I527398

(19)中華民國智慧財產局

(12)發明說明書公告本 **(11)證書號數：TW I527398 B**

(45)公告日： 中華民國 105 (2016) 年 03 月 21 日

(21)申請案號：103109415 (22)申請日： 中華民國 103 (2014) 年 03 月 14 日

(51)Int. Cl.： *H04L1/00 (2006.01)* *H04L1/20 (2006.01)*
H04L29/14 (2006.01)

(30)優先權：2013/03/15 美國 61/786,594
2014/03/12 美國 14/206,435

(71)申請人：廣達電腦股份有限公司 (中華民國) QUANTA COMPUTER INC. (TW)
桃園市龜山區文化二路 188 號

(72)發明人：弗里戈 馬泰奧 FRIGO, MATTEO (IT)；史都華德 勞倫斯 科爾姆 STEWART, LAWRENCE COLM (US)

(74)代理人：蔡濱陽

(56)參考文獻：

TW	200637168	CN	101621353A
US	6023783B	US	2002/0034261A1
US	2010/0095189A1	US	2011/0258514A1

審查人員：陳怡婷

申請專利範圍項數：22 項 圖式數：18 共 36 頁

(54)名稱

錯誤修正碼

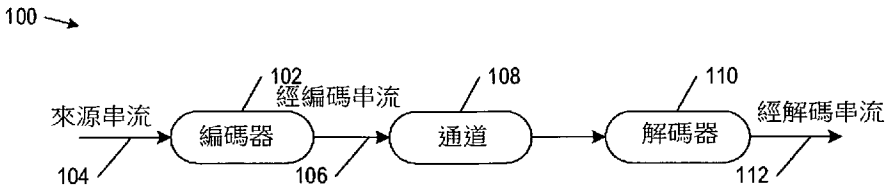
ERROR-CORRECTING CODE

(57)摘要

一種提供用於傳送一錯誤修正碼的方法。該方法包括接收一來源封包、形成包括該來源封包的一長度與該來源封包的一擴增封包、在形成該擴增封包之後推進一滑動編碼器窗、藉由將該擴增封包填塞一些零(0's)來形成一填塞封包，所以該填塞封包與在該滑動編碼器窗中一些較舊的填塞封包具有相同長度。該方法另包括產生編碼的封包，其係藉由當該填塞封包第一次被編碼時單獨編碼該填塞封包，而當該填塞封包在第一次之後被編碼幾次時，利用在該滑動窗中較舊的填塞封包來編碼該填塞封包。

A method is provided to transmit an error-correcting code. The method includes receiving a source packet, forming an augmented packet including a length of the source packet and the source packet, advancing a sliding encoder window after forming the augmented packet, forming a padded packet by padding the augmented packet with a number of zeroes (0's) so the padded packet and a number of older padded packets in the sliding encoder window have a same length. The method further includes generating coded packets by, when the padded packet is encoded for a first time, encoding the padded packet alone, and, when the padded packet is encoded for a number of times after the first time, encoding the padded packet with older padded packets in the sliding window.

指定代表圖：



第一圖

符號簡單說明：

102・・・編碼器

104・・・來源串流

106・・・編碼串流

108・・・通道

110・・・解碼器

112・・・解碼串流

I527398

公告本

發明摘要

※ 申請案號：103094115

※ 申請日：103. 3. 14　　※IPC 分類：H04L 1/00 (2006.01)
H04L 1/20 (2006.01)
H04L 29/14 (2006.01)

【發明名稱】（中文/英文）

錯誤修正碼/ ERROR-CORRECTING CODE

【中文】

一種提供用於傳送一錯誤修正碼的方法。該方法包括接收一來源封包、形成包括該來源封包的一長度與該來源封包的一擴增封包、在形成該擴增封包之後推進一滑動編碼器窗、藉由將該擴增封包填塞一些零(0's)來形成一填塞封包，所以該填塞封包與在該滑動編碼器窗中一些較舊的填塞封包具有相同長度。該方法另包括產生編碼的封包，其係藉由當該填塞封包第一次被編碼時單獨編碼該填塞封包，而當該填塞封包在第一次之後被編碼幾次時，利用在該滑動窗中較舊的填塞封包來編碼該填塞封包。

【英文】

A method is provided to transmit an error-correcting code. The method includes receiving a source packet, forming an augmented packet including a length of the source packet and the source packet, advancing a sliding encoder window after forming the augmented packet, forming a padded packet by padding the augmented packet with a number of zeroes (0's) so the padded packet and a number of older padded packets in the sliding encoder window have a same length. The method further includes generating coded packets by, when the padded packet is encoded for a first time, encoding the padded packet alone, and, when the padded packet is encoded for a number of times after the first time, encoding the padded packet with older padded packets in the sliding window.

S

【代表圖】

【本案指定代表圖】：第（一）圖。

【本代表圖之符號簡單說明】：

102	編碼器	108	通道
104	來源串流	110	解碼器
106	編碼串流	112	解碼串流

【本案若有化學式時，請揭示最能顯示發明特徵的化學式】：

無

發明專利說明書

（本說明書格式、順序，請勿任意更動）

【發明名稱】（中文/英文）

錯誤修正碼/ ERROR-CORRECTING CODE

【背景技術】

【0001】 更正錯碼用於降低在雜訊通道上通信時的錯誤率。一種典型的更正錯碼傳送比必須的要更多的符號，使得收到一破壞的符號串流時，只要該破壞不是很大，仍允許原始消息被重構。

【0002】 當一更正錯碼在源符號數目對於所傳送的符號數目有一固定比例時，該碼具有一固定速率。當一更正錯碼允許速率在傳送期間改變時，該碼具有可變速率。例如，速率可適應於通信通道當中變化的條件。

【0003】 典型的可變速率碼在消息的一實質部份被接收的情況下具有一長的解碼延遲。雖然對於文件傳送這樣可能不會造成問題，但如果消息為諸如一視頻會議的消息的某種即時音頻或視頻串流時，此延遲即為不可取的。要避免此問題的一種方法為通過使用一滑動視窗，其中所傳送的符號為源消息的一連續性小片段的一函數(視窗)。視窗初始時涵蓋該消息的一首碼，且其隨著傳送進行而向前移動。

【圖式簡單說明】

【0004】 本發明的前述及其它特徵將可結合附圖從下面的描述及附屬權利要求中更加顯而易見。應瞭解到這些圖僅描述根據本發明的數個實施例，因此不應視為本發明範圍的限制，本發明將透過使用附圖來利用附加的特定性與細節加以描述。

【0005】 在圖中:

【0006】 圖 1 為本發明的示例中一種具有一編碼器與一解碼器的系統的方塊圖。

【0007】 圖 2 為例示本發明的示例中圖 1 的編碼器的操作的方塊圖。

【0008】 圖 3 和圖 4 為在本發明的示例中一種用於圖 1 的編碼器來傳送一更正錯碼的方法的流程圖。

【0009】 圖 5、圖 6、圖 7 和圖 8 演示在本發明的示例中圖 3 和圖 4 的編碼方法。

【0010】 圖 9 和圖 10 分別示出伽羅瓦域(GF, “Galois field”)(16)的加法與乘法表。

【0011】 圖 11 示出在本發明的示例中由圖 1 的解碼器所維持的一解碼矩陣。

【0012】 圖 12 和圖 13 為在本發明的示例中一種用於圖 1 的解碼器來解碼一更正錯碼的方法的流程圖。

【0013】 圖 14、圖 15、圖 16 以及圖 17 演示在本發明的示例中圖 12 和圖 13 的解碼方法。

【0014】 圖 18 為在本發明的示例中一種用於實現圖 1 的編碼器或解碼器的計算設備的方塊圖。

【實施方式】

【0015】 如本文所使用的，術語“包括(include)”代表包括但不限於之意，術語“包含(including)”代表包含但不限於之意。術語“一”(“a" 與” an”)意圖注明一特定元件中的至少一個。術語“基於(based on)”代表至少部份基於之意。術語“或”用於指代一非排除性，使得 A 或 B 包括“A 但無 B”，“B 但無 A”，以及“A 與 B”，除非另有指明。

【0016】 在本發明的示例中，提供了方法與裝置來生成並解碼一改錯碼。該碼可在網際網路協定(IP, “Internet Protocol”)上操作，所以其可在軟體、硬體或其組合中實現，而都不會改變既有的網際網路基礎設施。

【0017】 高階說明

【0018】 圖 1 為本發明的示例中一系統 100 的方塊圖。系統 100 包括一編碼器 102，其處理一來源串流(Source stream)104(例如來自連接到一視頻攝影機的一視頻編解碼器)以產生一經編碼串流(encoded stream)106。來源串流 104 可具有可變大小的封包的形式。如果來源串流 104 是純位元組串流，則編碼器 102 可選擇方便的封包大小。編碼器 102 經由一通道 108 傳送經編碼串流 106 到一解碼器 110，其返回應該等於來源串流 104 的一經解碼串流

112(例如到連接至一視頻匯流（sink）的一視頻編解碼器)。來源串流 104 包括一封包序列。來源串流 104 中的封包可有不同長度。 通道 108 可刪除、延遲、重新排列或複製經編碼串流 106 中的封包。通道 108 可以不遞送損壞的封包，其將由通道 108 中的錯誤檢測機制檢測到並刪除，所述錯誤檢測機制諸如封包校驗和或迴圈冗餘校驗（CRC）碼。

【0019】 編碼器 102 與解碼器 110 二者事先都同意使用一有限域 F 進行編碼操作。該有限域 F 可為伽羅瓦域 GF(2^8)。編碼器 102 與解碼器 110 也同意採用在有限域 F 中生成偽隨機值的一偽亂數發生器(PRNG, "Pseudorandom number generator")。

【0020】 編碼器 102 在一滑動編碼器視窗 $L \leq i < R$ 中對來源封包 (source packet) s_i 進行操作，其中 L、i 與 R 為封包的名義序列號，其用於控制編碼器 102 的操作。名義序列號可由編碼器 102 選擇，因為它們確實出現在來源串流 104 中並且解碼器 110 不在經解碼串流 112 中傳遞它們。編碼器 102 選擇 W=R-L 個偽隨機係數 c_i，將每個來源封包 s_i 乘以一個對應的係數，並傳送這些乘積的總和。因為來源封包 s_i 具有變化的長度，一常規的編碼器可單獨地傳送來源封包 s_i 的長度，造成一正比於編碼器視窗大小 W 的協定負載。相反地，編碼器 102 具有無關於編碼器視窗大小 W 的一恒定負載。編碼器 102 對向量(l_i, s'_i) 進行編碼而非對來源封包 s_i 進行編碼，其中 l_i 為來源封包 s_i 的長度，而 s'_i 為填有足夠的零(0)的來源封包 s_i，使得編碼器視窗中的所有的封包 s'_i 都具有相同的長度。例如: 向量(l_i, s'_i)包含(l_L, s'_L) (l_{L+1}, s'_{L+1}) $(l_{L+2}, s'_{L+2})\ldots$ (l_{R-1}, s'_{R-1}) 。

【0021】 解碼器 110 重構向量(l_i, s'_i)，其足以唯一地重構來源封包 s_i。解碼器 110 接收來源封包 s_i 的線性組合，並通過求解一線性方程組來解碼它們。一常規的解碼器可以使用高斯-喬登（Gauss-Jordan）消去法來求解該線性方程組。相反地，解碼器 110 可使用上下(LU, "Lower upper")分解來求解該線性方程組。為了解碼長度 k 的 N 個封包，常規解碼器使用與 N*(N*N+N*k)成比例的時間，而解碼器 110 使用的時間為 W*(N*N+N*k)，只要解碼器窗口不太大，則其要小得多。注意解碼器 110 可以使用其他方法來求解線性方程組。

