因此本論文將針對分類規則其他欄位的內容，提出一個有效率的多維度分類規則衝突偵測演算法，並且與目前其他演算法比較，能夠在更快的時間內偵測所有分類規則的衝突。

關鍵詞：封包分類、分類規則、分類規則衝突。

1. 前言

封包分類器(Packet Classifier)位在網際網路上的路由器中，封包分類器透過所接受到封包的表頭(Header)資訊，並查詢在封包分類器上預先定義的規則表格(Predefined Rule Table)中所定義的規則，執行對應的動作，規則表格則包含在分類器上的分類規則資料庫(Rule Database)中。常見的分類欄位有來源端位址、目的地端位址、來源端通訊埠(Source Port)、目的地端通訊埠(Destination Port)或是通訊協定(Protocol)。如果一個分類規則 F 含有 d 個欄位，則我們稱 F 為一個 d 維的分類規則，表示為 $F = (f[1], f[2], ..., f[d])$ ，其中任意第 i 個欄位 $f[i]$ 的資料內容可以是變動長度的前置字元(Prefix)、準確值(Exact Value)、數值範圍(Range)或是通訊協定中用來表示某種特定的值，例如：TCP、UDP、ICMP 或其它等。當分類規則中某一個欄位用*表示時，則視為泛用規則(Wildcard Rule)，代表該欄位的值包含所有的範圍。對於一個封包 P 與某一個分類規則 F 而言，若封包的表頭內容與比對的欄位資訊皆符合 F，我們稱之為封包 P 符合分類規則 F。

舉例來說，表 1 為一個分類器內含有四個分類規則的範例，分類規則 a 的各個欄位意義如下：

- Source Address (SA): 來源端位址的前置位元為*，若來源端位址的欄位內容介於 0.0.0.0~255.255.255.255 時，符合分類規則 a。
- Destination Address (DA): 目的端位址的前置字元為 163.25.*.*，若目的端位址的欄位內容介於 163.25.0.0~163.25.255.255 時，符合分類規則 a。
- Source Port (SP): 若來源端通訊埠的值介於 0 到 65535，符合分類規則 a。
- Destination Port (DP): 若目的端通訊埠的值介於 23 到 80，符合分類規則 a。
- Protocol Type (Protocol): 通訊協定為 TCP 時，符合分類規則 a。

每一分類規則會定義其對應的動作(Action)，舉例來說，若分類規則 a 定義的動作為 Accept 時，表示封包若符合分類規則 a 內容時，則接受該封包。

表 1 具有五個欄位內容的分類規則之範例

F	SA	DA	SP	DP	PROT
a	*	163.25.*.*	*	23-80	TCP
b	140.113.*.*	163.25.*.*	*	*	*
c	*	140.113.215.*	*	0-1023	UDP
d	140.113.1.1	*	*	1024-65536	UDP

若此時有一表頭內容為(140.113.1.1, 163.25.1.1, 1000, 23, TCP)的封包 P 到達路由器時，由表 1 可知封包 P 同時符合分類規則 a 和分類規則 b 所定義的欄位內容。如果兩分類規則所設定的動作是彼此衝突的(假設分類規則 b 定義的動作為 reject 時)，這種情況下分類器無法精確判斷封包 P 該執行哪一分類規則的動作，這使得類似防火牆的服務將會有潛在的安全漏洞，而資料在傳輸上也有可能發生傳輸錯誤的問題[1]-[2]。所以我們需要對分類規則資料庫進行衝突偵測的處理。

以目前網際網路使用的趨勢而言，原本的網際網路通訊協定第四版(IPv4)所能提供的位址已不敷使用，透過網際網路通訊協定第六版(IPv6)的制定可以解決位址不夠用的情況，再加上多元化的網路服務如雨後春筍般的出現。未來網路傳送的速度和資料量勢必會不斷地增加。對於路由器上的分類規則資料庫來說，分類規則個數除了會更加的龐大之外，為滿足這些網路服務，分類規則動態的新增、

刪除也會更加頻繁。因此，除了必須要加速封包分類的執行之外，分類規則的衝突偵測在面對分類規則動態新增或刪除時也必須能有效率的運作，尤其是每新增一個分類規則時，要能快速找出分類規則資料庫中所有會產生衝突的分類規則集合，否則這些問題將會成為路由器效能上的瓶頸。

