

LRU AS DICTIONARY REPLACEMENT POLICY IN NEAR-LOSSLESS LZW IMAGE COMPRESSION

Macarie Breazu, Antoniu Pitic

University "Lucian Blaga" of Sibiu, Computer Science Department,
Emil Cioran 4, Sibiu, Romania
email: macarie.breazu@ulbsibiu.ro, antoniupitic@ulbsibiu.ro

Abstract: Classical image compression methods are based on error measuring at entire image level only. In some areas there is an obvious need for getting an upper bound for the error at pixel level. In the paper we analyze such a near-lossless method based on LZW dictionary algorithm and evaluate the LRU solution for dealing with the moment when dictionary becomes full. The changes needed to adapt LZW to become a near-lossless method are also presented. As far as we know our approach is the first attempt to use LZW as a near-lossless method. Experimental results obtained and presented in the paper prove that the LRU solution gives better than the solutions based on dictionary freezing or clearing or on quadtree partitioning.

Keywords: image, compression, near-lossless, quadro, LZW, LRU.

1. INTRODUCTION

Today, in the age of computers and communication, there is an obvious need for data compression and, especially, for image compression. Because the compression methods where no error is accepted (called lossless) gives poor results on images, we have to accept some error in the decompressed image in order to get much higher compression ratios. Those methods (called lossy) become very popular especially with the multimedia development.

In the classic approach the criteria that is minimized by the image compression methods is:

$$E_1(R, O) = \sum (R_{ij} - O_{ij})^2$$

where R_{ij} represents the restored (decompressed) image, O_{ij} represents the original (uncompressed) image and the sum is made over the entire image. Here we are interested on the error at entire **image level**. The error expression corresponds to the classic Euclidean distance measure.

Another class of applications is the one where it is very important to get an upper limit on the error at each pixel. Now the criteria to be minimized is:

$$E_2(R, O) = \max |R_{ij} - O_{ij}|$$

where R_{ij} and O_{ij} have the same meanings and the maximum is searched also over the entire image. Here we are interested on the error at **pixel level**. The error expression to be minimized corresponds to the Cebisev distance measure.

The first approach (based on E_1) is the classic one and a lot of research has been done during years on that topic. The most popular method is the one based on DCT and stated in the JPEG standard. The main drawback of the scheme is that we cannot have any guarantee about the error at pixel level. In some applications (satellite images, medical images, etc.), there is a need for some guarantees at pixel level because the artifacts introduced by the compression methods are unacceptable (without an upper limit), especially for further automatic processing. Therefore, in such area image compression was very rarely accepted (or not accepted at all).

In the last years the interest for such compression methods (based on E_2) is growing quickly and, therefore, the area requires a lot of research (Weinberger et al., 1998; Vleuten R.J., 2001; Yamauchi et al., 2001). Because usually we are

interested only in small values for the acceptable error limit ($\pm 1, \pm 2, \dots, \pm 10$) the method is called **near-lossless (NL)**. Certainly, if the acceptable error level becomes 0 we get the classical lossless compression.

It is very interesting to observe that such a small change, only in distance measure, will put us in front of a new area, all the old results (based on E_1) being now irrelevant (for E_2).

It is proven that the task of finding the best representation (and compression) within the acceptable error $\pm k$ is a NP-complete task.

This paper reflects our recent interest in NL compression area and continues the work published in (Breazu et al., 2003, Breazu et al., 2004) where only freezing and clearing were taken into consideration.

2. QUADRO PARTITIONING METHOD

One of the methods very popular in image processing is quadro (quadtree) partitioning (Salomon, 1998). It consists in dividing and representing a square image area by its 4 quadrants. The process is repeated recursively until the square area is uniform and then is represented by that value. The main advantages of the method are that it is very simple (implementation is almost always recursive), it is adaptive to the image and the amount of extra information needed to store the partitioning information (only 1 bit for describing a partitioning decision) is very low comparing to other partitioning methods. Even if its main application is adaptive partitioning, the method is popular also as a representation (and compression) method.

If the initial image is not square the image is decomposed in squares and the method is applied individually on each of them.

3. LZW COMPRESSION ALGORITHM

LZW is the most popular algorithm from the LZ family. It was proposed by Lempel, Ziv and modified to this form by Terry Welch (Welch, 1984). As in all dictionary-based methods, a dictionary of previously encoded strings is kept. The size of the dictionary is usually in the area between 512 (index represented on 9 bits) and 16384 (index represented on 14 bits). Initially, the first 256 entries are completed with the 0-255 symbols.