【0022】 針對每個接收到的封包，解碼器 110 知道編碼器視窗大小 W 以及該編碼器視窗的開始位置 L。編碼器 102 將這些值作為負載傳送到解碼器 110。解碼器 110 還知道偽隨機編碼係數。編碼器 102 使用自 PRNG 取得的係數。解碼器 110 具有 PRNG 的一拷貝，其將生成相同的係數序列，前提是解碼器的 PRNG 具有與編碼器的 PRNG 相同的內部狀態。內部狀態可通過傳送編碼器的 PRNG 的種子、傳送編碼器的 PRNG 的內部狀態作為負載或用於固定的 PRNG、傳送編碼器的 PRNG 的開始序列號用於一特定封包而被同步化。

【0023】 編碼器狀態

【0024】 在本發明的示例中，編碼器 102 維持以下的狀態變數：

L：整數，用於編碼器視窗的"左"邊界的助記符號，初始設定為 1(或另一數字，取決於最初第一個來源封包的名義序列值)。

R：整數，用於編碼器視窗的"右"邊界的助記符號，初始設定為 1(或另一數字，取決於最初第一個來源封包的名義序列值)。

M：實數，代表編碼進度，初始設定為 1(或另一數字，取決於最初第一個來源封包的名義序列值而定)，並可為一有理數或具有常數分母的分數。

【0025】 在其操作期間，編碼器 102 生成在編碼器視窗 $L \le i < R$ 中的來源封包 s_i 的組合。

【0026】 在本發明的示例中，以下兩個參數控制編碼器 102 的操作：

速率：實數，$0 <$ 速率 ≤ 1，其為改錯碼的"速率"。針對每一經編碼的封包，編碼器 102 消耗速率個來源封包。相等地，針對每一來源封包，編碼器 102 產生 1/速率個編碼封包。

W_E：最大編碼器視窗大小。在正常操作期間，$0 \le R - L \le W_E$。

【0027】 在本發明的示例中，編碼率與最大編碼器視窗大小 W_E 可隨時間變化。一較低編碼率適用於具有較大錯誤率的狀況，而一較大的編碼器視窗適用於可以容忍額外解碼遲延的狀況。

【0028】 編碼器的操作

【0029】 圖 2 為例示本發明的示例中編碼器 102 的操作的方塊圖。編碼器 102 維持一來源封包 s_i 的編碼器視窗[L, R)。例如，編碼器視窗[1, 4)包括來源封包 s_1、s_2 與 s_3。編碼器 102 通過對每個來源封包 s_i 預置其長度 l_i 來擴增來源封包 s_i 以形成經擴增的封包 a_i。可選地，編碼器 102 還對每個來源封包 s_i 預置其名義序列號 i。編碼器 102 通過將零(0)附加到經擴增的封包 a_i 來填塞來源封包 s_i 以形成經填塞的封包 p_i，如果需要，這樣在編碼器視窗[L, R)中的所有經填塞的封包 p_i 具有相同的長度。編碼器 102 輸出所有經填塞的封包 p_i 的線性組合。

【0030】 當編碼器 102 自來源串流 104 接收到一新的來源封包 s_i 時，在本發明的示例中編碼器 102 可依下述操作。

【0031】 1. (經擴增的封包的形成)假設編碼器接收到來源封包 s_i，即來源串流 102 中第 i 個封包。假定 l_i 為來源封包 s_i 的長度。

【0032】 編碼器 102 形成一經擴增的封包 $a_i = (i, l_i, s_i)$。也就是說，經擴增的封包 a_i 包括來源封包 s_i 的名義序列號 i、來源封包 s_i 的長度 l_i，及來源封包 s_i 本身。名義序列號 i 為可選地使用，並可用於偵錯。例如參照圖 2 中的經擴增的封包 a_1 與 a_3。

【0033】 2. 增量 R 來推進編碼器視窗的右邊緣的時間。

【0034】 3. (調整編碼器視窗)如果 $R - L > W_E$，則增量 L 來保持該編碼器視窗小於最大編碼器視窗大小 W_E。

【0035】 4. (經填塞的封包的形成)假定 k 為編碼器視窗 $L \le i < R$ 中的經擴增的封包 a_i 的最大長度。針對編碼器視窗 $L \le i < R$ 中的每個經擴增的封包 a_i，通過利用零擴展 a_i 直到其長度為 k 來形成一經填塞的封包 p_i。在此，“零”為有限域 F 中的零。例如參照圖 2 中的經填塞的封包 p_1 與 p_3。

【0036】 5. 假定第一（第一）=真。該變數“第一”用於確定一經填塞的封包 p_i 是否為第一次被編碼。

【0037】 6. (生成經編碼的封包)當 $M \le R$ 時，進行下述：

a) (偽隨機種子的選擇)如果第一為真，假定 r=0。否則，假定 r 為一任意的非零隨機種子。

當經填塞的封包 p_i 為第一次被編碼時，此邏輯結合於下一步驟由一經填塞的封包編碼其本身。否則經填塞的封包 p_i

通過將其結合於在編碼器視窗中的所有其他經填塞的封包來編碼。

b) (偽亂數的生成) 如果 $r \neq 0$，針對 $L \leq i < R$，使用 r 來生成隨機係數 $c_i \in F$。
否則(r=0)，針對 $L \leq i < R < 1$ 假定 $c_i = 0$，並假定 $c_{R-1} = 1_F$。此處 1_F 為 F 的乘法單位（identity）。

c) 設定第一:=假。

d) (產生經填塞的封包的一線性組合)將經填塞的封包 p_i 解譯為向量空間 F^k 之上的元素，計算下式：

$$b = \sum_{L \leq i < R} c_i p_i \tag{1}$$

因此，經編碼的封包 b 的首位元組為係數向量 c 與由經填塞的封包 p 的首位元組所構成的向量的內積，其中該算術使用所選擇的伽羅瓦域中的加法與乘法。換言之，第一個經填塞的封包中的每個值與第一係數相乘，第二個經填塞的封包中的每個值與第二係數相乘，以此類推。經編碼的封包 b 中的每個值是經相乘的封包中的對應符號的和。

e) (傳送)在通道 108 上發送元組(L, R, r, b)。該元組也可稱為經編碼的封包。在一些示例中，(L, R, r)的資料壓縮或隱式版本被傳送。隱式傳送可包括傳送一封包校驗和中的一值，而不需要實際傳送該值，其中解碼器 110 搜尋使得該校驗和正確的一值。

f) (增量 M) 設定 M := M + 速率。R、M 與編碼率的使用允許每一個經填塞的封包 p_i 由其本身編碼一次，並且在第一次之後由在編碼器視窗中的其他經填塞的封包編碼數次(≥0)。

【0038】 圖 3 和圖 4 示出在本發明的示例中一種編碼器 102 用來傳送一改錯碼的方法 300 的流程圖。方法 300 可開始於圖 3 中的方塊 302。

【0039】 在方塊 302，編碼器 102 設定以下變數：左編碼器視窗邊界 L、右編碼器視窗邊界 R、編碼進度 M、編碼率、及最大編碼器視窗大小 W_E。方塊 302 可由方塊 304 跟隨在後。

【0040】 在方塊 304，編碼器 102 開始接收來源串流 104 中的來源封包 s_i。方塊 304 可由方塊 306 跟隨在後。

【0041】 在方塊 306，編碼器 102 開始以來源封包 s_i 的名義序列迴圈

通過來源封包 s_i。方塊 306 可由方塊 308 跟隨在後。

【0042】 在方塊 308，編碼器 102 基於來源封包 s_i 形成一經擴增的封包 $a_i = (i, l_i, s_i)$。方塊 308 可由方塊 310 跟隨在後。

【0043】 在方塊 310，編碼器 102 通過增量編碼器視窗右邊界 R 來推進該編碼器視窗。方塊 310 可由方塊 312 跟隨在後。

【0044】 在方塊 312，編碼器 102 確定編碼器視窗是否小於最大編碼器視窗大小 W_E。如果小於，則方塊 312 可由方塊 316 跟隨在後。否則方塊 312 可由方塊 314 跟隨在後。

【0045】 在方塊 314，編碼器 102 通過增量編碼器視窗左邊界 L 來維持編碼器視窗大小。方塊 314 可由方塊 316 跟隨在後。

【0046】 在方塊 316，編碼器 102 確定在編碼器視窗 $L \leq i < R$ 中的經擴增的封包 a_i 的最大長度 k，並通過將零附加到經擴增的封包 a_i 直到它們全部具有長度 k 來填塞它們。方塊 316 可由圖 4 中的方塊 318 跟隨在後。

【0047】 在方塊 318，編碼器 102 設定變數第一為真(例如 1)以表示經填塞的封包 p_i 要進行第一次編碼。如下所述，在第一次編碼經填塞的封包 p_i 之後，編碼器 102 設定變數第一為假(例如 0)。方塊 318 可由方塊 320 跟隨在後。

【0048】 在方塊 320，編碼器 102 確定經填塞的封包 p_i 是否要(再次)被編碼。如果是，方塊 320 可由方塊 322 跟隨在後。否則，方塊 320 可由方塊 306 跟隨在後以選擇下一個來源封包 s_i 進行處理。

【0049】 經填塞的封包 p_i 要在其尚未被編碼某個次數時(再次)被編碼。編碼器 102 使用編碼進度 M 來做出此確定。如下述，編碼器 102 在編碼經填塞的封包 p_i 之後，增量編碼進度 M 該編碼率。編碼器 102 在編碼進度 M 小於或等於編碼器視窗右邊界 R 的同時(再次)編碼經填塞的封包 p_i。

【0050】 在方塊 322，編碼器 102 確定經填塞的封包 p_i 是否要第一次被編碼。換言之，編碼器 102 確定變數第一是否被設定為真。如果是，則方塊 322 可由方塊 324 跟隨在後。否則方塊 322 可由方塊 328 跟隨在後。

【0051】 在方塊 324，編碼器 102 設定 PRNG 的種子 r 為零(0)，以指示經填塞的封包 p_i 要由其本身編碼，而不需要在編碼器視窗中的任何其他

稍早的經填塞的封包 p_i 來編碼。方塊 324 可由方塊 326 跟隨在後。

【0052】 在方塊 326，編碼器 102 針對 $L \leq i < R-1$ 設定 $c_i = 0$，並設定 $c_{R-1} = 1_F$，其中 1_F 為 F 的乘法單位。方塊 326 可由方塊 332 跟隨在後。

【0053】 在方塊 328，編碼器 102 假定用於 PRNG 的種子 r 為一任意的非零亂數。方塊 328 可由方塊 330 跟隨在後。

【0054】 在方塊 330，編碼器 102 使用種子 r 來生成偽隨機係數 c_i，其中針對 $L \leq i < R$，$c_i \in F$。方塊 330 可由方塊 332 跟隨在後。

【0055】 在方塊 332，編碼器 102 設定變數第一為假(例如 0)。方塊 332 可由方塊 334 跟隨在後。

【0056】 在方塊 334，編碼器 102 依照方程式 1 線性地結合在編碼器視窗中的係數 c_i 與對應的經填塞的封包 p_i 的乘積以形成經編碼的封包 b。在編碼器視窗中的經填塞的封包 p_i 包括在編碼器視窗的當前的經填塞的封包 p_{R-1} 以及稍早的經填塞的封包 $p_L \ldots p_{R-2}$。方塊 334 可由方塊 336 跟隨在後。