2. 文獻探討

最早由Hari, Suri以及Parulkar提出分類規則衝突的問題[1]，定義出分類規則衝突的關聯式。存在兩個分類規則 f 與分類規則 g，假設分類規則含有 $d(d\geq 2)$個欄位，若以下條件成立時，則 f 與 g 互相衝突：

$f[i]$表示的範圍被 $g[i]$所包含且 $g[j]$表示的範圍被 $f[j]$所包含，其中 $1\leq i\leq d$、$1\leq j\leq d$ 且 $i\neq j$。

從表 1 中我們可以看出分類規則 a 欄位 DP 所表示的範圍被分類規則 b 欄位 DP 所包含，而分類規則 b 欄位 SA 所表示的範圍被分類規則 a 欄位 SA 所包含。因此根據上述條件判定分類規則 a 與分類規則 b 互相衝突。

由於衝突偵測的技術能保證封包分類演算法在執行上的正確性，因此現能找到許多與衝突偵測演算法相關的論文。而這些演算法大致上分為兩類：其一在現有的封包分類演算法上新增一些資料結構使得該演算法同時具有衝突偵測以及封包分類兩種功能[1]-[4]；另一類則是建立新的資料結構來執行分類規則衝突的偵測[5]-[6]。但上述文獻大多只著重前兩維度(來源端/目的地端位址)資訊的內容來進行衝突偵測。為了能更貼近實際的運用則必須要納入多維度的分類規則衝突偵測。但[5]-[6] 沒有提出針對多維度衝突偵測的作法，而所建立的資料結構亦無法支援多維度的偵測。[1]-[2]雖提出簡易的多維度衝突偵測方法，但複製大量分類規則的做法將大幅增加記憶體空間的使用量。[4]則是利用[3]的基礎提出多維度的衝突偵測，由於方法較為簡易，所需偵測的時間也較多。

考慮分類規則具有五個欄位的資訊，本篇論文將針對後三欄位(來源端通訊埠、目的地端通訊埠以及通訊協定)不同的資料特性，提出一個有效率的衝突偵測方法。此方法最大的優點在於可結合既有的二維度衝突偵測演算法，進行高效率的多維度分類規則衝突偵測。

3. 後三維度分類規則衝突偵測之方法

3.1 資料特性

相較於來源端/目的地端位址內容是以前置字元的資料型態儲存，來源端/目的地端通訊埠內容通常是以數值範圍或準確值的資料型態表示。當資料內容為一個準確值時，我們可以用一個前置字元來表示，假設字元長度為 4 時，可以用 1101 來表示 13 這個值；當要用前置字元來表示 1-14 這個數值範圍的時候，卻需要 0001(1)、001*(2-3)、01*(4-7)、10*(8-11)、110*(12-13)以及 1110(14)等六個前置字元集合(Prefix Set)才能表示其完整的範圍。換句話說，一個數值範圍往往需要多個前置字元才能完整表示。分析上述文獻所提出的方法，若資料結構是以前置字元的型態來建立[1]-[2]，針對來源端/目的地端通訊埠的欄位內容為數值範圍時通常有兩種做法，一是假設產生的分類規則其欄位儲存內容都可以用一個前置字元表示；二是將其儲存內容轉換成多個前置字元表示的集合。以前面的例子來說，拆解成六個前置字元之後，分類規則也必須對應複製成六個分類規則儲存至對應的資料結構中。隨著分類規則個數變多，因為分類規則經過先拆解後複製的步驟將會使得分類規則個數呈倍數成長，再加上通訊協定欄位碰到泛用規則時也同樣使用先拆解後複製的方法時，不但會造成記憶體空間使用上的瓶頸，分類規則個數龐大連帶影響執行衝突偵測的時間。因此將分類規則經拆解而複製的作法並不能有效地解決問題。

由於數值範圍的資料特性，轉換為前置字元集合的方式會引起大量複製分類規則的結果，為了避免這情況發生，現階段的文獻在處理後三維度衝突偵測的方法大都是將各欄位的資料取出之後再詳細比對，來判斷是否產生衝突。但每一次的詳細比對都會耗費多次記憶體存取(Memory Accesses)的動作，因此當分類規則資料庫中的分類規則個數增加時，偵測衝突後三維度所需要的記憶體存取次數也會倍增。所以，在比對後三維度是否有衝突發生時，如果能夠避免每次都要做詳細比對的動作，不但可以減少因讀取詳細資訊所大量耗費的記憶體存取次數，也能減少整體衝突偵測所需要的時間。

