TABU SEARCH ALGORITHMS TO VQ CODEVECTOR INDEX ASSIGNMENT FOR NOISY CHANNELS

S. C. Chu† and J. S. Pan‡

†Dept. of Industrial Engineering and Management, Kaohsiung Institute of Technology, Taiwan
‡Dept. of Electronic Engineering, Kaohsiung Institute of Technology, Taiwan

ABSTRACT - Vector quantization is a popular technique for data compression. It provides good performance against channel noise if suitable algorithm of codevector index assignment is used. The problem of VQ codevector index assignment is NP-hard. In this paper, tabu search approaches are applied to codevector indices assignment for noisy channels for the purpose of minimizing the distortion due to bit errors without introducing any redundancy. Experimental results compared with the standard parallel genetic algorithm and the binary switching algorithm confirm the usefulness of these approaches.

INTRODUCTION

Vector quantization (VQ) (Gray, 1984) is very efficient for data compression in speech and image signals. However, vector quantization as a central reduction scheme is highly sensitive to channel noises. By properly assigning the indices for the codevectors, the average distortion due to channel noise may be reduced without introducing any redundancy. Farvardin (1990) applied the algorithm of simulated annealing to minimize the distortion due to channel noise. Zeger and Gersho (1990) proposed the binary switching algorithm to codevector index assignment. channel-optimized vector quantizer is also developed for codevector indices assignment (Farvardin et al., 1991; Kumazawa, 1984). The spirit of channel-optimized vector quantizer is to optimize codebook generation and codevector indices assignment together by iterative algorithm that is different from the generation of the codebook first, then optimization algorithms are applied to codevector indices assignment. The evaluation methods of codevector indices assignment are also discussed (Knagenhjelm, 1993). Wu and Barba (1993) developed an efficient index allocation algorithm by using the information of a priori probability of codevectors. Potter and Chiang (1995) adopted a minimax design criterion instead of the mean squared error for codevector index assignment. By applying this criterion, the worst case performance is greatly improved while maintaining good average performance. Pan et al. (1996a, 1996b) improved the codevector index assignment by applying the genetic algorithms.

Assume that N codevectors C_i, $i = 1, 2, ..., N$, are assigned codevector indices with an m bit string $b(c_i)$, where $N = 2^m$. Let $P(c_i)$ and $d(c_i, c_j)$ denote the probability of sending codevector C_i and the distortion between codevector C_i and C_j, $i, j = 1, 2, ...N$. A memoryless binary symmetric channel with bit error probability ε is simulated in this letter. For a random assignment of the codevector indices $b = (b(c_1), b(c_2), ..., b(c_N))$, the average distortion for any possible bit errors caused by the channel noise is expressed as

$$D_c = \frac{1-(1-\varepsilon)^m}{N-1} \sum_{i=1}^{N} P(c_i) \sum_{j=1}^{N} d(c_i, c_j) \tag{1}$$

The objective performance for the transmission of indices $b(c_i)$, $i = 1, 2, ..., N$, can be written as

$$D = \sum_{i=1}^{N} P(c_i) \sum_{l=1}^{m} \varepsilon^l (1-\varepsilon)^{m-l} \sum_{b(c_j) \epsilon N^l(b(c_i))} d(c_i, c_j) \tag{2}$$

where $N^l(b(c_i)) = \{b(c_j)\epsilon I, H(b(c_i), b(c_j)) = l\}$, is the lth neighbour set of $b(c_i)$.

In this paper, tabu search approaches (Glover, 1989) are applied to codevector index assignment for noisy channels. Genetic algorithms are a group of methods which solve problems using approaches inspired by the processes of Darwinian evolution. Genetic algorithms are memoryless process. In contrast, tabu search approaches are high-level methods which employ memory functions to find the better optima. The idea of tabu search is to forbid some search directions at a present iteration in order to avoid cycling, so as to jump off local optimum. In addition, aspiration level is an important element of the tabu search approach which can also be applied to codevector index assignment so as the tabu condition can be rejected if the test solution meets the criterion of the aspiration level.

Two tabu algorithms are developed for codevector index assignment in this paper. The first algorithm is to store all indices of the codebook of the current best and non-tabu solution in the tabu list memory. No aspiration level is applied in this algorithm. In the second algorithm, the tabu list memory stores the swapping indices only. The aspiration level is also used in this algorithm. Experimental results confirm the usefulness of this new approach in codevector indices assignment for noisy channels.