Considering "x" being the current input stream character and "I" the current string, the encoding process can be stated as follows:

```
I = empty
while not end of input
  read next character x
  if Ix exists in dictionary
```

```
    I = Ix
  else
    emits index of I to the decoder
    add Ix to the dictionary
  I = x
```

The decoder operates accordingly. The only difference is that, when receiving an index and adding Ix to the dictionary, the symbol x is, for the moment, unknown. It will be known as the first symbol of the next encoded string.

The main advantage of the LZW method over other LZ-family methods is that there is no need to transmit to the decoder anything else than the dictionary index of the string (not even the new character). Even if it was not specifically designed for image compression, it can easily be used on images by considering the images a stream of bytes (for example by scanning the image line by line).

In our previous work, presented in (Breazu et al., 2004), we have studied two solutions for the moment when the dictionary becomes full: freezing (when the dictionary is kept in future as is, no other dictionary updates are made) and clearing (when the dictionary is put back into the initial empty state). Freezing was proved to be better. Now we propose to use the LRU solution for the moment when the dictionary becomes full – replacing the dictionary entry that was least recently used and keeping all the others. The LRU is widely used both in hardware (computer architecture) and software (operating systems – (Silberschatz et al., 1998)).

4. NEAR-LOSSLESS EXTENSIONS

In order to use them in NL environments the previous methods have been adapted accordingly.

Regarding the QUADRO method we have to change the criteria used to stop the partitioning process. The original one, of stopping only when the entire remaining image square is identical, is replaced by the one of stopping when the difference between the maximum and the minimum values inside the image block is less or equal twice the accepted error. In that case the partitioning process is stopped and the entire block is represented by the mean of the minimum and the maximum values.

Regarding the LZW method we have to search in the dictionary for the longest existing entry that meets the maximum accepted error requirements. Different from the original LZW method, now we can find more than one such entry. In that case a decision of choosing one of them has to be taken. We select always the entry that, for the same maximum accepted error, presents the smallest mean error among the bytes of the entry. If there are still more such entries, we pick the first one. The resulted method is labeled in the following chapters NL-LZW.

Table 1 – Results for QUADRO method

Accepted error (Err)	CR	Accepted error (Err)	CR
0	0,968	6	2,677
1	1,035	7	3,107
2	1,216	8	3,511
3	1,482	9	3,982
4	1,839	10	4,472
5	2,308		

An interesting aspect is that, when adding an entry to the decoder's dictionary, the real value of the next character is not known exactly (but only within the accepted error level). So, the coder and decoder dictionaries are not in synchronism! This is not really a problem (as it might look at first sight) because the search process is done only at the coder where the dictionary is based on exact values. Certainly, at decoding, different data streams will be decoded (based on different dictionaries) but this is OK, the error is in the accepted level.

Even if LZW was previously used in a lossy purpose (Pigeon, 2001) (the lossy aspect come only from restricting the dictionary search) our approach is the first attempt to use LZW as a near-lossless compression method.

Table 2 – Experimental results in Compression
Ratio CR for NL-LZW

Accepted error (Err)	Freezing			
	Dictionary size			
	9 bits	10 bits	11 bits	12 bits
	NL-LZW 9 F	NL-LZW 10 F	NL-LZW 11 F	NL-LZW 12 F
0	0,977	0,999	1,089	1,165
1	1,187	1,412	1,459	1,542
2	1,478	1,653	1,796	1,900
3	1,784	1,986	2,153	2,287
4	2,041	2,273	2,448	2,589
5	2,376	2,616	2,800	2,921
6	2,523	2,867	3,037	3,189
7	2,789	3,177	3,364	3,597
8	3,000	3,422	3,623	3,782
9	3,277	3,682	3,937	4,177
10	3,482	4,155	4,274	4,507

Accepted error (Err)	LRU			
	Dictionary size			
	9 bits	10 bits	11 bits	12 bits
	NL-LZW 9 LRU	NL-LZW 10 LRU	NL-LZW 11 LRU	NL-LZW 12 LRU
0	0,993	1,007	1,093	1,167
1	1,220	1,318	1,466	1,551
2	1,516	1,680	1,820	1,935
3	1,842	2,055	2,238	2,378
4	2,104	2,387	2,591	2,729
5	2,469	2,731	2,972	3,121
6	2,663	3,046	3,275	3,440
7	2,993	3,452	3,700	3,942
8	3,203	3,687	3,997	4,275
9	3,492	4,039	4,405	4,715
10	3,650	4,454	4,822	5,126

5. EXPERIMENTAL RESULTS

In order to test the proposed NL-LZW method we have compared it to the QUADRO method. The tested methods are the ones described in the previous paragraph. Experiments have been done on the standard 8-bit grayscale 512*512 LENA image. The