根據[7]中的定義，來源端/目的地端通訊埠欄位可以分成下列五個資料表示的形式：

1. WC: 泛用規則。
2. HI: 短暫的用戶端通訊埠範圍，介於 1024 至 65535 之間。
3. LO: 較著名的系統端通訊埠範圍，介於 0 至 1023 之間。
4. AR: 表示的是一個任意的範圍(Arbitrary range)。
5. EM: 表示的是一個精確值。

而通訊協定欄位可以分成下列數個資料表示的形式：

1. WC: 泛用規則。
2. TCP: 傳輸控制協議。
3. UDP: 用戶端資料報協議。
4. ICMP、GRE、OSPF、IGP、EIGRP、ESP、AH、IPE 等其他 8 個通訊協定。

我們可以發現 HI 和 LO 所涵蓋的範圍是不重疊的，當兩個欲比對的分類規則，在同一通訊埠欄位其內容若分別屬於 HI 和 LO 的形式時，我們可以稱它們的關係為互斥的(Mutually Exclusive)。如表 1

所示，因為分類規則 *c* 和分類規則 *d* 在欄位 DP 的內容是互斥的，根據[1]對於分類規則衝突的定義，我們不能找到一個封包 P 的表頭內容可以同時符合分類規則 *c* 和分類規則 *d*，因此這兩分類規則不具有衝突的關係。同樣地，通訊協定的欄位除了泛用規則外，不同的通訊協定也是彼此互斥的，例如我們不能找到一個封包可以同時符合一個通訊協定欄位設定為 TCP 且另外一個通訊協定欄位設定為 UDP 的兩分類規則。透過這些觀察可以得到一個結論，任兩個分類規則在判斷是否發生衝突時，只要其中一個欄位資訊比對的結果為互斥時，該兩分類規則必定不會發生衝突。

3.2 進行編碼

由於 HI 和 LO 兩個集合具有互斥的關係，我們可預先將通訊埠欄位內容為 AR 或 EM 的分類規則，根據欄位資料所涵蓋的區間預先分成 HI 以及 LO 兩種集合的表示方式，例如欄位資訊為 EM 且值為 23 時，就可以將其分類至 LO 的集合中；欄位資訊為 AR 且範圍值為 2000-5000 就可以分類至 HI 的集合中。其中要注意的是，因為 AR 所表示的範圍有可能橫跨 HI 和 LO 兩個區間，例如範圍值為 20-5000。所以我們將這類的資料內容分至一個新的集合中，並命名為 Both 集合。最後，通訊埠欄位內容的表示形式我們可以分成四個類別：

1. WC: 泛用規則。
2. HI: 此類別包含三個集合：短暫的用戶端通訊埠範圍、精確值以及表示範圍介於 1024-65535 之間的數值範圍。
3. LO: 此類別包含三個集合：較著名的系統端通訊埠範圍、精確值以及表示範圍皆介於 0 至 1023 之間的數值範圍。
4. Both: 若數值範圍涵蓋的範圍橫跨 HI 和 LO 兩個區間，則歸類在此範圍內。

根據[7]的統計，分類規則中通訊協定欄位較常出現的內容為 TCP、UDP 以及泛用規則，其餘通訊協定的出現占非常小的比例，我們可視其為一個集合，並命名為 Other 集合。因此通訊協定欄位亦分成 WC、TCP、UDP、Other 四個類別。其中 TCP、UDP、Other 這三個集合相互具有互斥的關係。

在比對的兩分類規則時，即使其來源端/目的地端通訊埠的欄位內容皆屬於 LO 或是 HI 的集合時，也有可能存在互斥的關係。例如兩來源端通訊埠的內容分為精確值 23 及 80，雖然座落區間皆屬於 LO 集合，但內容是彼此互斥的。所以比對欄位同為 LO 或是 HI 集合時，則需要做進一步讀取實際的資料內容來做精確的比對，以確定是否有互斥的發生。同樣的通訊協定欄位內容為 Other 時，也必須做進一步精確的比對才確定是否有互斥的發生。但欲比較的兩個欄位其中一個內容為 WC 時，我們可以確定該欄位資料彼此必定存在重疊(Overlap)的關係，則不需要做精確比對來判斷是否有互斥發生。因此當每一個欲比對的欄位皆至少存在一個內容為 WC 時，我們可以確定兩分類規則不會有互斥發生之外，也不需要做進一步詳細比對。

