Advances in Knowledge-Based and Intelligent Information and Engineering Systems
M. Graña et al. (Eds.)
IOS Press, 2012

doi: 10.3233/978-1-61499-105-2-1676

Efficient Wavelet Sign Prediction: Simulated Annealing vs Genetic Algorithms

J.M. Navarro[1], P. Moreno[2], F. Rodríguez-Ballester[3], A. Martí[3], M.A. Cruz-Chávez[2], M.P. Malumbres[1], and O. López[1]

[1] Miguel Hernandez University, Av. Universidad s/n, 03202 Elche, Spain
{jmnavarro90}@gmail.com, {mels,otoniel}@umh.es
[2] CIICAp-Universidad Autonoma Estado de Morelos
Av. Universidad 1001. Col. Chamilpa,
62209 Cuernavaca Morelos, Mexico
{pmoreno,mcruz}@uaem.mx
[3] Universidad Politécnica de Valencia
c/ Camino de Vera s/n 46222 Valencia - Spain
{prodrig,amarti}@disca.upv.es

Abstract. Wavelet transforms have proved to be very powerful tools for image compression, since many state-of-the-art image codecs employ DWT into their algorithms. One advantage of this transform is the provision of both frequency and spatial localization of image energy compacted into a small fraction of the transform coefficients, equally likely to be positive or negative. Previous studies have verified that there is a strong correlation between the sign of a wavelet coefficient and the signs of their neighbors. This correlation opens the possibility of using a sign predictor in order to improve the image compression process. In this work we evaluate two algorithms, one based on Genetic programming and other based on Simulated Annealing process in order to obtain a good wavelet sign predictor.

Keywords: sign coding, discrete wavelet transforms, image coding, simulated annealing, genetic algorithms

1 Introduction

In this work we are looking for optimal/suboptimal solutions of the particular problem related with wavelet image compressors. This kind of image compressor is based in the use of a mathematical transform called Discrete Wavelet Transform (DWT). Wavelet transforms have proved to be very powerful tools for image compression, since many state-of-the-art image codecs, including the JPEG2000 standard [1], employ DWT into their algorithms. One advantage of the wavelet transform is the provision of both frequency and spatial localization of image energy. The image energy is compacted into a small fraction of the transform coefficients and compression can be achieved by coding these coefficients. The energy of a wavelet transform coefficient is restricted to non-negative

real numbers, but the coefficients themselves are not, and they are defined by both a magnitude and a sign. Shapiro stated in [2] that a transform coefficient is equally likely to be positive or negative and thus one bit should be used to encode the sign. In recent years, several authors have begun to use context modeling for wavelet sign coding [3][1][4], showing that despite the equiprobability of wavelet sign values, some sign correlation can be found among wavelet coefficients, resulting in overall compression ratio improvements.

In a previous work [5] we have observed that the sign of a wavelet coefficient may be strongly correlated with the sign of some neighbor coefficients. However, this relationship is not uniform and constant for any image, or even consistent within the same image. Thus, although a careful analysis for the target image could be done in order to get the most accurate sign relationships and therefore better sign prediction and improved compression rates, these sign relationships would be only useful for this image. By increasing the number and kind of images under analysis, the relationship between the signs of the neighbor coefficients may be generalized. But the problem that arises is twofold. On the one hand, a thorough evaluation of all possible combinations for a large number of images can present a high computational cost. On the other hand, it is possible that, when increasing the number of images, some relationships become contradictory. That is, the sign for a combination of neighbors in an image is the opposite for the same combination of neighbors in another image.

Genetic algorithms (GA) were first introduced by Holland in [6] and they are nowadays well known techniques for finding nearly optimal solutions of very large problems and also, they have been used in image processing [7][8]. In a genetic algorithm, the evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated by means of a cost function that determines the optimal degree we are looking for (i.e compression rate). Multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

On the other hand, Simulated annealing (SA) is a global search optimization method which makes use of stochastic variation to avoid terminating in local extrema [9], and has proved extremely useful in locating the global extrema of objective, or cost functions derived from complex nonlinear systems. The technique was first introduced by Kirkpatrick [10]. The method is an adaptation of the Metropolis-Hastings algorithm [11] to generate sample states of a thermodynamic system. It has since been applied to many diverse problems in a wide range of disciplines, including the derivation of optimal quantization parameters for JPEG coding [12]. The method introduces a statistical guarantee that the global optimum for a given problem will be found [13].

