國立中央大學資訊工程學系

資料壓縮期末報告

LZMA與壓縮已壓縮的圖片

梁中瀚

資工三

目錄

1. 前言
2. LZMA壓縮後格式簡介

(一)、header內部構造

(二)、LZMA Compressed data

1. lzma、jpeg與先壓縮成jpeg再放到lzma壓縮Lena.raw的比較

(一)、方法

(二)、數據比較

(三)、結果

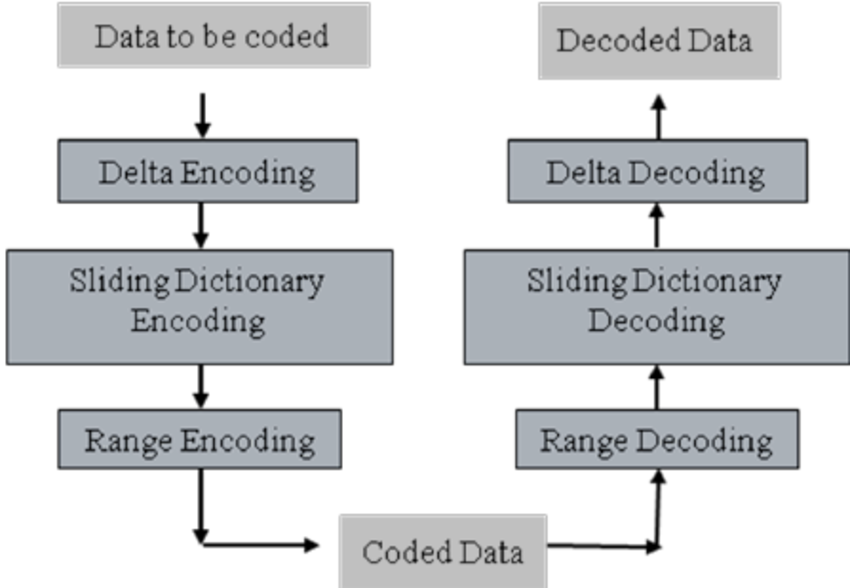
一、前言

Lzma為7z壓縮格式所支援的壓縮演算法，而7z又是近期大多數人會使用到的壓縮工具，並且大多數人都是直接將檔案放到壓縮檔裡面壓縮，實際測試先將圖片壓縮一次後，再放到lzma壓縮，與只用一次圖片壓縮的方式比較，觀察其檔案的大小。

Lzma是基於lz77的壓縮演算法，並且結合range coding(類似arithmetic encoding，但是可以用非二進位來壓縮)。

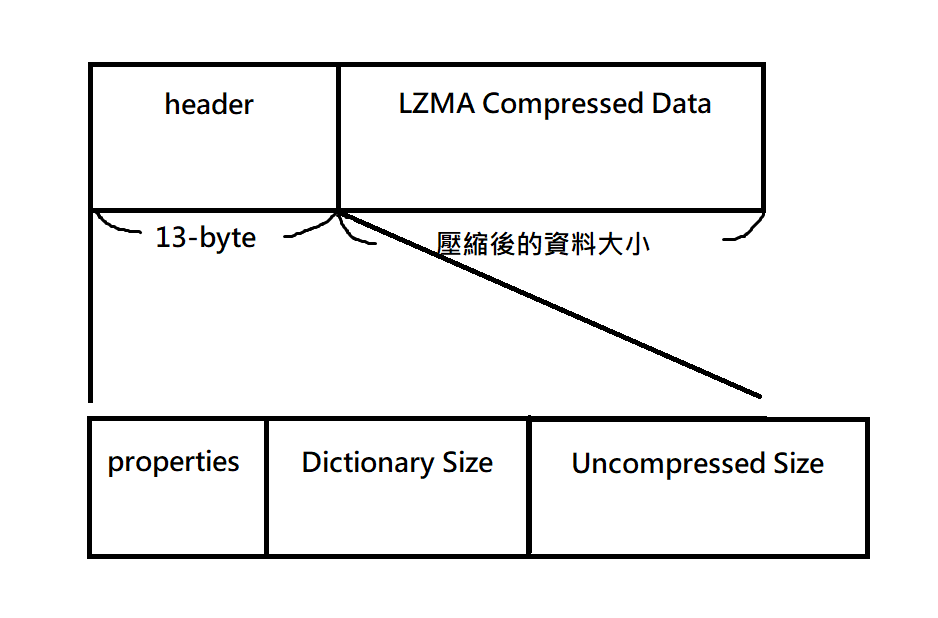
LZMA與lz77的不同點在於Lzma會記錄下不斷用到的dist長度，讓encoder不會一直紀錄重複的資料，並且會用很大的dictionary(window)紀錄近期input的字串