【0057】 在方塊 336，編碼器 102 在通信通道 108 之上傳送一元組(L, R, r, b)作為一經編碼的封包。方塊 336 可由 y 個方塊 338 跟隨在後。

【0058】 在方塊 338，編碼器 102 增量編碼進度 M 該編碼率。方塊 338 可由方塊 320 跟隨在後，以確定經填塞的封包 p_i 是否要再次被編碼。

【0059】 編碼過程的演示

【0060】 圖 5、圖 6、圖 7 以及圖 8 演示在本發明的示例中傳送一消息的方法 300。在此演示中，編碼器 102（圖 1）使用 GF(16)域，其中在該域中的每一符號可表示成從 0000 到 1111 的 4 位元的數字，或以十進位來說是 0 到 15。

【0061】 針對該演示，假設編碼器 102 要傳送消息“Quanta”。此消息為 6 個字元長，每一字元根據 ASCII 碼為 8 位元。該消息中的每一字元可由其低 4 位接著其高 4 位來表示。例如，“Q”的 ASCII 碼為 0x51，其可表示成兩個符號消息 1 5，而整個消息“Quanta”成為 1 5 5 7 1 6 14 6 4 7 1 6。假設此 12 個符號消息被分成具有隨機長度 3、4、3 與 2 的來源封包，如表 1 所示。

【0062】 表 1

封包	長度	數據
1	3	1 5 5
2	4	7 1 6 14
3	3	6 4 7
4	2	1 6

【0063】 因為該消息已經被分成 4 位元的符號，所以它們可表示成在 GF(16)域中的符號。這些來源封包被順序地發送到編碼器 102。

【0064】 在圖 5、圖 6、圖 7 以及圖 8 中，在該編碼過程中的偽隨機係數的生成示出為步驟 324/328 和 326/330，並且經填塞的封包的線性組合的產生示出為步驟 334。如上所述，編碼器 102 使用一 PRNG 生成偽隨機係數。當設定為一特定種子值時，該 PRNG 在此後產生同一偽亂數集，直到該種子被改變。編碼器 102 傳送用於生成經編碼的封包的種子值到解碼器 110，所以解碼器的 PRNG 與編碼器的 PRNG 被同步，即使封包損失，或未依順序傳遞。在本發明的示例中， PRNG 不會輸出任何的零(0)，因為零(0)並非一有利的係數。同時也如上所述，零(0)並未用作一種子值，因為編碼器 102 設定種子 r 為零(0)，以指示一經填塞的封包要由其本身編碼。

【0065】 經編碼的封包基於偽隨機係數由經填塞的封包的隨機線性組合所形成。考慮具有兩個係數 c_1 與 c_2 的兩個經填塞的封包 e 與 f，該線性組合 $c_1e + c_2f$ 系根據伽羅瓦域算術的規則來計算。圖 9 與圖 10 分別示出用於 GF(16)域的加法與乘法表。

【0066】 圖 5 示出在本發明的示例中來源封包 s_1 的處理，其長度為 3，且資料為 1 5 5。初始時，編碼器 102 設定編碼器視窗左邊界 L 為 1、編碼器視窗右邊界 R 為 1、編碼進度 M 為 1，編碼率為 0.5，且最大編碼器視窗大小 W_E 為 3。編碼器 102 基於來源封包 s_1 形成一經擴增的封包 a_1，並且基於經擴增的封包 a_1 形成一經填塞的封包 p_1。請注意編碼器 102 不會將任何的零(0)附加到經擴增的封包 a_1，因為在編碼器視窗[1, 1)中僅有一個經擴增的封包，所以不需要使得編碼器視窗[1, 1)中的所有經填塞的封包 p_i 都具有相同長度。同時請注意到，編碼器 102 在經擴增的封包 a_1 被形成之後增量編碼器視窗右邊界 R 以推進編碼器視窗到[1, 2)。

【0067】 經填塞的封包 p_1 被編碼三次。當經填塞的封包 p_1 第一次被編碼時，種子 r 被設定為零(0)，所以經填塞的封包 p_1 由其本身利用一單位係數進行"編碼"，所以經編碼的封包與經填塞的封包相同。當經填塞的封包 p_1 在每個後續次數被編碼時，種子 r 被設定為一亂數，且一偽隨機係數基於種子 r 利用 PRNG 來生成。因為在編碼器視窗[1, 2)中不存在稍早的經填塞的封包 p_1，所以方程式(1)僅產生經填塞的封包 p_1 與偽隨機係數的乘積。在每一次經填塞的封包 p_1 被編碼之後，編碼進度 M 被增量該編碼率。一旦編碼進度 M 大於編碼器視窗右邊界 R，編碼器 102 就處理下一個來源封包。

【0068】 圖 6 示出在本發明的示例中來源封包 s_2 的處理，其長度為 4，且資料為 7 1 6 14。編碼器 102 基於來源封包 s_2 形成一經擴增的封包 a_2，並基於經擴增的封包 a_2 形成一經填塞的封包 p_2。編碼器 102 在經擴增的封包 a_2 被形成之後增量編碼器視窗右邊界 R 以推進編碼器視窗到[1, 3)。編碼器 102 將一個零(0)附加到經擴增的封包 a_1，所以在編碼器視窗[1, 3)中經填塞的封包 p_1 與 p_2 具有相同長度。

【0069】 經填塞的封包 p_2 被編碼兩(2)次。當經填塞的封包 p_2 第一次被編碼時，種子 r 被設定為零(0)，所以經填塞的封包由其本身利用一單位係數進行"編碼"，所以經編碼的封包與經填塞的封包相同。當經填塞的封包 p_2 第二次被編碼時，種子 r 被設定為一亂數，且兩(2)個偽隨機係數基於種子 r 利用 PRNG 生成。依照方程式(1)將經填塞的封包 p_1、p_2 與兩個偽隨機係數的乘積進行線性組合。

【0070】 圖 7 示出在本發明的示例中來源封包 s_3 的處理，其長度為 3，且資料為 6 4 7。編碼器 102 基於來源封包 s_3 形成一經擴增的封包 a_3，並基於經擴增的封包 a_3 形成一經填塞的封包 p_3。編碼器 102 在經擴增的封包 a_3 被形成之後增量編碼器視窗右邊界 R 以推進編碼器視窗到[1, 4)。編碼器 102 將一個零(0)附加到經擴增的封包 a_1 與 a_3，所以在編碼器視窗[1, 4)中經填塞的封包 p_1、p2 與 p_3 具有相同長度。

【0071】 經填塞的封包 p_3 被編碼兩(2)次。當經填塞的封包 p_3 第一次被編碼時，種子 r 被設定為零(0)，所以經填塞的封包由其本身利用一單位係數進行"編碼"，所以經編碼的封包與經填塞的封包相同。當經填塞的封包

p_3 第二次被編碼時，種子 r 被設定為一亂數，且三(3)個偽隨機係數基於種子 r 利用 PRNG 產生。依照方程式(1)將經填塞的封包 p_1、p_2、p_3 與三個偽隨機係數的乘積進行線性組合。

【0072】 圖 8 示出在本發明的示例中來源封包 s_4 的處理，其長度為 2，且資料為 1 6。編碼器 102 基於來源封包 s_4 形成一經擴增的封包 a_4，並基於經擴增的封包 a_4 形成一經填塞的封包 p_4。編碼器 102 在經擴增的封包 a_4 被形成之後增量編碼器視窗右邊界 R 以推進編碼器視窗到[1, 5)。編碼器視窗左邊界由 1 增量到 2，以維持最大編碼器視窗大小 W_E。編碼器 102 將一個零(0)附加到經擴增的封包 a_3 與 a_4，所以在編碼器視窗[2, 5)中經填塞的封包 p_2、p_3 與 p_4 具有相同長度。

【0073】 經填塞的封包 p_3 被編碼兩(2)次。當經填塞的封包 p_3 第一次被編碼時，種子 r 被設定為零(0)，所以經填塞的封包由其本身利用一單位係數進行"編碼"，所以經編碼的封包與經填塞的封包相同。當經填塞的封包 p_3 第二次被編碼時，種子 r 被設定為一亂數，且三(3)個偽隨機係數基於種子 r 利用 PRNG 生成。依照方程式(1)將經填塞的封包 p_1、p_2、p_3 與三個偽隨機係數的乘積進行線性組合。

【0074】 解碼器概述

【0075】 解碼器 110 在步驟 6e 接收由編碼器 102 所傳送的元組(L_E, R_E, r, b)，並重構來源封包。以下具有下標"E"的變數為編碼器的值，具有下標"D"的變數為解碼器的值。在本發明的示例中，為了重構來源封包，解碼器 110 維持一解碼矩陣 1100，如圖 11 所示。

【0076】 請回想編碼器 102 根據方程式(1)組合經填塞的的封包。因此，編碼器 102 計算 $B = AX$，其中 A 為由 PRNG 所計算的一偽隨機係數的矩陣，X 為一矩陣，其列是經填塞的封包(經填塞的封包的列向量)，並且 B 為利用方程式(1)所計算的矩陣(經編碼的封包的列向量)。

【0077】 在原則上，A、B 與 X 為無限矩陣。然而對於某個視窗大小 W_D，針對 $L \le i、j < L + W_D$，解碼器 110 僅保持 A 的一子矩陣 A_{ij}。視窗大小 W_D 可被設定為至少與最大編碼器視窗大小 W_E 一樣大，並可能更大。

【0078】 當接收一元組(L_E, R_E, r, b)時，解碼器 110 重構在步驟 6b 中

由編碼器 102 所計算的係數 c，並輸入偽隨機係數 c 與經編碼的封包 b 作為解碼矩陣 1100 的一新的列。

【0079】 然後解碼器 110 降低解碼矩陣 1100 為一列階梯形矩陣（row-echelon form）。一矩陣如果一列的首項係數嚴格地在它上方的所有列的首項係數的右方時即為列階梯形矩陣，其中一列的首項係數為該列中最左方的非零元素。

【0080】 A 的子矩陣如果對於 $i < M_D$'來說所有對角線條目 A_{ii} 為非零並且如果對於 $i < M_D$'與 $j \geq M_D$'來說 A_{ij} 為零，則其被稱為可解碼最高到列 M_D'。因此，該矩陣可被解碼最高到列 2，而非最高到列 3，這是由於第四列尚未被填充，所以第三列中的第二個係數 c 無法利用第四列被消去。

$$\begin{pmatrix} c & c & & \\ & c & & \\ & & c & c \end{pmatrix}$$

【0081】 每當 A 的子矩陣可解碼最高到列 M_D'時，第一個 M_D'列可經由後向替換（backward substitution）來降低為單位，允許該第一個 M_D' 經填塞的封包被解碼。

【0082】 在本發明的示例中，解碼器 110 經由列階梯形矩陣(REF)加上後向替換來操作，而一常規的解碼器降低該矩陣為經降低的列階梯形矩陣(RREF)。REF 在利用由編碼器 102 所使用的矩陣的帶狀結構時可能遠勝於 RREF。

