Improving the Simulated Annealing Algorithm for the Index Assignment Method to Enhance the Robustness of Communication Systems

Tran Ngoc Tuan, Nguyen Quoc Trung
School of Electronics and Telecommunications, Hanoi University of Science and Technology
Email: tuan.tranngoc@hust.edu.vn, trung.nguyenquoc@hust.edu.vn

***Abstract* - Improving the performance of communication systems is one of the issues attracting the close attention by the researchers. Problems of improving the information systems are how to inherit the existing systems, in which only few modules will be upgraded without any effect over other modules and doing so will help to save expenses and has the high feasibility. In this article, we study the methods of optimizing the Index Assignment (IA), a joint source channel coding (JSCC) method to enhance the interference resistance of the communication systems. IA techniques have been effectively applied into communication systems using the vector quantization (VQ) technique, a widely-used lossly data compression technique for coding and transmitting high correlation signals such as speech, image and video [13]. We propose an improved algorithm based on Simulated Annealing (SA) algorithm to increase the speed and efficiency in optimizing the combination of source and channel coding of IA method. This has been proved by experiments.**

Keywords* - *Index Assignment, Joint Source-Channel Coding, Simulated Annealing, Speech Coding, Vector Quantization.

1. INTRODUCTION

Shannon's separation theorem of source and channel coding states that the source coding (data compression) and the channel coding (error protection) can be performed separately without loss in the optimality of the system [1]. However, this theorem is true only if both transmitter and receiver are permitted to have an unlimited complexity and delay, which is unrealistic for any practical applications. Under the realistic constraint of limited complexity, it is advantageous to design the source and channel codes jointly, as witnessed by a large body of literature on joint source channel coding (JSCC).

Index Assignment (IA) is a simple feasible JSCC method, which is considered in this work. It is possible to reduce the sensitivity of the bit stream to errors without adding redundant bits, simply by carefully allocating codewords to the signal parameter values. IA method can be employed as a method for improving the existing system without any effect or change over other modules in digital communication system (e.g. modulation). The task is thus to find the index assignment which minimises the expected coding error. In this work, we apply this JSCC approach for communication systems using vector quantization (VQ), a common technique to compress signals with high correlation (e.g. speech, audio, image, video...).

An n bit quantiser has $N!$ ($N = 2^n$) possible codeword assignments. To test $N!$ assignments is a NP-hard problem, which makes practically impossible finding an optimal solution for codebooks larger than 32 entries. For this reason, a number of different IA approximate solutions have been proposed. Zeger and Gersho proposed the binary switching algorithm (BSA) to improve the codevector IA [2]. However, the BSA is the descent algorithm and it always converges to a local minimum. Several algorithms have been proposed to overcome local minimum traps and reached better results than that of BSA algorithm. The Simulated Annealing (SA) algorithm is applied to design the codevector indices by Farvardin [3]. An improvement of SA algorithm name ISA was proposed by Bouzid and Djeradi [14]. The Tabu

search approach was developed for codeword IA by Pan and Chu [7]. Then, Modified Tabu Search (MTS) algorithm has been proposed [8]. Moreover, evolutionary algorithms have been studied and applied to IA problems such as Ant algorithm [10], Parallel Genetic Algorithm (PGA) [6], Evolutionary Algorithm Based Index Assignment Algorithm (EAIAA) [4], Immune Colon Algorithm (MCIAA) [5]. These prior methods have usually been based on assuming a Binary Symmetric Channel (BSC) and/or a quantiser with a mean-square-error distortion measure, which is not always true for real applications.

Of these algorithms, SA is the most effective and popular algorithm with fast convergence time, wide use in optimization problems in general and issue of JSCC in particular [11,12] and has been continuously improved [14,15]. However, this algorithm still has several drawbacks (stated in Section 4.1), especially in the case of large searching space. In this paper, we have proposed an improved algorithm based on SA algorithm for the general IA problem in order to reduce the running time of the algorithm and increase the optimization degree of results. The feasibility and efficiency of the proposed algorithm is confirmed by experimental results.

The remaining of the paper is organized into 6 sections. In Section 2, Vector Quantization technique is briefly presented with a focus on the Index Assignment problem. Section 3 describes the SA algorithm applied to the IA problem. We then discuss the limitations of the SA algorithm and propose an improved SA algorithm in Section 4. The performance of the new algorithm is envaluated by simulations in Section 5. Finally, Section 6 contains the conclusion of this paper.