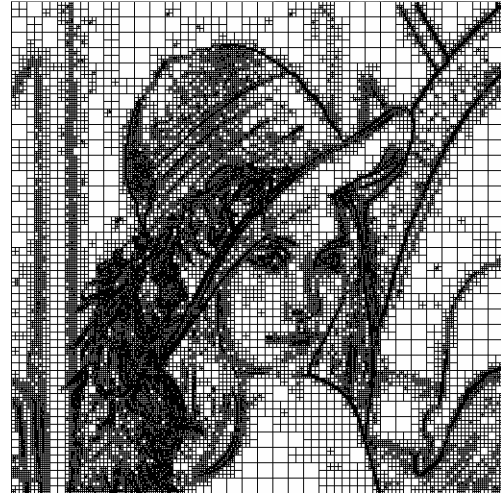


Fig. 1 Quadro partitioning



Fig. 2 Decompressed image for QUADRO method



Fig. 3 Decompressed image for NL LZW method

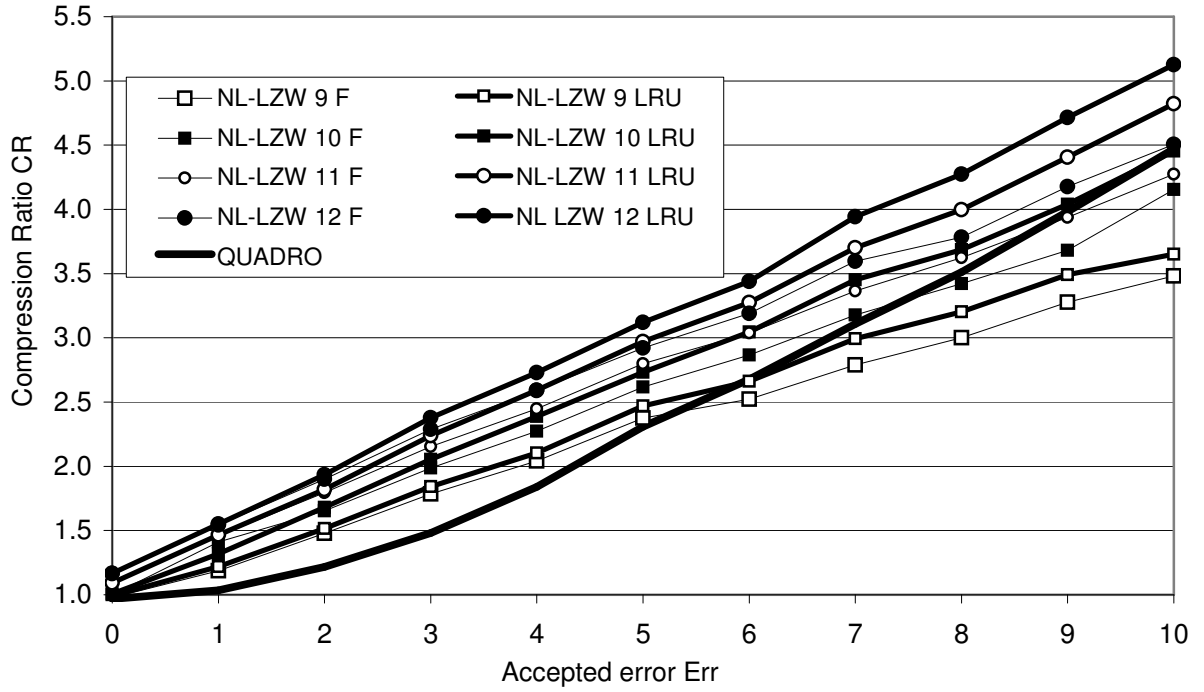


Fig.4 Rate-distortion curves obtained experimentally

accepted error is considered in the 0-10 domain (as usual in the near-lossless image compression papers).

For the QUADRO method we have considered the maximum size of the block being 16. Accepting a larger block size will overload the compressed stream with partitioning bits without any compression ratio (CR) improvement. Experimental results (not presented here) have proven that 16 is the best option to be taken. The values for the obtained CR are listed in Table 1.

For the NL-LZW method the CR results are presented in Table 2. Regarding the dictionary size, the numbers of 512, 1024, 2048 and 4096 entries are considered (corresponding to their index representation on 9 to 12 bits). In (Breazu et al., 2004) we have proven that the “Freezing” solution gives better CR results than the “Clearing” solution so we present here for comparison only the “Freezing” results and LRU results (and labeled F and LRU accordingly).