【0083】 請參照圖 11，在本發明的示例中，解碼器 110 維持解碼矩陣 1110。一空的條目表示零(0)，“c”表示一係數，並且“b”表示來自從通道 108 所接收的一經編碼的封包的一資料項目。解碼矩陣 1110 具有[A||B]形式，其中 A 為偽隨機係數，B 為資料元素。解碼器 110 要針對 X 來求解方程組 AX=B，其中 B 為自通道 108 所接收的經編碼的封包，X 為來源封包的重構，並且 A 為偽隨機係數的矩陣(在圖 11 中的 1、0 與 c)。每一列對應於一經解碼的封包，每一列對應於一來源封包。解碼器 110 將 A 維持在列階梯形矩陣，如圖 11 所示。每當有可能時，解碼器 110 針對 $i、j < M_D$，將解碼矩陣 1100 的一首碼 A_{ij} 降低為單位矩陣。當此狀況發生時，針對 $i < M_D$

的來源封包 i 已經被解碼。A 矩陣在原則上為無限的，但解碼器 110 僅將列/行的滑動解碼器視窗 i 維持在 $L_D \leq i < L_D + W_D$ 的範圍中。

【0084】 解碼器的操作

【0085】 解碼器 110 利用初始被設定為所接收的元組中的編碼器視窗左邊界 L_E 的相關聯的狀態變數解碼器視窗左邊界 L_D 和解碼進度 M_D 來維持如上所述的解碼矩陣 1100。解碼器 110 存儲矩陣 A 的列/行$[L_D, L_D + W_D)$，並且解碼矩陣 1100 始終可解碼最高到列 M_D 並且可解碼最高到列 M_D'。

【0086】 當接收一元組(L_E, R_E, r, b)時，解碼器的操作如下述：

【0087】 1. (丟下舊封包) 如果 $L_E < L_D$ 則停止。該封包在給定的解碼器窗口大小 W_D 之下太舊而無法被解碼。

【0088】 2. (調整解碼視窗) 如果 $R_E > L_D + W_D$，設定 $L_D = R_E - W_D$。在此步驟之後，$R_E \leq L_D + W_D$ 且所接收到的元組落在解碼器窗口之內。

【0089】 3. (遺失的通知) 如果 $M_D < L_D$，則通知使用者來源封包$[M_D', L_D)$已經遺失，並設定 $M_D := L_D$。使用者可為另一個應用，諸如一視頻編解碼器。

【0090】 4. (擴增解碼矩陣) 生成如解碼器 102 的步驟 6b 中的係數 c。將 c 與 b 加入到解碼矩陣 1100 的第一個零列。

【0091】 5. (降低到列階梯形矩陣) 經由初等列操作降低 A 到列階梯形矩陣。將相同的列更新應用到矩陣 A 與 B。

【0092】 6. (批量解碼) 假定 M_D'為矩陣 A 可被解碼最高的第 M_D'列的最大整數。

如果 $M_D' > M_D$，經由向後替換來解碼在範圍$[M_D, M_D')$中的經編碼的封包。

【0093】 7. (傳遞經解碼的封包) 當 $M_D < M_D$'時，進行下述動作：

a) 矩陣 B 的第 M_D 列包含第 M_D 個經填塞的封包 p_M。從經填塞的封包 p_M 擷取來源封包，並將其傳遞給使用者。

b) 增量 M_D。

【0094】 圖 12 和圖 13 為本發明的示例中一種解碼器 110(圖 1)用於解碼一更正錯碼的方法 1200 的流程圖。方法 1200 可開始於圖 12 中的方塊

1202。

【0095】 在方塊 1202，解碼器 110 設定最大解碼器視窗大小 W_D。方塊 1202 可由方塊 1204 跟隨在後。

【0096】 在方塊 1204，解碼器 110 開始在通信通道 108 上接收元組。解碼器 110 設定解碼器視窗邊界 L_D 和解碼進度 M_D 等於所接收的第一元組中的編碼器視窗左邊界 L_E，並且設定解碼器視窗右邊界 R_D 等於所接收的第一元組中的編碼器視窗右邊界 R_E。方塊 1204 可由方塊 1206 跟隨在後。

【0097】 在方塊 1206，解碼器 110 開始以元組被接收的次序迴圈通過它們。方塊 1206 可由方塊 1208 跟隨在後。

【0098】 在方塊 1208，解碼器 110 確定元組中的經編碼的封包是否對於一給定的解碼器視窗為太舊(例如 $L_E < L_D$)。如果是，則方塊 1208 可由方塊 1210 跟隨在後。否則方塊 1208 可由方塊 1212 跟隨在後。

【0099】 在方塊 1210，解碼器 110 丟下舊的經編碼的封包。方塊 1210 可迴圈回到方塊 1206 以處理下一個元組。

【0100】 在方塊 1212，解碼器 110 確定經編碼的封包是否在解碼器視窗之內(例如 RE > LD + WD)。如果是，則方塊 1212 可由方塊 1214 跟隨在後。否則方塊 1212 可由方塊 1216 跟隨在後。

【0101】在方塊 1214，解碼器 110 移動解碼器視窗左邊界 L(例如設定 $L_D = R_E - W_D$)，所以經編碼的封包落在經解碼窗口之內。方塊 1214 可由方塊 1216 跟隨在後。

【0102】在方塊 1216，解碼器 110 確定是否已經遺失比當前的經編碼的封包舊的任何經編碼的封包(例如 $M_D < L_D$)。如果是，則方塊 1216 可由方塊 1218 跟隨在後。否則方塊 1216 可由方塊 1220 跟隨在後。

【0103】在方塊 1218，解碼器 110 通知使用者來源封包[M_D, L_D)的遺失，並設定解碼進度 M_D 等於解碼視窗左邊界 L_D。方塊 1218 可由方塊 1220 跟隨在後。

【0104】在方塊 1220，解碼器 110 通過基於在元組中所接收的種子 r 生成偽隨機係數 c，將偽隨機係數 c 與經編碼的封包 b 相加作為解碼矩陣 1110 的一新的列來擴增解碼矩陣 1100。方塊 1220 可由方塊 1222 跟隨在後。

【0105】在方塊 1222，解碼器 110 降低解碼矩陣 1110 為一列階梯形矩陣。方塊 1222 可由圖 13 中的方塊 1224 跟隨在後。

【0106】在方塊 1224，解碼器 110 確定解碼矩陣 1110 中的矩陣 A 是否能夠被進一步解碼最高到一列 M_D'。例如，解碼器 110 確定矩陣 A 是否可被解碼最高到列 M_D'，然後確定矩陣 A 是否已經被解碼最高到列 M_D' (例如 $M_D' > M_D$)。如果矩陣 A 可被進一步解碼最高到列 M_D'，方塊 1224 可由方塊 1226 跟隨在後。否則方塊 1224 可迴圈回到方塊 1206 以處理下一個元組。

【0107】在方塊 1226，解碼器 110 解碼在範圍[M_D, M_D')中的經編碼的封包。如上述，解碼器 110 可以使用向後替換來解碼經編碼的封包。方塊 1226 可由方塊 1228 跟隨在後。

【0108】在方塊 1228，解碼器 110 確定是否存在在方塊 1226 中從經編碼的封包所解碼的任何來源封包要被傳遞到使用者(例如 $M_D < M_D'$)。如果存在，方塊 1228 可由方塊 1230 跟隨在後。否則方塊 1228 可迴圈回到方塊 1206 以處理下一個元組。

【0109】在方塊 1230，解碼器 110 自矩陣 B 的第 M_D 列取得一經填塞的封包，自該經填塞的封包擷取一來源封包，並傳遞該來源封包到使用者。方塊 1230 可由方塊 1232 跟隨在後。

【0110】在方塊 1232，解碼器 110 增量解碼進度 M_D。方塊 1232 可迴圈回到方塊 1228。

【0111】解碼過程的演示

【0112】圖 14、圖 15、圖 16 以及圖 17 演示在本發明的示例中對元組/經編碼的封包進行解碼的方法 1200。此為該稍早演示的延續，其中編碼器 102(圖 1)生產九(9)個經編碼的封包，如表 2 所示。

【0113】表 2

L_E	R_E	r	數據
1	2	0	1 3 1 5 5
1	2	477	10 13 10 4 4
1	2	448	2 6 2 10 10
1	3	0	2 4 7 1 6 14
1	3	244	3 11 11 11 13 12

1	4	0	3 3 6 4 7 0
1	4	503	9 6 8 2 13 10
2	5	0	4 2 1 6 0 0
2	5	305	0 1 12 3 1 11

【0114】為了演示解碼器 110(圖 1)的能力，假設前四(4)個經編碼的封包已經遺失，並且經編碼的封包 5、6、7 以及 8 已被接收。

【0115】圖 14 示出在本發明的示例中第五個經編碼的封包的處理。初始時，解碼器 110 設定解碼器窗口左邊界 L_D 為 1、解碼進度 M 為 1、並且解碼器窗口大小 W_D 為 3。因為在解碼矩陣中僅存在一列，所以解碼器 110 並不降低解碼矩陣為列階梯形矩陣，並繼續處理下一個經編碼的封包。

【0116】圖 15 示出在本發明的示例中第六個經編碼的封包的處理。現在在解碼矩陣中存在兩列，但解碼器 110 不會降低解碼矩陣為列階梯形矩陣，因為該解碼矩陣已經為列階梯形矩陣。

【0117】圖 16 示出在本發明的示例中第七個經編碼的封包的處理。現在在解碼矩陣中存在三列。為了降低該解碼矩陣成為列階梯形矩陣，在步驟 1222，解碼器 110 使用 A(0,0)來清除 A(2,0)。換言之，解碼器 110 使用初等列操作來組合列 0(其包含 A(0,0))與列 2(其包含 A(2,0))，以這樣的方式來使得 A(2,0)==0。這通過將列 0 乘以因數 A(2,0)/A(0,0)，然後從列 2 減去它來完成。

$$A(2,:) = A(2,:) - (A(2,0) / A(0,0)) * A(0,:)$$

【0118】注意標記（notation）A(0,:)指代二維矩陣 A 的列 0。類似地，解碼器 110 通過使用初等列操作來互換列 1 與 2，以交換 A(1,:)與 A(2,:)。

【0119】在步驟 1224，解碼器 110 使用初等列操作來將新的可解碼列由列階梯形矩陣轉換成經降低的列階梯形矩陣。解碼器 110 通過將 A(0,0)乘以一因數 1/A(0,0)來縮放 A(0,0)，使得之後 A(0,0)將等於 1。這是可能的，因為在一域中的每一個非零的元素具有一乘法逆元（multiplicative inverse）。解碼器 110 通過將列 2 乘以因數 A(1,2)/A(2,2)，然後自列 1 減去列 2 來替換 A(2,2)以清除 A(1,2)。這將設定 A(1,2)為 0。

【0120】在步驟 1224 中的過程被稱為向後替換，並可應用在從矩陣中的右下方到左上方的一序列中以最小化工作量。

【0121】當解碼矩陣的係數區段具有一單個 1 時，則該係數矩陣的資料部分恰好為對應於該 1 的位置的來自編碼器 102 的經填塞的封包。降低為列階梯形矩陣確保解碼矩陣的每一列不包含關於稍早的來源封包的資訊。在列成為可解碼的之後，最後的向後替換步驟確保每一列不包含關於稍後的來源封包的資訊。因此，在該點處每一經解碼的列恰好具有可被讀出並傳遞的一個來源封包。

【0122】圖 17 示出在本發明的示例中第八個經編碼的封包的處理。在步驟 1212 到 1214，解碼器 110 通過設定解碼器視窗左邊界 L_D 等於 2($L_D = R_E - W_D = 5 - 3 = 2$)來推進解碼器窗口。注意，為了節約記憶體，解碼器 110 可以推進解碼器視窗以丟棄偽隨機係數和所傳遞的在解碼器窗口之外的來源封包。