2. VECTOR QUANTIZATION AND INDEX ASSIGNMENT PROBLEM

2.1. Vector Quantization.

When a set of discrete-time amplitude values is quantized jointly as a single vector, the process is known as Vector Quantization (VQ), also known as block quantization or pattern-matching quantization. VQ encodes each vector from a sequence of source vectors with a channel symbol – a binary word chosen from a finite set. A typical VQ system contains a finite predetermined collection of codevectors (a codebook), and a vector distortion measure which, when given two vectors, yields a distance (or distortion) between them. At the encoder, the input vector is compared to each codevector in order to find the closest match and a binary index is transmitted to the decoder in order to inform about the selected codevector.

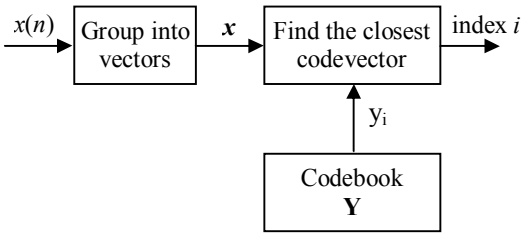

Fig.1 Block diagram of a vector quantizer

The codebook design process is also known as trainning or populating the codebook. A well known algorithm for VQ codebook design is the Linde-Buzo-Gray (LGB) algorithm [9].

A vector quantizer Q of dimension M and size N can be defined as a mapping of M-dimensional Euclidean space R^M into a finite subset Y containing N vectors of R^M given by:

$$Q: R^M \rightarrow Y$$
$$x \rightarrow y_i \quad (1)$$

The codebook $Y = \{ y_i ; 1 \le i \le N \}$ is the set of M-dimensional codevectors, also known as reconstruction vectors or quantization vectors. The output of the vector quantizer is the index i of the codevector y_i which satisfies:

$$i = \underset{k}{\operatorname{argmin}}\, d\left(x, y_k\right) \quad (2)$$

where $d(x,y_k)$ is the nonnegative distance between two vectors and a widely used distortion measure is the squared Euclidean distance, given by:

$$d\left(x, y\right) = \left\| x - y \right\|^2 \quad (3)$$

2.2. The Index Assignment Problem.

The channel noise will include channel errors in communication. The effect of channel errors is to

cause errors in the received indices which can result in significant distortion in decoded vectors.

The sum of possible distortions when transmitting vector c_i is:

$$D_i(b) = P(c_i)\sum_{j=1}^{N} P_c\left(b(j),b(i)\right) d\left(c_i,c_j\right) \quad (4)$$

The IA function b is a permutation of the integers $\{1,2,...,N\}$; $b(i)$ assigns an index to the i^{th} codevector. Let $P(c_i)$ and $d(c_i,c_j)$ denote the probability of sending codevector c_i and the distortion (or distance) between codevector c_i and c_j. P_c is the $N\times N$ matrix and $P_c(i,j)$ denote the probability that the index i is received given the index j is sent:

$$P_c(i,j) = P(i|j) \quad (5)$$

P_c can be obtained by simulation or theoretical model. If we assume the channel model is a binary symmetric channel (BSC) with bit error probability ε, P_c can be calculated as follow:

$$P_c(i,j) = \varepsilon^{h(i,j)}(1-\varepsilon)^{n-h(i,j)} \quad (6)$$

where $h(i,j)$ denote the Hamming distance between i and j, i.e., the number of bits in which i and j differ.

The overall distortion is given by:

$$D\left(b\right) = \sum_{i=1}^{N} P(c_i)\sum_{j=1}^{N} P_c\left(b(j),b(i)\right) d\left(c_i,c_j\right) \quad (7)$$

Different index assignments do not change the distortion of the source code, but they do affect the overall distortion of a communication system $D(b)$ in case of channel error. Therefore, index assignment can be optimized with respect to channel statistics to mitigate the impact of channel error. The IA problem is to find the best codebook rearrangement b which minimize $D(b)$. $D(b)$ is also called the objective function in combinatorial optimization problems.

3. SIMULATED ANNEALING FOR INDEX ASSIGNMENT

Simulated annealing (SA) is an intuitive while effective algorithm to search for a good approximation of the global minimum in a large search space. It has been widely applied to designing good codes in the literature. The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution in order to escape from local minima. The probability of doing such a move is decreased during the search.