經過上述的分類與探討，每一個欄位比對結果會發生下列三種情況；(1)互斥、(2)需要精確比對，(3)不需要精確比對。當單一欄位衝突偵測拓展至三維度衝突偵測時，我們重新定義這三種比對結果；

1. 互斥：任何一個欄位比對結果含有互斥的關係時，其比對結果必為互斥。
2. 需要精確比對：當三個欄位關係皆無互斥的情況下，來源端或目的地端通訊埠欄位內容同屬於 HI 或 LO 集合時，或者通訊協定欄位皆為 Other 時，需要進行詳細比對。
3. 不需要精確比對：排除上述兩項的情況皆屬於不需要精確比對的結果。

我們利用表 2 來說明實際的例子；分類規則 *a* 和分類規則 *b* 在欄位 DP 和欄位 Protocol 的關係皆是互斥的情況，因此兩分類規則無衝突的關係。分類規則 *c* 和分類規則 *d* 在欄位 DP 的內容分別是 LO 和 Both，需要做進一步精確比對才能判斷是否有無衝突的可能。分類規則 *e* 和分類規則 *f* 則是因為三個欄位中皆至少有一個 WC 的內容，因此不需要精確比對。

表 2 後三維度比對的情況

範例	Filter	SP	DP	Protocol	結果
(1)	*a*	HI	LO	Other	互斥
	b	WC	HI	TCP	
(2)	*c*	WC	LO	TCP	需要精確比對
	d	WC	Both	WC	
(3)	*e*	WC	WC	TCP	不須精確比對
	f	WC	LO	WC	

根據上述定義的三種結果以及展示的範例可以發現，將通訊埠以及通訊協定的資訊內容做分類，再根據分類的項目我們可以預先知道每一組欄位內容比對的情況，當比對結果為互斥和不需精確比對時，就不需經過詳細的比對就可以判定兩個欲比對的分類規則是否發生衝突。透過我們所設計的前置資訊，可以篩選不需要經過詳細比對的分類規則對，藉此減少因讀取詳細資訊以及因詳細比對所造成大量耗費記憶體存取的次數，進而加速整體偵測衝突的時間，而篩選的比例越高，可以加速的時間也就越多。為了實作後三維衝突偵測的方法，我們針對通訊埠和通訊協定所分類的結果作對應的編碼，如表 3 所示：

表 3 根據分類結果所對應的編碼表

通訊埠	對應編碼
WC	00
HI	01
LO	10
Both	11

通訊協定	對應編碼
WC	00
TCP	01
UDP	10
Other	11

3.3 建立查詢表

將後三維度的欄位資訊經過對應編碼後，每一個分類規則都含有一個 6 位元長度的編碼串列(Bit Stream)。例如表 2 範例(1)中的分類規則 *a*，三個欄位的內容分別是 HI、LO 以及 Other，因此對應的編碼為 011011；而分類規則 *b* 的編碼為 000101。從分類規則資料庫讀取每一筆分類規則資訊時，同時新增並儲存對應的編碼串列於該分類規則中。根據前小節的分析，欄位資訊比對的結果可以分為三種，因此我們針對三種結果來設定對應值，如表 4 所示。

透過列舉每一種配對的結果，我們可以建立一個查詢表。將兩分類規則的編碼串列合併作為 12 位元長度的索引值(index value)，每一項索引值的內容代表兩分類規則其後三維度欄位的資訊內容，再根據 **3.2** 節所定義的比對結果，設定該項的比對結果對應值。在執行後三維度衝突偵測時，根據比對的結果來決定是否需要做詳細比對的動作，比對查詢表的範例如表 5 所示。例如比對的兩分類規則在前二維衝突偵測有發生衝突時，經過索引查詢後發現後三維度的關係為互斥時，就不須再詳細比對。

表 4 比對結果對應值

比對結果	對應值
互斥	00
不須精確比對	01
需要精確比對	10

表 5 利用對應編碼為索引值、配對結果為對應值所建立的查詢表

對應鍵值 (Filter *f*, Filter *g*)		比對結果對應值
(000000, 000000)		01
(000000, 000001)		01
(000000, 000010)		01
⋮		⋮
(000000, 111111)	比對搜尋 →	01
⋮		⋮
(111111, 1111101)		00
(111111, 1111110)		00
(111111, 1111111)		10