In this paper, we will compare a genetic algorithm with a simulated annealing algorithm that efficiently predict the wavelet coefficient signs based on the correlation found in a given neighborhood set. We will compare the performance of both algorithms in terms of convergence, computational time and memory requirements.

The remainder of the paper is organized as follows: in Section 2 we define the optimization problem and propose both GA and SA algorithms that match the problem definition. Section 3, we compare both algorithms in terms of convergence time, computational resources as well as the solution goodness. Finally, in Section 4 some conclusions are drawn.

2 Wavelet Sign Prediction: Problem Statement

To estimate sign correlation in a practical way, we have applied a 6-level Dyadic Wavelet Transform decomposition of the source image. as Deever assesses in [4] the sign neighborhood correlation depends on the subband type (HL,LH,HH). In a previous work [5], we used three different neighbors depending on the subband type. So, for HL subband, the neighbors used are N, NN and W. Taking into account symmetry, for the LH subband, those neighbors are W, WW, and N. For the HH subband they are N, W, and NW, exploiting the correlation along and across the diagonal edges. This lead us to a maximum of 3^3 Neighbor Sign Patterns (NSP) for each subband type.

C	N	NN	W	Occurrences	%Probability
+	+	+	+	13	20.31
+	+	+	-	8	12.50
-	-	-	+	8	12.50
-	+	+	+	6	9.38
-	-	+	+	6	9.38
Others				23	35.93

Table 1. Probability distribution of neighbor sign patterns (NSPs) of HL_6 subband ($8x8$ coefficients) in Lena image

In Table 1 we show the NSP probability distribution for HL_6 subband (from the sixth decomposition level) of Lena test image. As shown, the probability that the current coefficient (C) is positive when its N, NN and W neighbors are also positive is around 20%. Besides, if the N and NN neighbors have the same sign and the W neighbor has the opposite sign, the current coefficient (C) has the opposite sign of its W neighbor with a probability of 25% as shown in rows two and three in Table 1. The visible sign neighborhood correlation suggest that the sign bits of wavelet coefficients are compressible.

After running the genetic or simulated annealing algorithm for each subband type, we obtain the sign prediction table that contains the sign predictions for

every $NSP[k]$, $k = 0 \ldots 3^{neighbors} - 1$. So, when coding the sign of a wavelet coefficient in a particular subband, first we will get the sign value of the corresponding neighbor set in order to form the actual NSP. Then we will look up this NSP in the table to find the sign prediction of the actual wavelet coefficient. Finally, what we are going to encode is the correctness of this prediction, i.e., a binary valued symbol resulting from expression $\hat{SC}_{i,j}[k] \cdot SC_{i,j}$, where $SC_{i,j}$ represents the sign value of actual coefficient $C_{i,j}$ (0: positive, 1: negative) and $\hat{SC}_{i,j}[k]$ represents the sign prediction of coefficient $C_{i,j}$. The performance of a binary entropy encoder will depend on the behavior of our sign predictor, the higher the success prediction ratio the higher the compression rate.

In the first place we need to map our problem to the definitions and processes that define both a GA and a SA as introduced above.

2.1 Genetic Algorithm Definition

The genetic algorithm created follows the classic structure for this type of algorithms: population initialization, population evaluation and new population generation through the classification of individuals using a fitness function and the use of crossover and mutation operators to create new individuals.

Then we need to create a population (universe) of individuals that during the evolution process will improve their goodness in base to a fitness function that will determine their quality. The genetic information of one individual is the set of sign prediction values of every $NSP[k]$.

For our purposes we will define the fitness function in such a way that its result indicates the sign prediction performance of one individual for the set of images. In other words, the fitness function will estimate the compression rate of the sign prediction encoding that would be achieved if the prediction table defined by this individual is used to encode the images in the image set (see Eq. 1).

Finally, single point crossover is used in all variations where the locus point to split the parent gene is randomly selected. On the other hand, the mutation policy inverses the prediction value of a randomly chosen gene. Also, two best individuals survive to the next generation and they can not be modified by the mutation operator.

Each individual is represented by a binary vector where its elements represent a combination of signs from a predefined neighborhood set of coefficients, and the stored values into these elements correspond to the sign prediction for the coefficient (binary value). The size of this vector depends on the number of neighbors that conforms the neighborhood. The greater the number of neighbors considered, the greater the number of sign combinations, namely 3^n being n the number of neighbors, since the possible sign values of neighbor wavelet coefficients are ternary values (positive, negative or null).

For the test, we have used an initial population of 100 individuals, and the evolution process is performed 1000 times.