壓縮流程:



二、Lzma的壓縮後格式簡介:

.lzma檔案就是被lzma壓縮後的資料格式，它包含了兩個部分，header以及LZMA Compressed Data，如下圖。



(一)、header內部構造:

1. Properties:

Properties內部由三個變數所組成:

1. lc: literal context的bits數量，範圍是[0, 8]

之前的byte當中有多少是1的bit會拿來當作context(前後文)。

1. lp: literal position的bits數量，範圍是[0, 4]
2. pb: postion的bits數量，範圍是[0, 4]

最後用這個式子把以上三個變數encode:

Properties = (pb \* 5 + lp) \* 9 + lc

Decode就會用以下的sudo code來達成:

pb = properties / (9 \* 5);

properties -= pb \* 9 \* 5;

lp = properties / 9;

lc = properties - lp \* 9;

2. Dictionary Size:

用unsigned 32-bit little endian integer來儲存字典大小，所以總共是4個byte。

但是為了提供好的移植性(不同硬體)，應該只用2^n或者是2^n+2^(n-1)這兩種size。

3. Uncompressed Size:

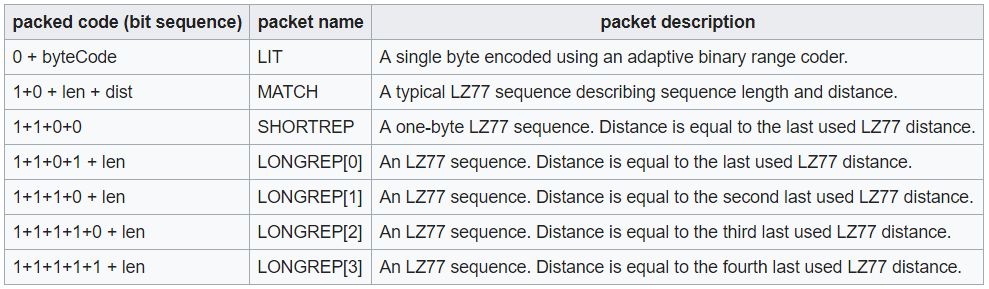
未壓縮大小是用unsigned 64-bit little endian integer來儲存的。

比較特別的是，如果64個bit都是1的話，代表未壓縮大小是未知的。

(二)、LZMA Compressed Data

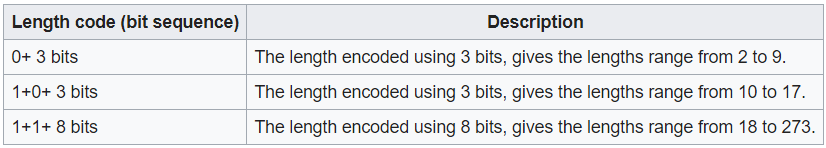
經由LZMA壓縮過後的資料會存放在這個區塊。

壓縮後的形式為bit stream，並且用 adaptive binary range coder做encode，而bit stream會被劃分成很多packets，packets的形式如下:

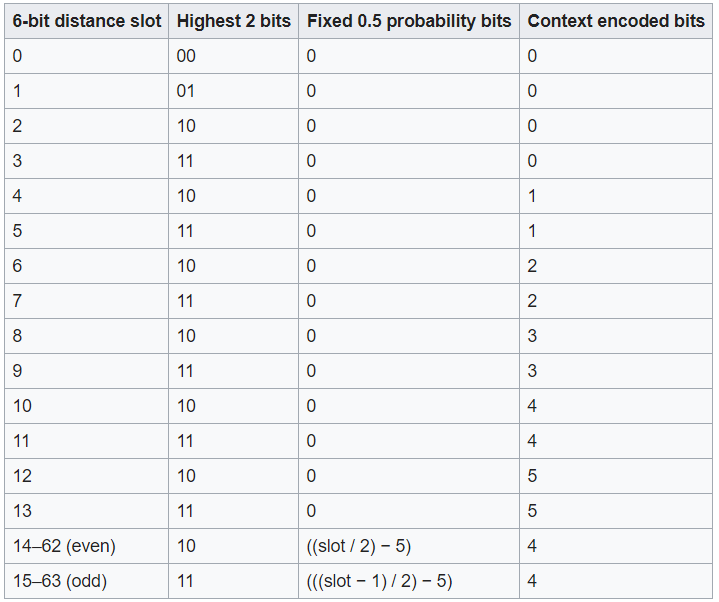


1. LIT: literal，代表單獨的byte並且用adaptive binary range coder encode，新出現的character。
2. MATCH: 標準的LZ77格式，有長度(在window找到look ahead buffer的code的長度)還有距離(look ahead buffer的code出現在window的哪裡)
3. SHORTREP: 一個byte的LZ77序列，距離等同於上一個LZ77使用的距離
4. LONGREP[n]: 一個LZ77的序列，並且距離等同於上n個LZ77的距離

長度的encode用以下的表格:



Distance用以下表格編碼:



每一個distance都會用6-bit distance slot作為起始， 6-bit distance slot存的數字就是後面會再接多少bit的意思。

LZMA在壓縮的時候要從dictionary快速地尋找match，會使用

三、lzma、jpeg與先壓縮成jpeg再放到lzma壓縮Lena.raw的比較

(一)、方法

Jpeg壓縮直接用irfanview把Lena.raw存檔成jpeg檔案並且把畫質調到100(最高)，png壓縮也是用irfanview，而lzma壓縮採用的是python預設的套件”lzma”，最後再將兩者的檔案大小計算出來。

(二)、數據比較

原始檔案大小(.RAW): 262144 bytes

壓縮後檔案大小

(1) jpeg: 154679 bytes (有損)

(2) png: 224248 bytes

(3) lzma: 180133 bytes (無損)

(4) jpeg再做lzma並存成.lzma格式: 156012 bytes

(5) jpeg再做lzma並存成.7z格式: 156138 bytes

(5) png再做lzma並存成.lzma格式: 223698 bytes

(6) png再做lzma並存成.7z格式: 223853 bytes

(三)、結果

得到的結果是jpeg壓縮完之後再用lzma壓縮反而讓檔案越壓縮越大，而png壓縮完之後再用lzma壓縮則能降低少許檔案大小。

而使用官方提供的解壓縮cpp檔案，發現python壓縮的預設設定是，dictionary size一律是8388608=2^23，並且lc=3、lp=0、pb=2

而存成.7z格式後檔案變大應該是因為標頭檔變大而導致壓縮完後的檔案變大了。

Code:

用python lzma套件壓縮圖片

<https://github.com/Louislar/DataCompression_finalReport>

參考:

[1]: <https://dev.twsiyuan.com/2018/06/how-to-compress-and-decompress-gamesaves-in-unity.html>

[2]: <http://html.rhhz.net/BJHKHTDXXBZRB/20150302.htm#R-4>

[3]: <https://gautiersblog.blogspot.com/2016/08/lzma-compression.html>

[4]: <https://www.slideshare.net/LevanChelidze/7-zip-compression-settings-guide>