The SA algorithm consists of the following steps:

Input parameters: P, P_c and d.

Control parameters: T_0, T_{min}, Th_1, Th_2, Th_3.

Step 1: Initialization.

- Set $T = T_0$ as a sufficiently high temperature.
- Choose the initial solution b.

Step 2: Randomly choose a neighbour solution $b¢$ by swaping two indices in b.

- Set $\Delta D = D(b) - D(b')$
- **If** $\Delta D < 0$ **then** set $b = b'$
  else set $b = b'$ with probability $e^{-\Delta D/T}$

Step 3: **If** the number of cost drops in step 2 exceeds Th_1 or the number of unsuccessful solutions is reached Th_2 occur **then** go to step 4 **else** return to step 2.

Step 4: Reduce slowly the temperature.

Step 5: **If** $T < T_{\min}$ or the number of iterations exceeds Th_3 **then** stop **else** return to step 2.

In step 4, several formula can be used to define the cooling scheme. We retain particularly [3]:

$$T_i = \alpha_T T_{i-1}; \quad 0 < \alpha_T < 1 \quad (8)$$

There exists several options for SA algorithm depending on the control parameter sets (T_0, T_{min}, Th_1, Th_2, Th_3 and α_T). These parameters have effects on the algorithm repetitions. When N is large, in order to make SA algorithm run effectively, it is necessary to increase the number of iterations and the quantity of approved solutions through each iteration (by increasing α_T, Th_1 and Th_2). If the number of iterations are not large enough, the results of the algorithm are even worse than that of BSA Algorithm (always converges at a local minimum point).

An improvement of SA algorithm (ISA algorithm) [14] is storing the best solution after each iteration. The final solution is the best solution of all iterations. Nevertheless, the improvement is only effective in the case the factor α_T is not large enough.

4. IMPROVEMENT APPROACH OF THE SA ALGORITHM FOR THE IA PROBLEM

4.1. Limitations of SA Algorithm.

Although the SA algorithm can jump out of local minima within limits and can find the global minimum or approximate global minimum, it also has limitations:

- The algorithm takes much time to compute $\Delta D = D(b') - D(b)$ in step 2. In order to evaluate the function $D(b')$ by eq (7), we need $2N^2$ multiplications and N^2 additions.

- In step 2, there is no regime of avoiding the coincidence when generating neighbours of the current solution b.

- When the temperature T is low, it is very hard to escape from local minimum trap [15].

- Results of SA algorithms are not guaranteed to be a local minima, especially when the number of iterations is few (factor α_T is not large enough) and N is large, then algorithm results are not equal to the results of BSA Algorithm.

4.2. Improvements of SA Algorithm for the IA problem.

In order to overcome the above-mentioned limitations, we propose improved SA Algorithm (MSA) used for the IA problem with the following improvements:

- ***The first improvement***: Reducing the computational time.

The overall distortion function $D(b)$ can be rewritten as:

$$D(b) = \sum_{i=1}^{N}\sum_{j=1}^{N} P_c\left(b(j), b(i)\right) d'(i,j) \qquad (9)$$

where the square matrix d' can be calculated as:

$$d'\left(c_i, c_j\right) = P(c_i) d\left(c_i, c_j\right) \qquad (10)$$

In step 2, the neighbour b' is resulted from the permutation of parameter pairs (r, s) in b, corresponding to the matrix P_c changed at 2 rows and 2 columns with the permutations $b(r)$ and $b(s)$. The difference ΔD can be computed as follows:

$$\begin{aligned}\Delta D = \sum_{k=1}^{N} \{ & d'(k,r)\left[P_c\left(b'(k), b'(r)\right) - P_c\left(b(k), b(r)\right)\right] \\ & + d'(r,k)\left[P_c\left(b'(r), b'(k)\right) - P_c\left(b(r), b(k)\right)\right] \\ & + d'(k,s)\left[P_c\left(b'(k), b'(s)\right) - P_c\left(b(k), b(s)\right)\right] \\ & + d'(s,k)\left[P_c\left(b'(s), b'(k)\right) - P_c\left(b(s), b(k)\right)\right]\}\end{aligned} \qquad (11)$$

With this computation, we need $4N$ multiplications and $8N$ additions. In comparision with the original algorithm, the time complexity to evaluate ΔD is reduced from $O(N^2)$ to $O(N)$ and the total computational time will be significantly reduced because step 2 is repeated many times.