【0123】編碼器 102 與解碼器 110(圖 1)可在軟體、硬體或軟體與硬體的組合中實現。圖 18 為在本發明的示例中一種用於實現編碼器 102 或解碼器 110 的計算設備 1800 的方塊圖。用於傳送並解碼改錯碼的碼 1802 存儲在一非暫時性電腦介質 1804 中，諸如一唯讀記憶體。一處理器 1806 執列指令 1802 來提供所描述的特徵與功能性，並通過一網路介面 1808 傳送或接收經編碼的封包。

【0124】商業應用

【0125】本發明可以改善 Wi-Fi 或移動電話網絡上的視頻會議的品質。Wi-Fi 網路已經被測量，且已經觀察到有意義的封包遺失，其可由本發明的編碼器 102、解碼器 110 以及改錯碼來解決。

【0126】本發明可應用到其中存在多個通信路徑的情況，諸如具有多個蜂窩數據機的移動系統。

【0127】本發明可以授權給一移動運營商(3G/4G 等)以改善手機與基站之間的通信。

【0128】本發明可以作為一雲服務(使用在該雲中的一代理端點)提供給個人客戶以改善移動通信並提供一多路徑解決方案。

【0129】由前所述，將理解本文已出於例示的目的描述了本公開的各實施例，並且可做出各種修改而不脫離本公開的精神和範圍。因此，本文所公開的各實施例並非意圖限制，其真實範圍和精神由以下的權利要求所指明。

【符號說明】

100 系統

102 編碼器

104 來源串流

106 編碼串流

108 通道

110 解碼器

112 解碼串流

300 方法

302, 304, 306, 308, 310, 312,

314, 316, 318, 320, 322, 324,

326, 328, 330, 332, 334, 336,

338 方塊

1200 方法

1202, 1204, 1206, 1208, 1210,

1212, 1214, 1216, 1218, 1220,

1222, 1224, 1226, 1228, 1230,

1232 方塊

1800 運算裝置

1802 碼

1804 非暫時性電腦可讀取媒體

1806 微處理器

1808 網路介面

104 年 11 月 30 日修正替換頁

申請專利範圍

1. 一種傳送一更改錯碼的方法，所述方法包含：

接收一來源封包；

在接收所述來源封包之後，推進一滑動編碼器視窗；

生成經編碼的封包，包含：

第一次編碼所述來源封包，包含單獨編碼所述來源封包；以及

在所述第一次之後編碼所述來源封包數次，其中所述數次等於或大於 0，並且在所述第一次之後所述數次中的每次編碼所述來源封包包含利用所述滑動視窗中的數個較舊來源封包來編碼所述來源封包，所述滑動視窗中的較舊來源封包的數目等於或大於 0；以及

在一通信通道上傳送所述經編碼的封包。

2.根據權利要求 1 所述的方法，在生成經編碼的封包之前，進一步包含：

利用所述來源封包的長度值，擴增所述來源封包；以及

利用數個 0 填塞所述來源封包，使得所述來源封包和所述滑動編碼器視窗中的所述較舊來源封包具有相同的長度。

3. 根據權利要求 1 所述的方法，其中第一次編碼所述來源封包包含設定所述來源封包為所述經編碼的封包之一。

4. 根據權利要求 3 所述的方法，其中在所述第一次之後每次編碼所述來源封包包含：

設定一種子為一隨機值；

自所述種子生成偽隨機係數；

確定所述偽隨機係數與所述滑動編碼器視窗中的來源封包的乘積；以及

線性地組合所述乘積。

5. 根據權利要求 4 所述的方法，其中傳送所述經編碼的封包包含傳送每一經編碼的封包連同所述滑動編碼器視窗的邊界與所述種子。

6. 根據權利要求 2 所述的方法，其中在所述第一次之後於編碼經填塞的所述來源封包數次之前，進一步包含：

確定在其中傳送所述經編碼的封包的一通信通道的狀況；以及

改變所述數次為一新值。

7. 根據權利要求 1 所述的方法，在推進所述滑動編碼器視窗之前，進一步包含：

確定在其中傳送所述經編碼的封包的一通信通道的狀況；以及

基於所述狀況改變所述滑動編碼器視窗為一新的大小。

8. 一種解碼一更改錯碼的方法，所述方法包含：

於一通信通道上接收一包括一種子與一經編碼的封包的元組；

基於所述種子，重構由一編碼器用來產生所述經編碼的封包的偽隨機係數集；

輸入所述偽隨機係數集與所述經編碼的封包作為一解碼矩陣中的一列，其中所述解碼矩陣包含一第一矩陣與一第二矩陣，所述第一矩陣包含其中包括所述偽隨機係數集的偽隨機係數集合，並且所述第二矩陣包含其中包括所述經編碼的封包的經編碼的封包；

降低所述解碼矩陣為一列階梯形矩陣；

在降低所述解碼矩陣為一列階梯形矩陣之後，確定所述解碼矩陣是否可解碼最高到數列；

當所述解碼矩陣可解碼最高到所述數列時，通過向後替換來降低所述解碼

矩陣最高到所述數列，使得所述第一矩陣的一部分成為一單位矩陣，並且所述第二矩陣的一對應部分包含來源封包；以及

自經填塞的封包擷取所述來源封包。

9. 根據權利要求 8 所述的方法，進一步包含在重構所述偽隨機係數集之前，確定所述元組是否並不比一滑動解碼視窗更舊，其中任何比所述滑動解碼視窗更舊的元組被丟棄。

10. 根據權利要求 9 所述的方法，進一步包含：

確定所述元組是否比所述滑動解碼窗口更新；以及

當所述元組比所述滑動解碼窗口更新時，推進所述滑動解碼窗口以包括所述元組。

11. 根據權利要求 8 所述的方法，進一步包含：

在擷取每一來源封包之後，增量一解碼進度變數；

接收另一個元組；

確定所述解碼進度變數是否比一滑動解碼窗口更舊；以及

當所述解碼進度變數比所述滑動解碼視窗更舊時，生成指示至少已經遺失一來源封包的一警報。

12. 一種計算設備，其包含：

一網路介面；

一編碼器，用於：

接收一來源封包；

在接收所述來源封包之後，推進一滑動編碼器視窗；

生成經編碼的封包，包含：

第一次編碼所述來源封包，包含單獨編碼所述來源封包；以及

在所述第一次之後編碼所述來源封包數次，其中所述數次等於或大於 0，並且在所述第一次之後所述數次的每次編碼所述來源封包包含利用所述滑動視窗中的數個較舊來源封包來編碼所述來源封包，所述滑動視窗中的較舊來源封包的數目等於或大於 0；以及

在一通信通道上傳送所述經編碼的封包。

13. 根據權利要求 12 所述的計算設備，在生成經編碼的封包之前，所述處理器要執行指令以：

利用所述來源封包的長度值，擴增所述來源封包；以及

利用數個 0 填塞所述來源封包，使得所述來源封包和所述滑動編碼器視窗中的所述較舊來源封包具有相同的長度。

14. 根據權利要求 12 所述的計算設備，其中第一次編碼所述來源封包包含設定所述來源封包為所述經編碼的封包之一。

15. 根據權利要求 14 所述的計算設備，其中在所述第一次之後每次編碼所述來源封包包含：

設定一種子為一隨機值；

自所述種子生成偽隨機係數；

確定所述偽隨機係數與所述滑動編碼器視窗中的來源封包的乘積；以及

線性地組合所述乘積。

16. 根據權利要求 15 所述的計算設備，其中傳送所述經編碼的封包包含傳送每一經編碼的封包連同所述滑動編碼器視窗的邊界與所述種子。

17. 根據權利要求 13 所述的計算設備，其中在所述第一次之後於編碼經填塞的所述來源封包數次之前，所述處理器要執行指令以：

確定在其中傳送所述經編碼的封包的一通信通道的狀況；以及

改變所述數次為一新值。

18. 根據權利要求 12 所述的計算設備，在推進所述滑動編碼器視窗之前，所述處理器要執行指令以：

確定在其中傳送所述經編碼的封包的一通信通道的狀況；以及

基於所述狀況改變所述滑動編碼器視窗為一新的大小。

19. 一種計算設備，包含：

一網路介面；

一解碼器，用於：

經由所述網路介面接收一包括一種子與一經編碼的封包的元組；

基於所述種子，重構由一編碼器用來產生所述經編碼的封包的偽隨機係數集；

輸入所述偽隨機係數集與所述經編碼的封包作為一解碼矩陣中的一列，其中所述解碼矩陣包含一第一矩陣與一第二矩陣，所述第一矩陣包含其中包括所述偽隨機係數集的偽隨機係數集合，並且所述第二矩陣包含其中包括所述經編碼的封包的經編碼的封包；

降低所述解碼矩陣為一列階梯形矩陣；

在降低所述解碼矩陣為一列階梯形矩陣之後，確定所述解碼矩陣是否可解碼最高到數列；

當所述解碼矩陣可解碼最高到所述數列時，通過向後替換來降低所述解碼矩陣最高到所述數列，使得所述第一矩陣的一部分成為一單位矩陣，並且所述第二矩陣的一對應部分包括來源封包；以及