To visually inspect the decompressed images we present in Fig. 1 and Fig. 2 the image partitioning

and the decompressed image for the QUADRO method and in Fig. 3 the decompressed image for the NL-LZW 10 LRU method. In both methods the level of accepted error considered is 10 (for smaller values the image is visually identical with the original and useless for direct comparison). The presented CR’s are almost the same (4.47 vs. 4.45). Even in that case, when the CR advantage of our proposed NL-LZW method is gone (as we shall notice later), we notice that the decompressed image looks clearly better, without suffering from the block artifacts of the QUADRO method.

Compression times were measured and a few are presented in Table 3 (time not being a very important criteria, results are presented briefly). Surprisingly, for the LRU case, the time costs are the same (and not greater as it could be expected). This fact proves that the searching for the replaceable entry is insignificant compared to the regular search in the dictionary.

The rate-distortion curves corresponding to Table 2 are presented in Fig. 4. We notice that:

- By increasing the size of the dictionary the NL-LZW method performances are always increased (both for freezing and LRU methods).
- The NL-LZW method always outperforms the QUADRO method for small levels of accepted error – where in fact the near-lossless methods are of maximum interest. For the 12 bits case the NL-LZW 12 LRU gives better for the entire tested area.

Table 3 – Experimental results in Compression Time (seconds) for NL-LZW

Method	Accepted error (Err)				
	0	2	4	7	10
NL-LZW 9 F	48	33	26	20	17
NL-LZW 9 LRU	47	32	25	20	17
NL-LZW 12 F	223	147	115	95	85
NL-LZW 12 LRU	222	146	115	95	86

- LRU gives always better results than Freezing and the gain is greater as the dictionary size grows.

The good results of LRU method can be explained by the fact that it solves one of the most common problems in computer science (and in other areas as well): the *stability-plasticity* dilemma. Freezing the dictionary is an extreme stability solution while clearing the dictionary is an extreme plasticity solution, both having obvious disadvantages. LRU is, in fact, an adaptive stability-plasticity solution.

6. CONCLUSIONS AND FUTURE WORK

We have proposed a new method based on the use of the well-known lossless LZW method in a near-lossless compression scheme – as far as we know an original approach.

Results presented in the paper prove that the presented NL-LZW method outperforms the QUADRO method for small levels of the accepted error, exactly in the range of maximum interest for near-lossless image compression domain. Even more, our method did not suffer from the block artifacts that appear in the quadro-based method.

The LRU replacement policy has been proposed and evaluated. Experimental results prove the superiority of the LRU method compared to the other solutions considered. The good results can be explained by the fact that LRU is an adaptive solution to the stability-plasticity dilemma.

Certainly, further research has to be done, especially dealing with:

- Testing other solutions for the moment when the dictionary becomes full, i.e. using an LFU-like policy (Least Frequently Used) for finding a replaceable entry in the dictionary.

- Optimizing the encoding program in order to allow us to test our method for larger dictionaries (i.e. represented on 13 bits or more), not done yet due to computational requirements.
- Other decisions that can be taken into consideration when choosing the best dictionary entry for the current input stream. The most interesting one is not to consider strictly the longest entry but the one that gives us (considering also the future data stream) the smallest number of strings (entries) coded.

REFERENCES

- Breazu M., Pitic T., Volovici D., Brad R., 2003 - *Near-lossless LZW image compression*, SINTES 10 Conference, Craiova, Romania
- Breazu M., Pitic T., Volovici D., Sima D., 2004 - *Dictionary clearing vs. freezing in near-lossless LZW image compression*, CONTI 2004 Conference, Timisoara, Romania
- Pigeon S., 2001 - *An Optimizing Lossy Generalization of LZW*, Data Compression Conference (DCC '01), Snowbird, Utah, USA
- Salomon D., 1998 - *Data Compression*, Springer Verlag
- Silberschatz A., Galvin P., 1998 - *Operating System Concepts, Fifth Edition*, Addison Wesley
- Vleuten R. J., 2001 - *Low-Complexity Lossless and Fine-Granularity Scalable Near-Lossless Compression of Color Images*, Data Compression Conference (DCC '02), Snowbird, Utah, USA
- Weinberger M., Seroussi G., Sapiro G., 1998 - *The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS*, Computer Systems Laboratory, HPL-98-193
- Welch, T.A., 1984 - *A technique for high-performance data compression*, IEEE Computer 17(6)
- Yamauchi M., Wakatani A., 2001 - *A New Lossless Compression Scheme for Medical Images by Hierarchical Segmentation*, Data Compression Conference (DCC '01), Snowbird, Utah, USA