- ***The second improvement***: Avoiding the coincidence when considering the neighbour b' in step 2.

Let $k_{\max}$ as denote the total neighbour of the solution b. We realize $k_{\max}$ is the number of ways of opting for index pairs (for interchanging) of N indices of b, therefore, $k_{\max}$ is a constant:

$$k_{\max} = C_N^2 = N(N-1)/2 \qquad (12)$$

For this reason, we can number index pairs from 1 to $k_{\max}$. Then, to create the random neighbour b' of the solution b, we will interchange the index pair k, in which k is a random ordinal number between 1 and $k_{\max}$.

In order to generate random neighbours of b without repeat, we need to generate unique random number k. Firstly, a random permutation of integers from 1 to k_{max} , *K_RAND*, is generated and then k is chosen alternatively by this permutation at each iteration (k = *K_RAND*(p) with p = 1,2,...$k_{\max}$). *K_RAND* will be regenerated whenever a new solution is accepted. With this approach, we can know when falling into the local minimum trap (when examining all $k_{\max}$ neighbours without finding out better solutions) in order to proactively escape from the local minima.

- ***The third improvement***: Optimizing result.

If the final results have not been the local minima, then the results can be optimized by executing hill

climbing algorithm to find the nearest local minimum. Examine all neighbours $b¢$ of the current solution, then select the neighbour with the smallest ΔD. The algorithm will be stopped if we cannot find any neighbours better than the current solutions ($\Delta D<0$). The hill climbing algorithm runs relatively fast because the results of above steps themselves are quite close to the local minimum points.

The below is MSA Algorithm:

Input parameters: P_c and $d¢$

Control parameters: T_0, T_{min}, α_T, Th_1, Th_2, Th_3.

Step 1: Initialisation.

- Number $k_{\max}$ index pairs (chosen from N indices from 1 to N). $k_{\max}$ is obtained by Eq(12).
- Set $T = T_0$ as a sufficiently high temperature.
- Choose the initial solution $b = b_{\text{init}}$.
- Set $p = 0$.

Step 2:

- **If** $p = 0$ **then** generate the random permutation K_RAND from *1* to $k_{\max}$.
- Set $p = p + 1$.
- **If** $p > k_{\max}$ **then** b is a local minimum. Select k randomly within $1 \le k \le k_{\max}$ and generate neighbour b'. Goto **ACCEPT** (escape from the local minimum).
- Set $k = K\_RAND(p)$.
- **If** $k = k_{old}$ **then** go to step 2 (repeated searching)
- Generate the neighbour b' by interchanging the index pair k in the solution b.
- Calculate $\Delta D = D(b) - D(b')$ by Eq. (11)
- **If** $\Delta D < 0$ **then** update the best solution b_{opt} and goto **ACCEPT else** goto **ACCEPT** with probability $e^{-\Delta D/T}$.

ACCEPT: Set $b = b'$; $k_{old} = k$; $k = 0$ (reset k).

Step 3, 4, 5: The same as in SA algorithm.

Step 6: If b_{opt} is a local minimum **then** STOP **else** improve the result solution by hill climbing algorithm to find the nearest local minimum.

In order to effectively promote SA and MSA algorithm, the number of iterations must be large enough in line with the size of the searching space. With the experiments, we have suggested a set of control parameters for SA and MSA algorithm corresponding to different common values of N as follows: $T_0 = 10$; $T_{\min} = 2.5\times10^{-5}$; and other parameters in Table 1.

Table 1. Parameters of the SA algorithm

N	k_{max}	α_T	Th_1	Th_2	Th_3
32	496	0.97	5	250	12000
64	2016	0.97	5	1000	50000
128	8128	0.98	5	4000	100000
256	32640	0.98	5	16000	800000

With the above improvements, the new algorithm will be optimized better than the original algorithm both in terms of speed and optimization degree of results. They will be proved by experiments in the next section.

5. EXPERIMENTS AND DISCUSSION

5.1. Experimental setup.

The previous works showed that the lower overall distortion of the IA solution $D(b)$ is, the better performance the communication system archieves. Hence, our experiments only focus on the optimization and the efficacy of different IA algorithms without any attention to the performance of the entire system.