自經填塞的封包擷取所述來源封包。

20. 根據權利要求 19 所述的計算設備，其中所述處理器進一步用於在重構

所述偽隨機係數集之前，確定所述元組是否不比一滑動解碼視窗更舊，其中任何比所述滑動解碼視窗更舊的元組被丟棄。

21. 根據權利要求 20 所述的計算設備，其中所述處理器進一步用於：

確定所述元組是否比所述滑動解碼窗口更新；以及

當所述元組比所述滑動解碼窗口更新時，推進所述滑動解碼窗口以包括所述元組。

22. 根據權利要求 19 所述的計算設備，其中所述處理器進一步用於：

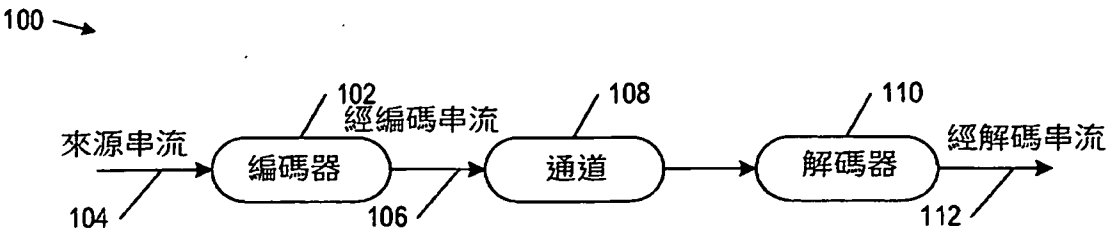
在擷取每一來源封包之後，增量一解碼進度變數；

接收另一個元組；

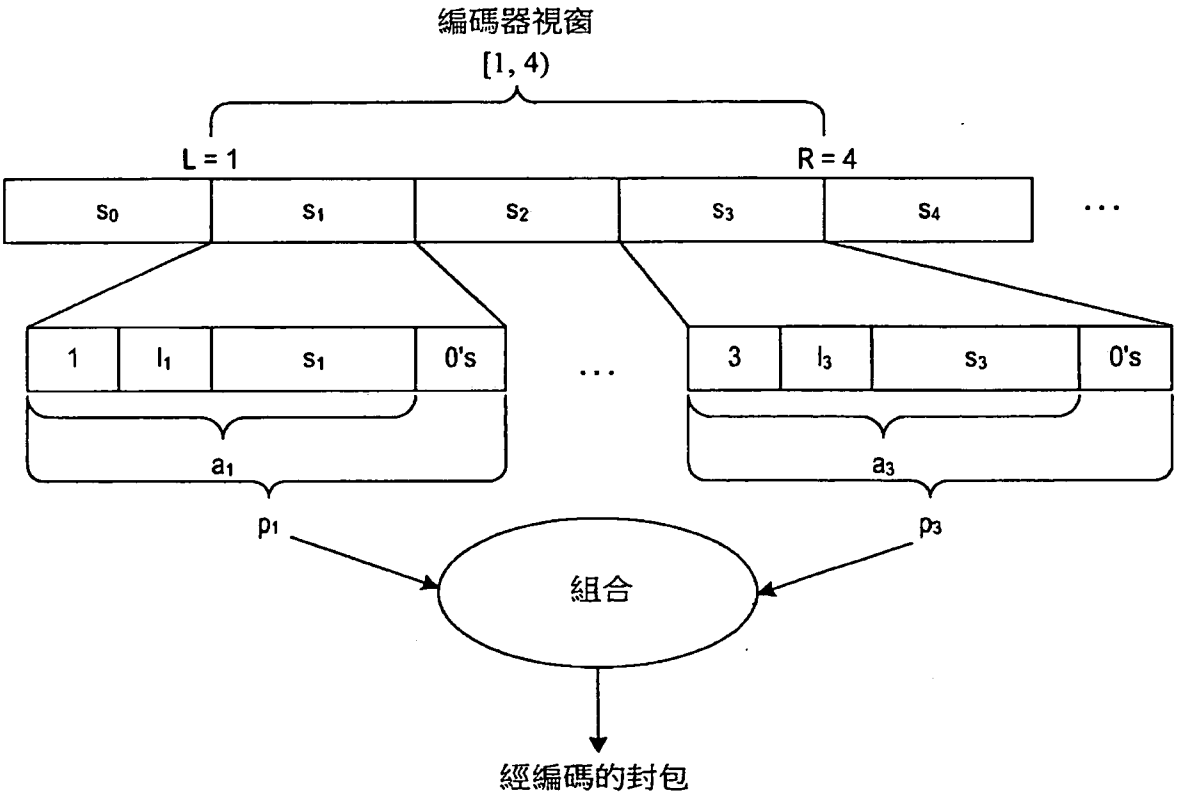
確定所述解碼進度變數是否比一滑動解碼窗口更舊；以及

當所述解碼進度變數比所述滑動解碼視窗更舊時，生成指示至少已經遺失一來源封包的一警報。

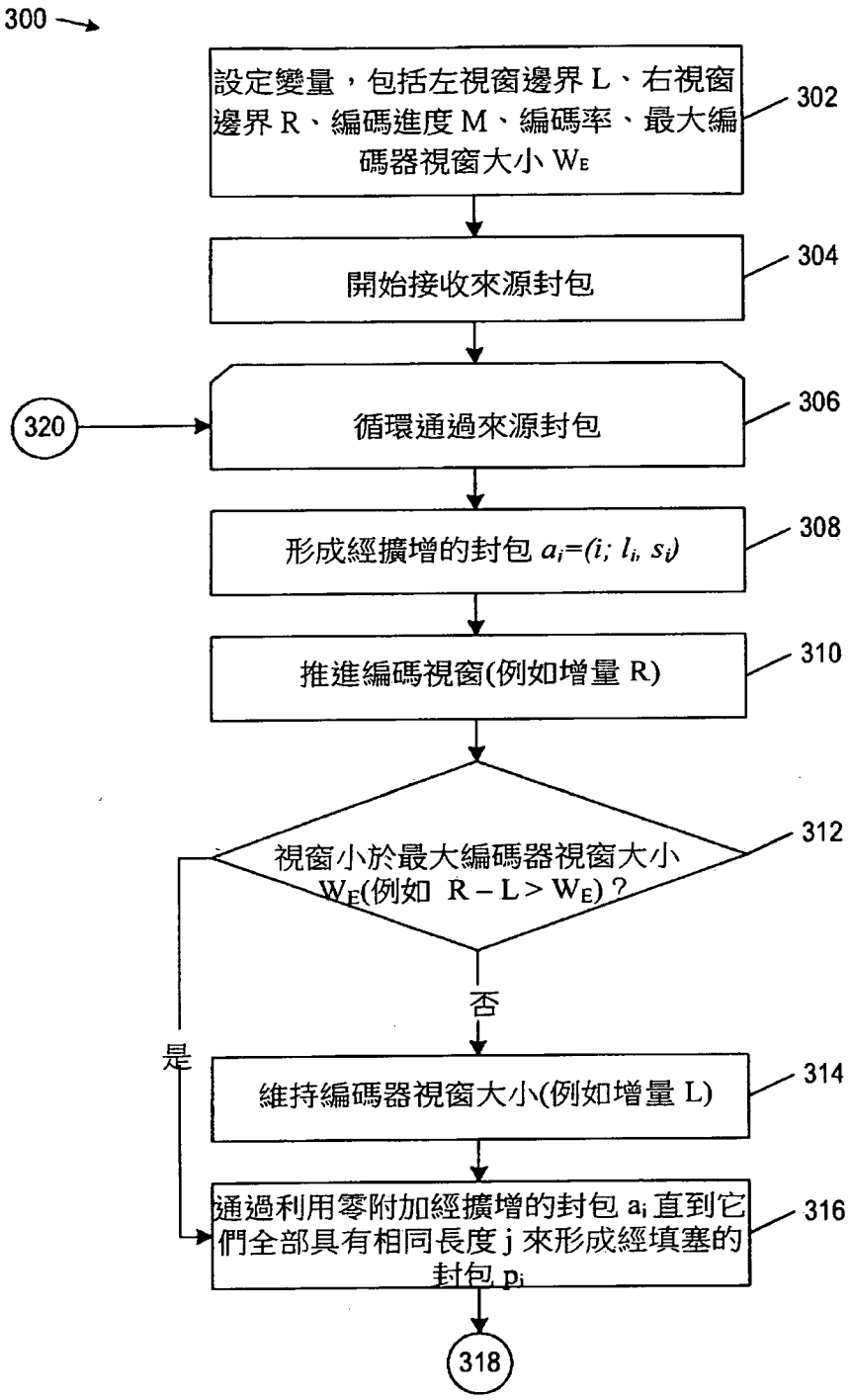
圖式


100
來源串流
104
編碼器
102
經編碼串流
106
通道
108
解碼器
110
經解碼串流
112

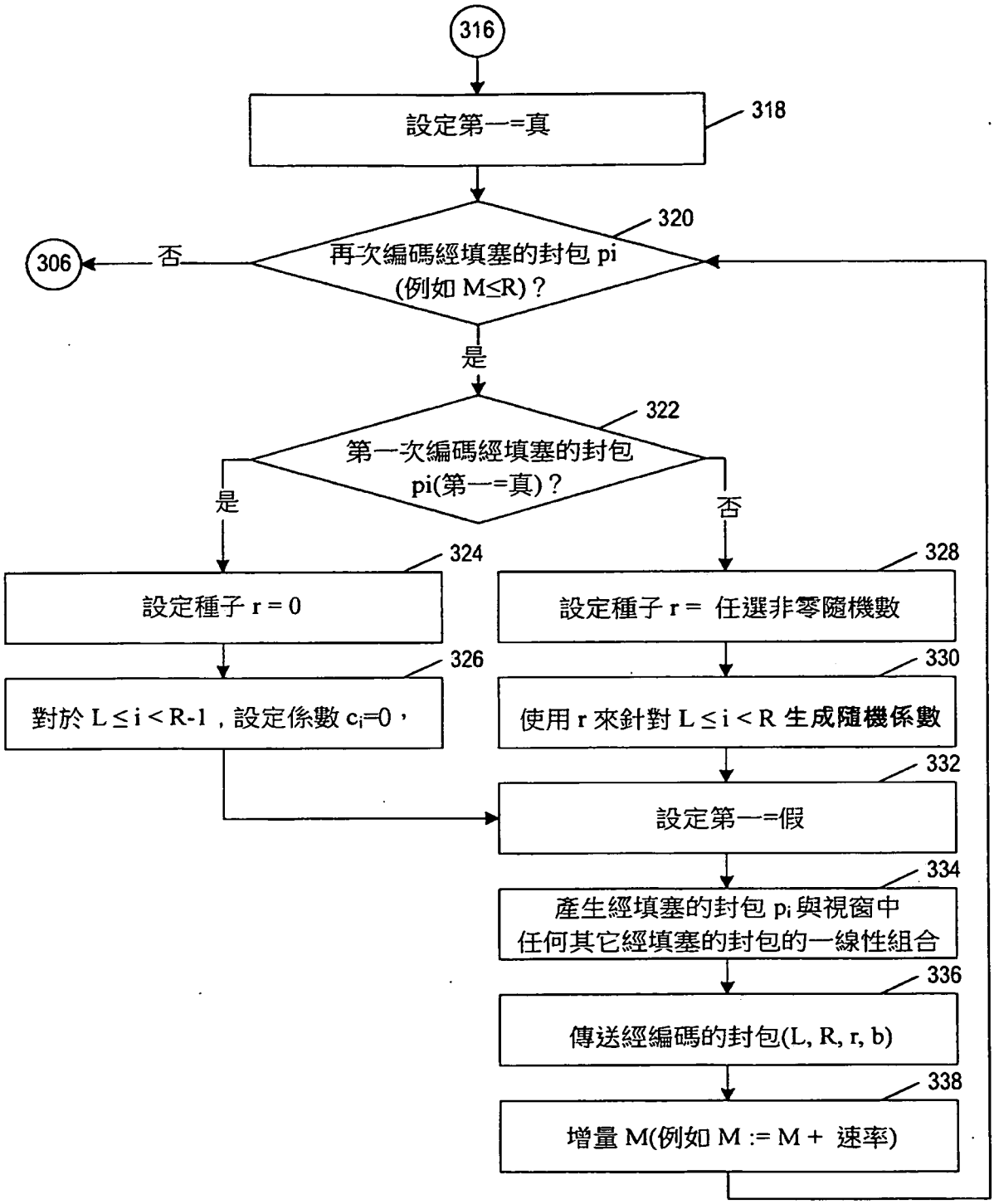
第一圖


編碼器視窗
[1, 4)
L = 1
R = 4
s0
s1
s2
s3
s4
1
l1
s1
0's
3
l3
s3
0's
a1
a3
p1
p3
組合
經編碼的封包

第二圖


300
設定變量，包括左視窗邊界 L、右視窗邊界 R、編碼進度 M、編碼率、最大編碼器視窗大小 W_E
302
開始接收來源封包
304
320
循環通過來源封包
306
形成經擴增的封包 $a_i=(i; l_i, s_j)$
308
推進編碼視窗(例如增量 R)
310
視窗小於最大編碼器視窗大小 W_E(例如 $R-L>W_E$)？
312
否
是
維持編碼器視窗大小(例如增量 L)
314
通過利用零附加經擴增的封包 a_i 直到它們全部具有相同長度 j 來形成經填塞的封包 p_j
316
318

第三圖


316
設定第一=真
318
320
再次編碼經填塞的封包 pi
(例如 M≤R)？
否
306
是
322
第一次編碼經填塞的封包
pi(第一=真)？
是
否
324
設定種子 r = 0
326
對於 L ≤ i < R-1，設定係數 ci=0，
328
設定種子 r = 任選非零隨機數
330
使用 r 來針對 L ≤ i < R 生成隨機係數
332
設定第一=假
334
產生經填塞的封包 pi 與視窗中
任何其它經填塞的封包的一線性組合
336
傳送經編碼的封包(L, R, r, b)
338
增量 M(例如 M := M + 速率)