2.2 Simulated Annealing Algorithm Definition

The SA algorithm is based on an analogy with metallurgical annealing, a process whose objective is to cause a solid to reach a minimal energy state with a highly regular crystalline structure. Basically, the analogy is formed as follows: a) A method for generating random changes in the state space is defined, b) The objective (cost) function of the problem is equivalent to energy in the mechanical process, c) A control parameter is defined ('temperature') which is lowered in accordance with an annealing schedule and d) An acceptance distribution is defined, based on a suitable physical law which determines, at a given 'temperature', whether a given generated state, which does not lead to a lower cost, is to be accepted.

The simulated annealing approach is able to avoid local optimum by means of the inclusion of an acceptance test. This test allows non-optimal solutions to be accepted with a Boltzmann probability function [9].

The simulated annealing algorithm created follows the classic structure for this type of algorithms: First of all we define an initial solution (R), containing a sign prediction for each 3^n NSPs being n the number of neighbors, then each NSP sign prediction is randomly initialized as a positive or negative sign and finally, we evaluate the goodness of the solution by means of the cost function (Eq. 1) over all selected images.

$$\frac{\sum_{i=0,j=0}^{N,M} \sum_{k=1}^{3^n NSP} \hat{SC}_{i,j}\,[k] \cdot SC_{i,j}}{N \times M} \forall\, image \tag{1}$$

where N, M are the image dimensions, $\hat{SC}_{i,j}\,[k]$ is the sign prediction for NSP (k) and $SC_{i,j}$ is the sign of wavelet coefficient $C_{i,j}$. The division by $N \times M$ is performed to normalize the cost function because the different evaluated images could be of different sizes.

Then, during cooling process, SA algorithm attempts to replace the current solution (R) by a new generated solution (R') in which one of the NSP sign predictions is changed. The new solution (R') will be accepted if its cost value (cost function) is better than the current solution (R) one. Also, in case the new solution cost value is not improved, the new solution (R') may then be accepted with a probability that depends both on the difference between the corresponding cost values and also on a global parameter T_0 (called the initial temperature), that is gradually decreased during the process.

Several parameters should be taken into account in the simulated annealing algorithm: The temperature cooling control parameters (T_0, T_f) defined by a decrement parameter (β) and the number of NSPs changed on the next simulated annealing iteration of new solutions (R'). We have performed tests varying these parameters to tune the algorithm in such a way that the algorithm convergence to the optimal/suboptimal solution is fast enough. The parameters used to obtain the sign prediction are: $T_0(5)$, $T_f(2)$, $\beta(0.98)$ and number of NSP changed on each generation of random solutions (1).

3 Performance Evaluation

In order to analyze the performance of both GA and SA algorithms, we have run both algorithms over three different scenarios.

- **S1**- $512x512$ Lena image (low textured image)
- **S2**- $512x512$ Barbara image (high textured image)
- **S3**- Both Lena and Barbara images.

Furthermore, we have defined 3 different neighborhood patterns varying the number of neighbors to predict the wavelet coefficient sign (see Table 2).

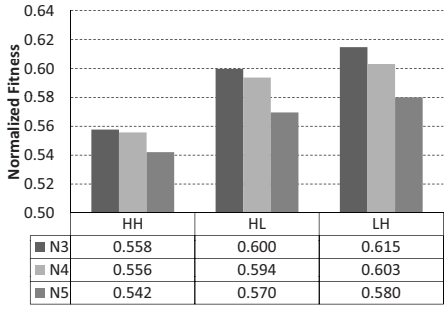
	Neighborhood size (n)	**HL**	**LH**	**HH**	**NSPs** 3^n
N3	3	N,NN,W	W,WW,N	N,W,NW	27
N4	4	N,NN, W,WW	W,WW, N,NN	N,W, NW,NNWW	81
N5	5	N,NN, NNN,W,WW	W,WW WWW,N,NN	N,W,NW, NNWW,NNNWWW	243

Table 2. Neighborhood tested for HL, LH, and HH wavelet subbands.

In Fig. 1 we show a comparison between GA and SA for scenarios S1, S2 and S3 using N1, N2 and N3 neighborhood. As shown, both GA and SA algorithms obtain the same fitness value in all scenarios when applied over a neighborhood with 3 neighbors (N3). However, SA algorithm obtains better fitness as the number of neighbors increases, while the GA algorithm get worse as the number of neighbors increases in all scenarios. Note that the better the fitness, the greater the compression performance will be achieved.