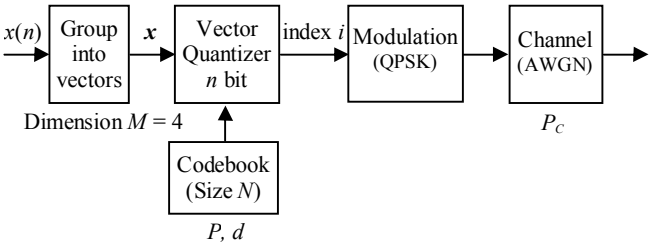

Fig.2 *Experimental model of communication system*

We consider a communication system where the source encoder/decoder is a vector quantizer and its experimental model is demonstrated in Fig.2. The message emitted from the source is first partitioned into vectors of dimension $M = 4$; the vector quantizer

is then used to compress the input vectors; after that the signals are modulated to transmitting over the channel by the QPSK modulator. The channel is assumed to be noisy, so the output of the channel is the sum of its input and noise which is modeled as Gaussian noise.

The VQ scheme is generally employed to effectively encode the signals with high correlation. The input signal tested is accordingly a highly correlated random source, that is, 1st-order Gauss-Markov process:

$$x(n) = \alpha x(n-1) + w(n) \quad (13)$$

where $\alpha < 1$ is a correlation coefficient and $w(n)$ is a zero-mean, unit variance, Gaussian white noise process. The value for α in our experiments is 0.9.

Applying LBG algorithm with splitting method [9] and training ratios (number of training vectors divided by N) of at least 1000, codebooks of N = 32, 64, 128, 256 codevectors are generated. The squared Euclidean distance measure is used for codebook training.

In order to implement IA algorithms, input matrices P, P_c and d are pre-computed. Probability matrix P is obtained by quantizing a vast majority of training vectors with given codebooks. The channel in this case can be modeled as a BSC channel, in which matrix P_c is obtained according to Eq (6) and the bit error rate ε for QPSK modulation over AWGN channel is given by:

$$\varepsilon = Q\left(\sqrt{\gamma_s}\right) \quad (14)$$

where $\gamma_s = E_S/N_0$ is the received channel signal-to-noise ratio (CSNR) per symbol. Matrices P_c in our experiments are all evaluated with γ_s = 5dB.

Experiments were carried out to test the speed and the performance of algorithms. The control parameters of SA and MSA algorithm are set as described in Section 4.2. Algorithms are executed in MATLAB and in the same computer.

5.2. Results and discussion

Experiment 1 compares the running time of the improved algorithm MSA with the original algorithm SA. Both algorihms were carried out under the same initial conditions, with different sizes of the codebook N = 32, 64, 128, 256. Each algorithm was executed 10 times with the same initial solutions. Average execution time in seconds of each algorithm with different N values is demonstrated in Table 2.

Table 2. *Average calculation time of two algorithms*

N	SA	MSA
32	6.39	0.98
64	40.98	4.22
128	533.62	31.47
256	3009.21	138.62

We can see that MSA algorithm significantly reduces the execution time, especially the larger N becomes, the more the running time of MSA algorithm is reduced compared with original SA algorithm. Furthermore, because the execution time is shorter, we can increase the number of iterations to find out more optimal results in the resonable time.

Table 3. *Performance comparison (overall distortion) of IA algorithms for 64 codevectors*

Trials	SA/ISA	MSA	BSA	TS	MCIAA
1	4.3020	**4.2610**	4.6331	4.3010	4.4698
2	4.2913	**4.2865**	4.3372	4.4676	4.3619
3	4.3082	**4.2633**	4.4152	4.3203	4.5142
4	4.2865	**4.2742**	4.3286	4.2987	4.3388
5	4.2955	4.2990	4.3438	**4.2680**	4.3483
6	4.2941	**4.2790**	4.3972	4.4829	4.3600
7	4.3213	4.3115	4.3460	**4.2896**	4.2949
8	4.3118	**4.2352**	4.4942	4.2938	4.3226
9	4.2868	**4.2643**	4.3843	4.3642	4.3623
10	4.2833	**4.2813**	4.3774	4.3207	4.3172
Average	4.2981	**4.2755**	4.4057	4.3407	4.3691

The experiment 2 was carried out to compare the performance of the SA algorithm, ISA algorithm, MSA algorithm and other algorithms (BSA [2], TS[7], MCIAA[5]) in the same condition for using in codevector index assignment for N = 64 codevectors. The number of iterations of all algorithms is about 500 and the other parameters are the same as in the corresponding references. Using the same group of input parameters (initial solution b_{init}, matrices P, P_c,

d), each algorithm is executed 10 times with 10 different initial solutions and the overall distortion of results are demonstrated in Table 3.