第四圖

步驟	描述	數據
308	形成經擴增的封包	1 3 1 5 5
310	增量 R	R = 2
312-316	調整視窗	L = 1
316	在[L.R)視窗中形成填塞的封包	1 3 1 5 5
322-338	生成經編碼的封包	
324 或 328	生成種子	第一=1、種子=0
326 或 330	生成係數	1
332	設定第一為 0	第一=0
334	產生封包的一線性組合	1 3 1 5 5
336	傳送(L, R, r, 數據)	1 2 0 1 3 1 5 5
338	設定 M := M + 速率	M = 1.500000
324 或 328	生成種子	第一=0、種子=477
326 或 330	生成係數	10
332	設定第一為 0	第一=0
334	產生封包的一線性組合	10 13 10 4 4
336	傳送(L, R, r, 數據)	1 2 477 10 13 10 4 4
338	設定 M := M + 速率	M = 2.000000
324 或 328	生成種子	第一=0、種子=448
326 或 330	生成係數	2
332	設定第一為 0	第一=0
334	產生封包的一線性組合	2 6 2 10 10
336	傳送(L, R, r, 數據)	1 2 448 2 6 2 10 10
338	設定 M := M + 速率	M = 2.500000

第五圖

步驟	描述	數據
308	形成經擴增的封包	2 4 7 1 6 14
310	增量 R	R = 3
312-316	調整視窗	L = 1
316	在[L..R)窗中形成經填塞的封包	1 3 1 5 5 0 2 4 7 1 6 14
322-338	生成經編碼的封包	
324 或 328	生成種子	第一=1、種子=0
326 或 330	生成係數	0 1
332	設定第一為 0	第一=0
334	產生封包的一線性組合	2 4 7 1 6 14
336	傳送(L, R, r, 數據)	1 3 0 2 4 7 1 6 14
338	設定 M := M + 速率	M = 3.000000
324 或 328	生成種子	第一=0、種子=244
326 或 330	生成係數	13 7
332	設定第一為 0	第一=0
334	生成封包的一線性組合	3 11 11 11 13 12
336	傳送(L, R, r, 數據)	1 3 244 3 11 11 11 13 12
338	設定 M := M + 速率	M = 3.500000

第六圖

步驟	描述	數據
308	形成經擴增的封包	3 3 6 4 7
310	增量 R	R = 4
312-316	調整視窗	L = 1
316	在[L..R)視窗中形成填塞封包	1 3 1 5 5 0 2 4 7 1 6 14 3 3 6 4 7 0
322-338	生成經編碼的封包	
324 或 328	生成種子	第一=1、種子=0
326 或 330	生成係數	0 0 1
332	設定第一為 0	第一=0
334	產生封包的一線性組合	3 3 6 4 7 0
336	傳送(L, R, r, 數據)	1 4 0 3 3 6 4 7 0
338	設定 M := M + 速率	M = 4.000000
324 或 328	生成種子	第一=0、種子=503
326 或 330	生成係數	10 13 6
332	設定第一為 0	第一=0
334	產生封包的一線性組合	9 6 8 2 13 10
336	傳送(L, R, r, 數據)	1 4 503 9 6 8 2 13 10
338	設定 M := M + 速率	M = 4.500000

第七圖

步驟	描述	數據
308	形成經擴增的封包	4 2 1 6
310	增量 R	R = 5
312-316	調整視窗	L = 2
316	在[L..R)視窗中形成填塞封包	2 4 7 1 6 14 3 3 6 4 7 0 4 2 1 6 0 0
322-338	生成編碼的封包	
324 或 328	生成種子	第一=1、種子=0
326 或 330	生成係數	0 0 1
332	設定第一為 0	第一=0
334	產生封包的一線性組合	4 2 1 6 0 0
336	傳送(L, R, r, 數據)	2 5 0 4 2 1 6 0 0
338	設定 M := M + 速率	M = 5.000000
324 或 328	生成種子	第一=0、種子=305
326 或 330	生成係數	14 10 9
332	設定第一為 0	第一=0
334	生成封包的一線性組合	0 1 12 3 1 11
336	傳送(L, R, r, 數據)	2 5 305 0 1 12 3 1 11
338	設定 M := M + 速率	M = 5.500000

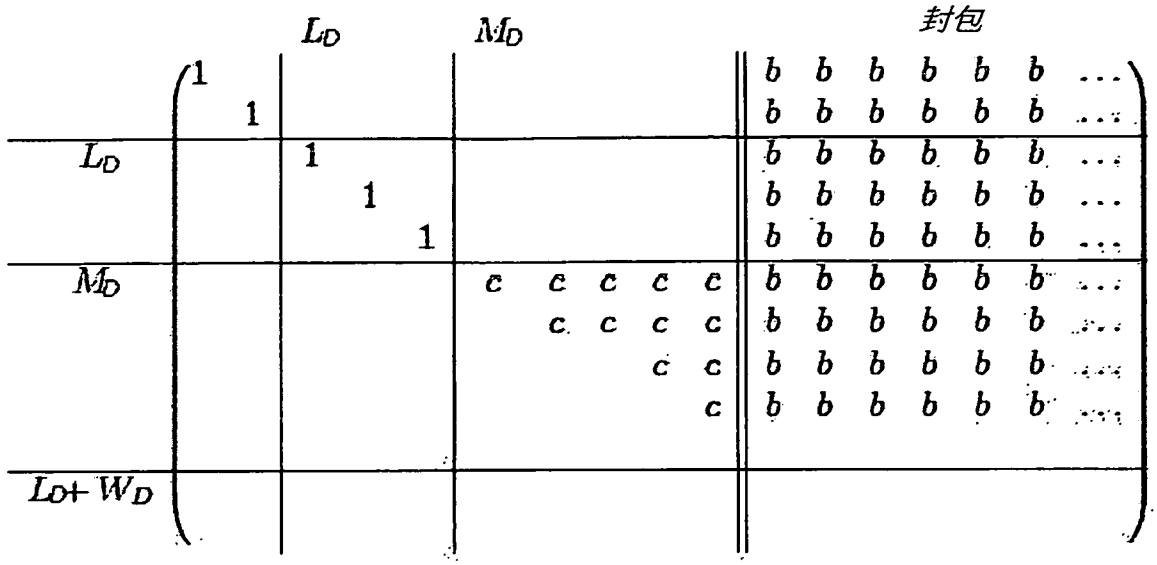
第八圖

+	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	3	2	5	4	7	6	9	8	11	10	13	12	15	14
2	2	3	0	1	6	7	4	5	10	11	8	9	14	15	12	13
3	3	2	1	0	7	6	5	4	11	10	9	8	15	14	13	12
4	4	5	6	7	0	1	2	3	12	13	14	15	8	9	10	11
5	5	4	7	6	1	0	3	2	13	12	15	14	9	8	11	10
6	6	7	4	5	2	3	0	1	14	15	12	13	10	11	8	9
7	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
8	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
9	9	8	11	10	13	12	15	14	1	0	3	2	5	4	7	6
10	10	11	8	9	14	15	12	13	2	3	0	1	6	7	4	5
11	11	10	9	8	15	14	13	12	3	2	1	0	7	6	5	4
12	12	13	14	15	8	9	10	11	4	5	6	7	0	1	2	3
13	13	12	15	14	9	8	11	10	5	4	7	6	1	0	3	2
14	14	15	12	13	10	11	8	9	6	7	4	5	2	3	0	1
15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

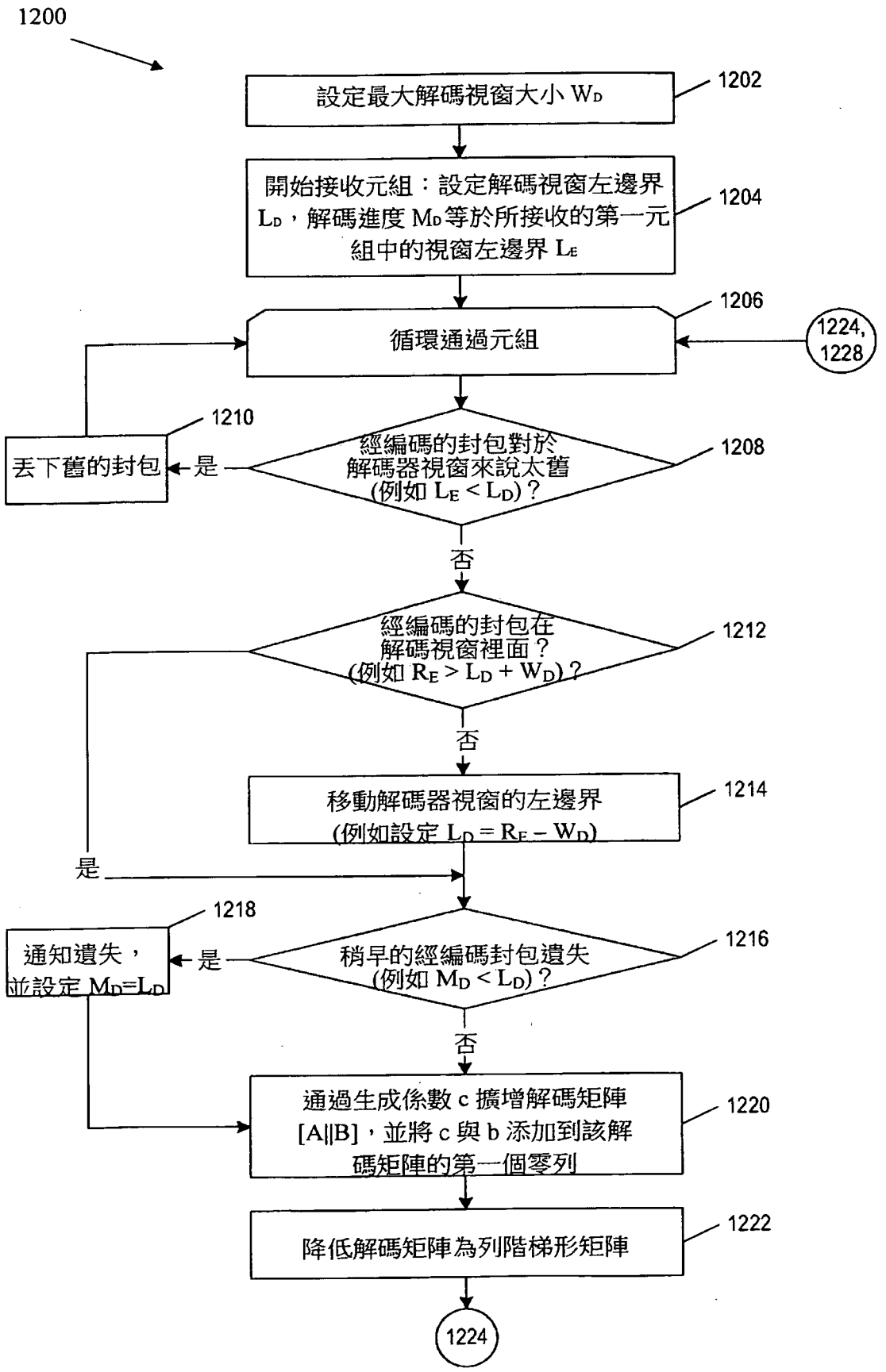
第九圖

×	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	0	2	4	6	8	10	12	14	3	1	7	5	11	9	15	13
3	0	3	6	5	12	15	10	9	11	8	13	14	7	4	1	2
4	0	4	8	12	3	7	11	15	6	2	14	10	5	1	13	9
5	0	5	10	15	7	2	13	8	14	11	4	1	9	12	3	6
6	0	6	12	10	11	13	7	1	5	3	9	15	14	8	2	4
7	0	7	14	9	15	8	1	6	13	10	3	4	2	5	12	11
8	0	8	3	11	6	14	5	13	12	4	15	7	10	2	9	1
9	0	9	1	8	2	11	3	10	4	13	5	12	6	15	7	14
10	0	10	7	13	14	4	9	3	15	5	8	2	1	11	6	12
11	0	11	5	14	10	1	15	4	7	12	2	9	13	6	8	3
12	0	12	11	7	5	9	14	2	10	6	1	13	15	3	4	8
13	0	13	9	4	1	12	8	5	2	15	11	6	3	14	10	7
14	0	14	15	1	13	3	2	12	9	7	6	8	4	10	11	5
15	0	15	13	2	9	6	4	11	1	14	12	3	8	7	5	10