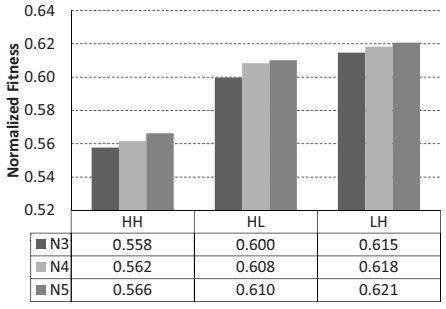
Regarding computational cost, both algorithms performs an intensive search and the main computational cost is due to the fitness function which is computed for all pixels ($N \times M$) of the image and for all evaluated images. In Fig. 2 we show the computational time in seconds required by both GA and SA algorithms until the convergence to the maximum fitness value obtained by each algorithm is reached. As it can be seen, SA algorithms quickly converges to the maximum fitness value for N3 and N4 neighborhood setting, being on average 4.2 times and up to 8.4 times as fast as the GA algorithm for N3 and N4 neighborhood pattern, respectively. However, for N5 pattern, the GA algorithm converges 1.8 times on average as fast as the SA algorithm. This is mainly due to the greater number of iterations performed by the SA algorithm in our test. The SA algorithm performs the NSP size (243 for the N5 pattern) for each temperature value, being the total iterations performed in our test 6318, while the GA algorithm performs only 1000 iterations.

As Table 3 shows, both algorithms have similar memory requirements, being nearly the total amount of it needed to store the sign of wavelet coefficients sign and directly affected by the number of images to be evaluated and its dimensions.



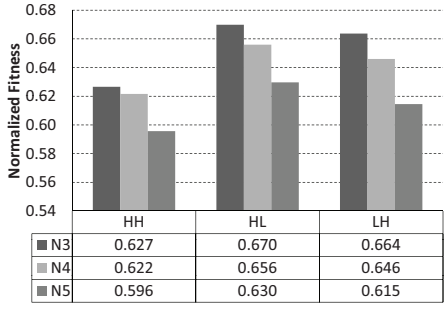
	HH	HL	LH
N3	0.558	0.600	0.615
N4	0.556	0.594	0.603
N5	0.542	0.570	0.580

(a) Genetic algorithm - S1



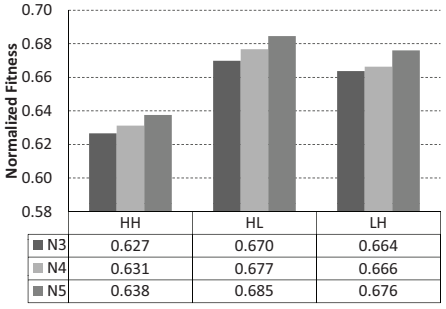
	HH	HL	LH
N3	0.558	0.600	0.615
N4	0.562	0.608	0.618
N5	0.566	0.610	0.621

(b) Simulated Annealing - S1



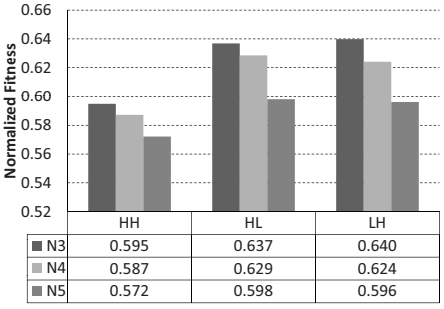
	HH	HL	LH
N3	0.627	0.670	0.664
N4	0.622	0.656	0.646
N5	0.596	0.630	0.615

(c) Genetic algorithm - S2



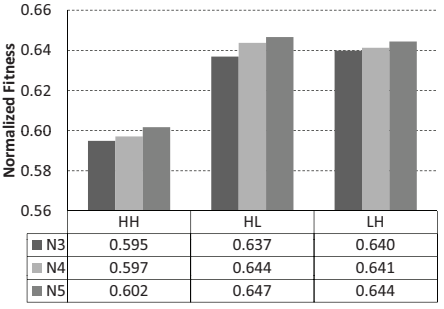
	HH	HL	LH
N3	0.627	0.670	0.664
N4	0.631	0.677	0.666
N5	0.638	0.685	0.676

(d) Simulated Annealing - S2



	HH	HL	LH
N3	0.595	0.637	0.640
N4	0.587	0.629	0.624
N5	0.572	0.598	0.596

(e) Genetic algorithm - S3



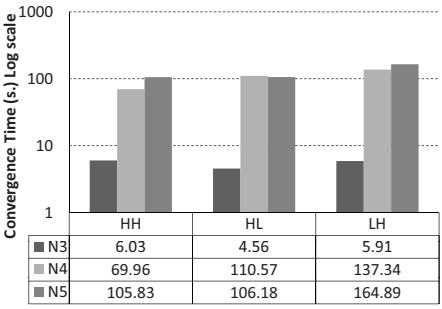
	HH	HL	LH
N3	0.595	0.637	0.640
N4	0.597	0.644	0.641
N5	0.602	0.647	0.644