當索引值為(000000, 000000)時，表示兩個分類規則其後三維度的欄位資訊皆為泛用規則，因此查表的結果為 01，即為不需精確比對；當對應鍵值為(111111, 1111101)時，表示前者分類規則後三維度的欄位資訊為(Both, Both, Other)，後者為(Both, Both, UDP)，可以發現在通訊協定欄位兩者的內容為互斥，因此查表的結果為 00，即互斥；當對應鍵值為(111111, 1111111)時，表示兩個分類規則其後三維度的欄位資訊皆為(Both, Both, Other)，必須作精確比對才能判斷兩分類規則是否發生衝突，所以得到的配對結果為 10。由於配對的結果是不變的，所以可以根據我們設定的編碼事先建立好該查詢表。而整體查詢表的大小為 2^{12}(索引值的個數)×2(配對結果所需要的位元數) = 2^{13} 位元，只需要相當於一千位元組(1KB)的大小。對於每一個分類規則我們則也只需要額外 6 位元的空間來儲存對應的編碼資訊。

經過上述所提出的方法，將後三維度的欄位資訊透過編碼的方式並建立對應的比對結果查詢表。兩分類規則在執行衝突比對時，讀取兩分類規則的編碼串列做為鍵值，並查詢比對的結果。就可以有效率地過濾封包分類在衝突偵測中，不需要經過詳細比對的流程，進而減少大量的記憶體存取次數，加速衝突偵測的時間。另外，我們演算法的設計也可以直接套用在已有的二維度分類規則衝突偵測演算法，輕易地擴展至多維度分類規則衝突偵測。

4. 模擬實驗結果分析

模擬實驗中所需的分類規則資料庫我們利用 Class-Bench[7]的分類規則產生器，Class-Bench 可以產生三種貼近真實的分類規則資料庫，分別是存取控制列表(Access Control Lists；ACL)、防火牆(Firewalls；FW)以及分類規則鏈(IP Chains；IPC)。各項參數設定皆使用[8]所提供的預設值，如表 6 所示。

表 6 模擬實驗參數表

參數名稱	設定值
Smoothing	2
Address scope	0.5
Port scope	-0.1
Size	1K, 5K, 10K, 20K, 30K

我們模擬實驗將針對 GOT[1]和 ABV[2]這兩種演算法，探討在後三維度衝突偵測上，比較其使用傳統的比對方式以及使用我們所提出的演算法在時間效能上的差異。模擬實驗結果如表 7、表 8 所示(由於資料庫 ACL5 衝突數少，使得偵測時間非常短，故精確值計算至小數點以下第二位)；表 7 和表 8 分別顯示在分類規則衝突數低以及分類規則衝突數高的情況下執行衝突偵測所需要的時間，執行的單位時間為毫秒(ms)。我們利用表 7 的內容做一些說明，欄位 GOT 代表原 GOT 演算法在執行後三維度衝突偵測時所需要的時間，而欄位 Proposed 代表用我們所提出演算法來取代原有的方法，在執行後三維度衝突偵測所需要的時間。相對的，表 8 中欄位 Proposed 即是代表用我們所提出的演算法來取代原有 ABV 在後三維度衝突偵測所使用的方法。從模擬實驗結果可以看出，不管在分類規則衝突對數多或少的情況下，我們所提出的演算法皆能明顯減少後三維度衝突偵測所需要的時間。

5. 結論

在本篇論文中，我們利用少量的記憶體空間，針對分類規則後三維度的欄位內容分類成數個集

合並且對其編碼。討論不同集合之間的關係，分析出每種組合的比對結果。最後，根據每種比對的結果建立對應的比對結果查詢表。當對分類規則後三維度進行衝突偵測時，藉由預先查表的過程可以過濾不需要精確比對的分類規則對，進而減少分類規則在衝突偵測上所需要的時間。此外我們的方法也能輕易的套用在現有的二維度的分類規則衝突偵測演算法，擴展至多維度的分類規則衝突偵測。

致謝

本研究承蒙國科會研究計畫(計畫編號 NSC 102-2221-E-182-034 以及 NSC 101-2221-E-182-074)之經費補助，特此感謝。

參考文獻

[1] A. Hari, S. Suri, and G. Parulkar, "Detecting and resolving packet filter conflicts," in INFOCOM, pp. 1203–1212, 2000.