From the limited experiments, the MSA algorithm may reach the best results for most trials and obtains the lowest average value of all solutions. Meanwhile, since the factor α_T is near 1 and threshold parameters of SA algorithms are large enough, the results of SA and ISA algorithm are almost the same. On the other hand, the MSA algorithm also has stable performance because the difference between its results is small. However, it does not always achieve the best result because all these algorithms have random factors (except BSA algorithm).

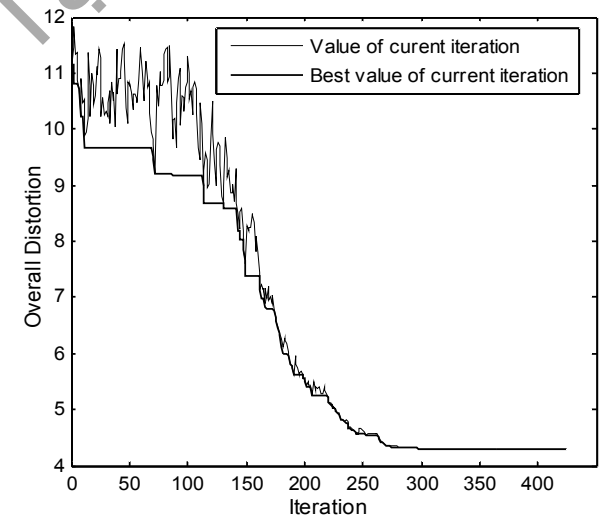

Fig.3a *Overall Distortion vs iteration of SA algorithm*

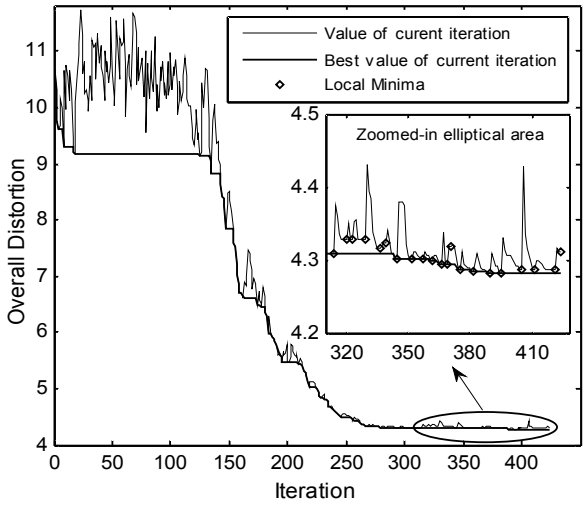

Fig.3b *Overall Distortion vs iteration of MSA algorithm*

In order to further demonstrate the superiority of MSA algorithm over original SA algorithm, two examples of the relationship between the value of current iteration and the number of iterations are illustrated in Fig.3a (for SA) and Fig.3b (for MSA).

In Figure 3a we can see in last 100 interations, when the temperature is very low, the SA algorithm has no ability to find a better solution than before, while in Fig.3b the MSA algorithm can seek better solutions than before. Because in MSA algorithm, we can advoid repeated searching and determine whether the current solution is a local minimum to escape from local minimum traps proactively by accepting worse solutions to further examine.

6. CONCLUSION

In this work, we investigated an JSCC method: IA. This is the scheme of labeling source codewords by binary integer numbers (channel codewords). The main advantage of IA method over other JSCC techniques is that no added delay and redundancy occurs in the coding system. The only costs is some fairly intensive off-line computation during the design process.

However, optimal design of IA is one of the hardest problems in the field of combinatorial optimization problems. SA algorithm is an effective and widely used method to solve such problems approximately. It is improved for the general IA problem in this paper such as reducing the running time, avoiding repeated searching and having mechanism for determining whether the current solution is local minimum. The solution of the proposed algorithm (MSA) is better and the efficiency is higher, which is proved to be true through experiments. Because of the short computational time and the low complexity, the MSA algorithm is appropriate to optimize systems using a large number of codebooks. Moreover, this algorithm can be applied to other combinatorial optimization problems.