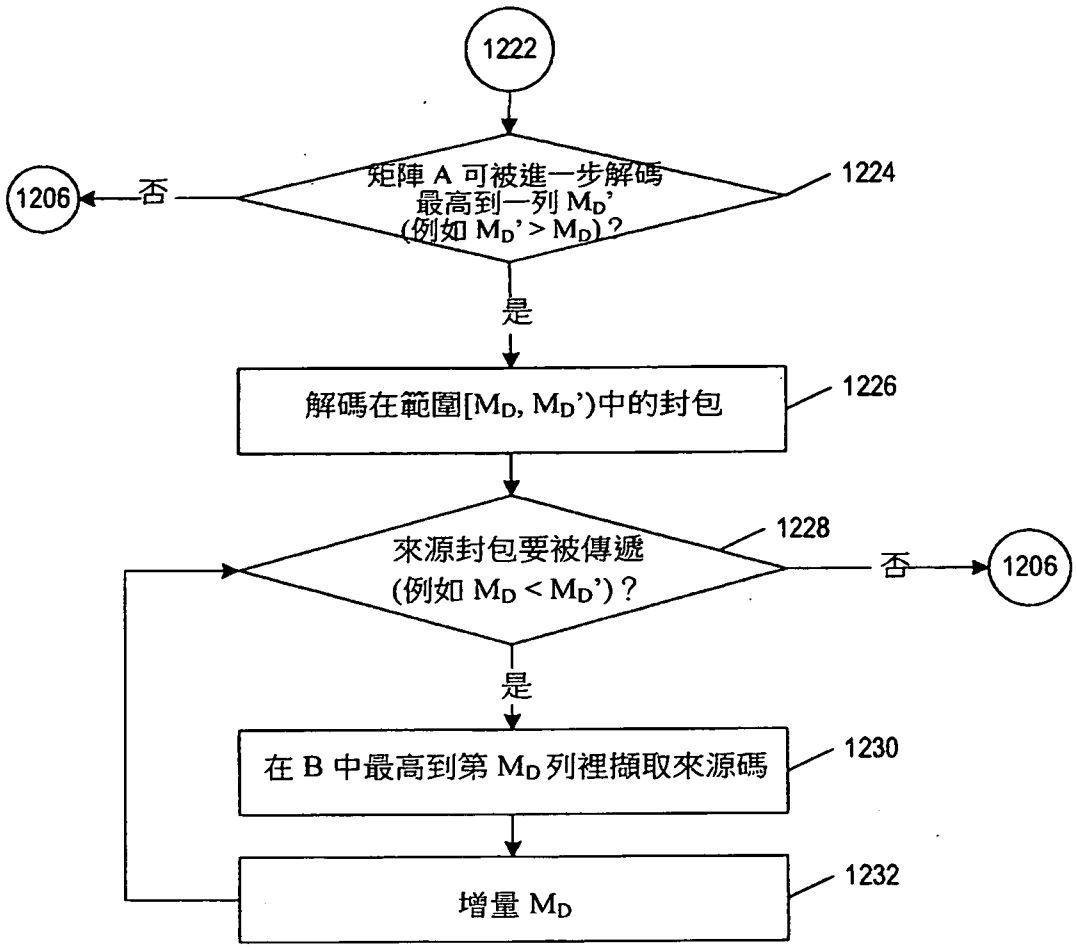
第十圖


L_D
M_D
封包
1
1
L_D
1
1
1
M_D
c c c c c
c c c c
c c
c
L_D+W_D
b b b b b b ...
b b b b b b ...
b b b b b b ...
b b b b b b ...
b b b b b b ...
b b b b b b ...
b b b b b b ...
b b b b b b ...

第十一圖


1200
設定最大解碼視窗大小 W_D
1202
開始接收元組：設定解碼視窗左邊界 L_D，解碼進度 M_D 等於所接收的第一元組中的視窗左邊界 L_E
1204
循環通過元組
1206
1224, 1228
經編碼的封包對於解碼器視窗來說太舊 (例如 $L_E < L_D$)？
1208
是
丟下舊的封包
1210
否
經編碼的封包在解碼視窗裡面？ (例如 $R_E > L_D + W_D$)？
1212
否
移動解碼器視窗的左邊界 (例如設定 $L_D = R_E - W_D$)
1214
是
稍早的經編碼封包遺失 (例如 $M_D < L_D$)？
1216
是
通知遺失，並設定 $M_D = L_D$
1218
否
通過生成係數 c 擴增解碼矩陣 [A||B]，並將 c 與 b 添加到該解碼矩陣的第一個零列
1220
降低解碼矩陣為列階梯形矩陣
1222
1224

第十二圖


1222
矩陣 A 可被進一步解碼
最高到一列 M_D'
(例如 $M_D' > M_D$)？
1224
否
1206
是
解碼在範圍[M_D, M_D')中的封包
1226
來源封包要被傳遞
(例如 $M_D < M_D'$)？
1228
否
1206
是
在 B 中最高到第 M_D 列裡擷取來源碼
1230
增量 M_D
1232

第十三圖

步驟	描述	數據
1220	擴增解碼矩陣	13 7 0 0 | 3 11 11 11 13 12
1222	降低為列階梯形矩陣	
1224	解碼矩陣	13 7 0 0 | 3 11 11 11 13 12

第十四圖

步驟	描述	數據
1220	擴增解碼矩陣	13 7 0 0 | 3 11 11 11 13 12 0 0 1 0 | 3 3 6 4 7 0
1222	降低為列階梯形矩陣	
1224	解碼矩陣	13 7 0 0 | 3 11 11 11 13 12 0 0 1 0 | 3 3 6 4 7 0

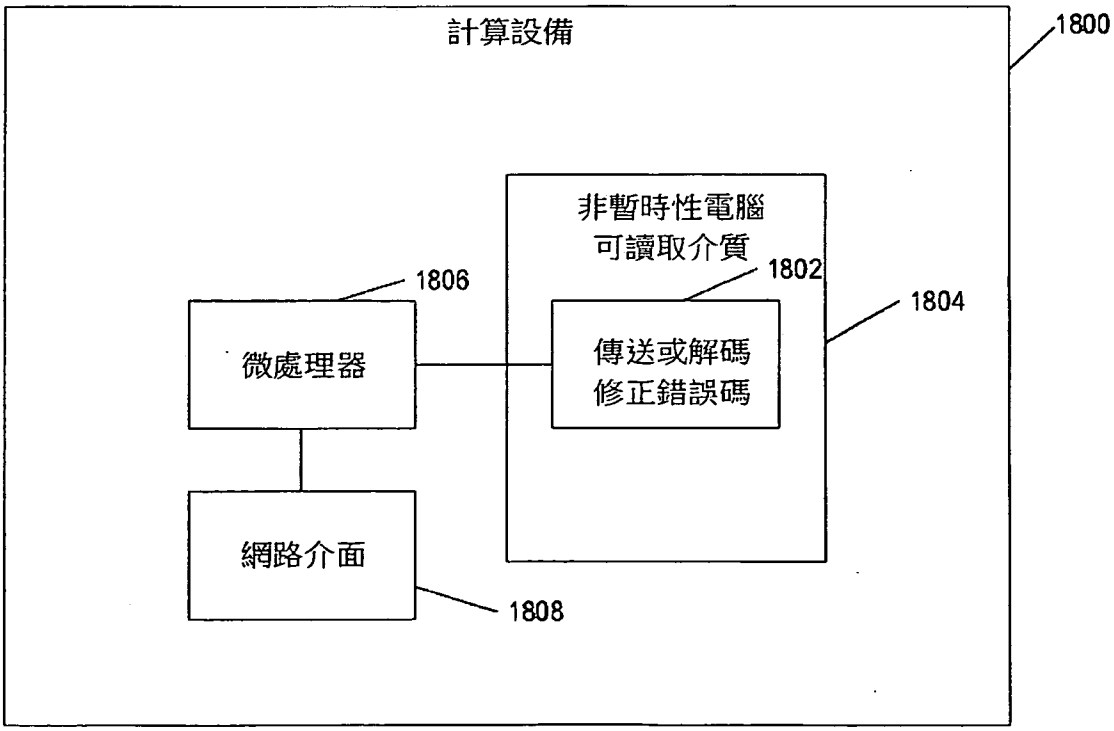
第十五圖

步驟	描述	數據
1220	擴增解碼矩陣	13 7 0 0 | 3 11 11 11 13 12 0 0 1 0 | 3 3 6 4 7 0 10 13 6 0 | 9 6 8 2 13 10
1222	降低為列階梯形矩陣	使用 A(0,0)來清除 A(2,0) 交換 A(1,:)和 A(2,:)
1224	解碼矩陣	13 7 0 0 | 3 11 11 11 13 12 0 1 6 0 | 8 14 0 10 7 14 0 0 1 0 | 3 3 6 4 7 0
	列[0 – 3)現在可解碼	縮放 A(0,0) 減去 A(2,2)以清除 A(1,2) 減去 A(1,1)以清除 A(0,1)
1226	解碼矩陣	1 0 0 0 | 1 3 1 5 5 0 0 1 0 0 | 2 4 7 1 6 14 0 0 1 0 | 3 3 6 4 7 0
1230	傳遞分組 1、長度 3	1 5 5
1230	傳遞分組 2、長度 4	7 1 6 14
1230	傳遞分組 3、長度 3	6 4 7

第十六圖

步驟	描述	數據
1212-1214	調整解碼矩陣	L_D 1 改變為 2
1220	擴增解碼矩陣	1 0 0 0 | 1 3 1 5 5 0 0 1 0 0 | 2 4 7 1 6 14 0 0 1 0 | 3 3 6 4 7 0 0 0 0 1 | 4 2 1 6 0 0
1222	降低為列階梯形矩陣	
1224	解碼矩陣	1 0 0 0 | 1 3 1 5 5 0 0 1 0 0 | 2 4 7 1 6 14 0 0 1 0 | 3 3 6 4 7 0 0 0 0 1 | 4 2 1 6 0 0
	列[3 – 4)現在可解碼	
1226	解碼矩陣	1 0 0 0 | 1 3 1 5 5 0 0 1 0 0 | 2 4 7 1 6 14 0 0 1 0 | 3 3 6 4 7 0 0 0 0 1 | 4 2 1 6 0 0
1230	傳遞封包 4、長度 2	1 6

第十七圖



第十八圖