[2] F. Baboescu and G. Varghese, "Fast and scalable conflict detection for packet classifier," Computer Networks, vol. 42, no. 6, pp. 717-735, 2003.

[3] Lee, C.-L., Lin, G.-Y., Chen, Y.-C., "An efficient conflict detection algorithm for packet filters," IEICE Transactions on Information and Systems, vol. E95.D, issue 2, pp. 472-479, 2012.

[4] Lee, C.-L., Lin, G.-Y., Chen, Y.-C., "Fast and memory efficient conflict detection for multidimensional packet filters," Advances in Intelligent and Soft Computing 145 AISC, vol. 2 , pp. 205-211, 2012.

[5] H. Lu and S. Sahni, "Conflict detection and resolution in two-dimensional prefix router tables," IEEE/ACM Transactions on Networking, vol. 13, no. 6, pp. 1353–1363, 2005.

[6] Maindorfer, C., Mohamed, K.A., Ottmann, T., Datta, A., "A new output-sensitive algorithm to detect and resolve conflicts in internet router tables," in INFOCOM, pp. 2431-2435, 2007.

[7] D.E. Taylor and J.S. Turner, "Classbench: a packet classification benchmark," IEEE/ACM Transactions on Networking, vol 15, no. 3, pp. 499–511, 2007.

表 7 衝突對數少之時間比較(ms)

資料庫種類	ACL5			FW4			IPC1		
大小	GOT	Proposed	衝突對數	GOT	Proposed	衝突對數	GOT	Proposed	衝突對數
1K	0.05	0.04	0	5.4	3.9	16,678	0.7	0.5	5,692
5K	0.21	0.19	2	160.2	114.2	517,766	15.2	11.9	110,457
10K	0.62	0.51	6	1,209.6	989.8	2,099,778	66.5	52.6	462,587
20K	0.82	0.69	3	4,050.5	3,465.7	7,871,958	252.7	201.3	1,867,448
30K	1.66	1.01	8	9,164.6	7,722.4	18,148,187	689.8	57.5	4,470,090
大小	ABV	Proposed	衝突對數	ABV	Proposed	衝突對數	ABV	Proposed	衝突對數
1K	2.0	0.8	0	2.7	1.2	16,678	2.1	0.7	5,692
5K	29.4	17.8	2	42.8	27.7	517,766	32.8	19.9	110,457
10K	100.2	70.2	6	146.6	106.0	2,099,778	106.9	77.3	462,587
20K	378.0	281.9	3	518.6	408.3	7,871,958	393.3	302.2	1,867,448
30K	834.9	640.1	8	1,141.1	926.5	18,148,187	894.9	695.3	4,470,090

表 8 衝突對數多之時間比較(ms)

資料庫種類	ACL2			FW2			IPC2		
大小	GOT	Proposed	衝突對數	GOT	Proposed	衝突對數	GOT	Proposed	衝突對數
1K	1.1	0.7	9,017	4.6	3.1	40,161	2.7	2.2	29,208
5K	22.5	14.5	153,524	122.6	102.8	1,051,308	164.4	93.3	732,267
10K	85.8	59.9	649,795	580.0	471.1	4,588,158	546.2	475.9	3,035,893
20K	341.9	218.1	2,677,799	3,120.7	2,233.8	18,407,548	2,166.2	1,657.8	12,510,032
30K	642.8	403.8	4,659,061	11,188.7	5,573.8	39,029,636	5,110.6	3,747.1	26,687,173
大小	ABV	Proposed	衝突對數	ABV	Proposed	衝突對數	ABV	Proposed	衝突對數
1K	2.5	0.8	9,017	2.1	0.9	40,161	1.8	0.8	29,208
5K	36.8	20.5	153,524	33.9	22.4	1,051,308	32.1	19.8	732,267
10K	129.9	78.0	649,795	117.1	86.7	4,588,158	119.5	81.1	3,035,893
20K	462.0	310.0	2,677,799	437.5	361.7	18,407,548	394.5	339.4	12,510,032
30K	969.2	673.3	4,659,061	960.7	804.9	39,029,636	862.2	735.1	26,687,173