Studies are in process to optimize more complex coding structures, such as in MultiStage VQ, Switched Split VQ, Switched MultiStage VQ [13]. In addition, combining IA method and modulation scheme would be one of future works.

REFERENCES

[1] C.E. Shannon, "A Mathematical Theory of Communication", *Bell System Techical Journal*, vol. 27, pp. 379-423, 623-656, 1948.

[2] K. Zeger and A. Gersho, "Pseudo-Gray Coding", *IEEE Trans. on Commun.*, Vol. 38, No. 12, 2147-2158, 1990.

[3] N. Farvadin, "A Study of Vector Quantization for Noisy Channels", *IEEE Trans. on Information Theory,* Vol. 36, No. 4, 799-809, 1990.

[4] L. Tianhao, Y. Songyu, "Evolutionary Algorithm Based Index Assignment Algorithm For Noisy Channel", *Journal of Systems Engineering and Electronics*, Shanghai, Vol. 15, No. 3, 2004, pp. 431-435.

[5] W. Yue, "An Index Assignment Algorithm over Noisy Channel", *Software Engineering and Knowledge Engineering: Theory and Practice,* Springer Berlin Heidelberg, 2012, pp. 663-671.

[6] J. S. Pan, F. R. McInnes and M. A. Jack, "Application of Parallel Genetic Algorithm and Property of Multiple Global Optima to VQ Codevector Index Assignment", *IEE Electronics Letters,* Vol. 32, No. 4, 296-297, 1996.

[7] S. C. Chu, J. S. Pan, "Tabu Search Algorithms to VQ Codevector Index Assignment for Noisy Channels", *Proc. of The Sixth Australian Int. Conf. on Speech Science and Technology (SST-96)*, 1996, pp. 169-174.

[8] J. S. Pan, Z. M. Lu, S. C. Chu and S. H. Sun, "Non-redundant VQ Channel Coding Using Modified Tabu Search Approach With Simulated Annealing", *Third Int. Conf. on Knowledge-Based Intelligent Inform. Engineering Systems*, Australia, 1999, pp. 242-245.

[9] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantization design", *IEEE Trans. on Commun.*, Vol. COM-28, pp. 84-95, 1980.

[10] R. Iordache and I. Tabus, "Index assignment using an ant system approach", *Proc. CSCS-12, 12-th Int. Conf. on Control Systems and Computer Science*, Bucharest, Romania, 26-29 May 1999, pp. 391-395.

[11] W. Xiaohan and X. Wu, "Index Assignment Optimization for Joint Source-Channel MAP Decoding", *IEEE Trans. on Commun.*, Vol 58.3, pp. 901-910, 2010.

[12] H. Oztoprak, S. Villette and A. Kondoz, "Index assignment-based channel coding", *IET Commun.* , vol. 6, iss. 2, pp. 172-178, 2012.

[13] A. Gersbo and R. Gray, "Vector quantization and signal compression", Boston, Ma. Kluwer Academic Publishers, 1992.

[14] M. Bouzid and A. Djeradi, "Joint channel-source coding for robust vector quantization: Application to speech coding", *7th Int. Workshop on Digital Signal Processing for Space Commun. (DSP2001)*, Portugal, 1-3 Oct. 2001.

[15] W. Qiang, Z. Huoming, J. Juan, G. Wenjun & Z. Zhou, "Study on the application of improved simulated annealing algorithm for several types of optimization problem", *7th Int. Conf. on Natural Computation (ICNC)*, IEEE, vol. 3, pp. 1574-1577, 2011.

AUTHORS' BIOGRAPHIES

Tran Ngoc Tuan is working at School of Electronics and Telecommunications, Hanoi University of Science and Technology, Vietnam. He received the Engineer and M.Sc. degrees in electrical engineering from Hanoi University of Science and Technology in 2005 and 2008. His research interests include digital signal processing, speech coding and joint source channel coding.

Nguyen Quoc Trung is working at School of Electronics and Telecommunications, Hanoi University of Science and Technology, Vietnam. He received Ph.D degree from Hungarian Academy of Science in 1982. He was promoted to Accociate Professor in 2004. His research interests include digital signal processing, subband coding and digital communication.