(f) Simulated Annealing - S3

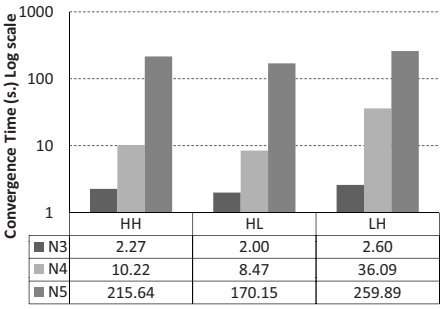
Fig. 1. Fitness comparison between GA and SA for S1, S2 and S3 scenarios using N3, N4 and N5 neighborhood pattern.

4 Conclusions

We have presented an evaluation between a genetic algorithm and a simulated annealing algorithm that performs a prediction for wavelet coefficient signs. Both algorithms are able to obtain good predictors which will be used by an image encoder to compact the wavelet sign information. Between them, the SA algo-



(a) Genetic algorithm - S3



(b) Simulated Annealing - S3

Fig. 2. Time elapsed until convergence to the final fitness value for GA and SA algorithms (log scale)

	Pixels	Genetic Algorithm	Simulated Annealing
S1	262144	1732	1724
S3	524288	2764	2756

Table 3. Memory requirements in KBytes for GA and SA algorithms over different scenarios.

rithm performs better as the number of neighbors is increased using the same configuration parameters. On the other hand, the GA algorithm must be tuned specifically to each scenario, being its behavior slightly lower in the test performed. Regarding computational cost, the SA algorithm is faster than the GA algorithm, because usually converges quickly than the GA algorithm, being up to 15 times as fast as the GA algorithm in the test performed. So, we think that SA algorithm is the better choice for this particular problem.

Acknowledgments. Thanks to Spanish Ministry of Education and Science under grants TIN2011-27543-C03-03 and TEC2010-11776-E for funding.

References

1. Taubman, D.: High performance scalable image compression with EBCOT. IEEE Transactions on Image Processing **9**(7) (July 2000) 1158–1170
2. Shapiro, J.: A fast technique for identifying zerotrees in the EZW algorithm. Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing **3** (1996) 1455–1458
3. Wu, X.: High-order context modeling and embedded conditional entropy coding of wavelet coefficients for image compression. In: Proc. of 31st Asilomar Conf. on Signals, Systems, and Computers. (1997) 1378–1382
4. Deever, A., Hemami, S.S.: What's your sign?: Efficient sign coding for embedded wavelet image coding. In: Proc. IEEE Data Compression Conf., Snowbird, UT. (2000) 273–282

5. Lopez, O., Martinez, M., Pinol, P., Malumbres, M., Oliver, J.: E-LTW: An enhanced ltw encoder with sign coding and precise rate control. In: Image Processing (ICIP), 2009 16th IEEE International Conference on. (nov. 2009) 2821 –2824
6. Holland, J. In: Adaption in Natural and Artificial Systems. University of Michigan Press (1975)
7. Chabrier, S., Rosenberger, C., Emile, B., , Laurent, H.: Optimization-based image segmentation by genetic algorithms. EURASIP Journal on Image and Video Processing **2008** (2008) 1–10
8. Anam, S., Islam, M.S., Kashem, M., Islam, M., Islam, M., Islam, M.: Face recognition using genetic algorithm and back propagation neural network. In: International MultiConference of Engineers and Computer Scientists, Hong Kong (2009)
9. Aarts, E., Korst, J. In: Simulated Annealing and Boltzmann Machines. John Wiley and Son (1989)
10. Vecchi, M., Kirkpatrick, S.: Global wiring by simulated annealing. IEEE Trans. Computer-Aided Design **CAD-2** (1983) 215–222
11. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E.: Equation of State Calculations by Fast Computing Machines. The Journal of Chemical Physics **21**(6) (1953) 1087–1092
12. Sherlock, B., Nagpal, A., Monro, D.: A model for jpeg quantization. In: Speech, Image Processing and Neural Networks. (1994) 176–179
13. Ingber, L.: Simulated annealing: Practice versus theory. Mathematical and Computer Modelling **18**(11) (1993) 29–57