ALGORITHMS

Let S_t, s_c and s_b be the test, current and best solutions and V_t, v_c and v_b denote the corresponding test values, best value of current iteration and best value of all iterations, respectively. $S_t = \{s_t^1, s_t^2, ..., s_t^{N_s}\}$, $s_t^i = \{s_t^i(1), s_t^i(2), ..., s_t^i(N)\}$, $1 \leq i \leq N_s$, $V_t = \{v_t^1, v_t^2, ..., v_t^{N_s}\}$, $s_c = \{s_c(1), s_c(2), ..., s_c(N)\}$ and $s_b = \{s_b(1), s_b(2), ..., s_b(N)\}$, where N_s is the number of test solutions and N is the number of codevector indices. The initial test solutions are generated randomly. After the first iteration, the test solutions are generated from the best solution of current iteration as shown in step 2 of the following algorithm. The tabu list memory stores all indices. It is a tabu condition if the record of test solution is the same as any record in the tabu list memory. The first algorithm is as follows:

1. Set the tabu list size T_s, probability threshold P_t, number of test solutions N_s and the maximum number of iterations I_m. Set the iteration counter i=1 and the tabu list length t_l=1. Generate N_s initial solutions $S_t = \{s_t^1, s_t^2, ..., s_t^{N_s}\}$ randomly, calculate the corresponding objective values $V_t = \{v_t^1, v_t^2, ..., v_t^{N_s}\}$ using Eq. 2 and select the current best solution $s_c = s_t^j$, $j = arg\min_l v_t^l$, $1 \leq l \leq N_s$. Set $s_b = s_c$ and $v_b = v_c$.

2. Given the current best assignment s_c of the previous iteration, generate a random number r, $0 \leq r \leq 1$, if $r < P_s$, then $s_t^q(i) = s_c(i)$, $1 \leq q \leq N_s$; otherwise, assign a random number l to $s_t^q(i)$, where $l \neq s_c(j)$, $1 \leq j < i$, $1 \leq l \leq N$, N is the number of codevector indices. Calculate the corresponding objective values $v_t^1, v_t^2, ..., v_t^{N_s}$.

3. Sort $v_t^1, v_t^2, ..., v_t^{N_s}$ in an increasing order. From the best test solution to the worst test solution, if the test solution is non-tabu solution, then choose this test solution as the current best solution and go to step 4; otherwise, try the next test solution. If all test solutions are tabu solutions, then go to step 2.

4. Set $s_b = s_c$ and $v_b = v_c$. Insert s_c to the tabu list. Set the inserting point of the tabu list $t_l = t_l + 1$. If $t_l > T_s$, set $t_l = 1$. If $i < I_m$, set $i = i + 1$ and go to step 2; otherwise, record the best codevector index assignment and terminate the program.

In the first algorithm, no aspiration level is used and the size of the tabu list memory is $N \times N_s$. In the second algorithm, the tabu list memory only store the swapping indices. It is a tabu condition if the swapped indices to generate the test solution from the best solution of current iteration is the same as any record in the tabu list memory. The size of the tabu list memory is $2 \times N_s$. The aspiration level is also applied in this algorithm. The second algorithm is described as follows:

1. Set the tabu list size T_s, number of test solutions N_s and the maximum number of iterations I_m. Set the iteration counter i=1 and the inserting point of the tabu list t_l=1. Generate N_s initial solutions $S_t = \{s_t^1, s_t^2, ..., s_t^{N_s}\}$ randomly, calculate the corresponding objective values $V_t = \{v_t^1, v_t^2, ..., v_t^{N_s}\}$ using Eq. 2 and select the current best solution $s_c = s_t^j$, $j = arg \min_l v_t^l$, $1 \leq l \leq N_s$. Set $s_b = s_c$ and $v_b = v_c$.
2. Given the current best assignment s_c, generate two random integers r_1 and r_2 for each test solution, $1 \leq r_1 \leq N$, $1 \leq r_2 \leq N$, $r_1 \neq r_2$, N is the number of codevector indices. Generate the test solutions by swapping $s_c(r_1)$ and $s_c(r_2)$. Calculate the corresponding objective values $v_t^1, v_t^2, ..., v_t^{N_s}$.
3. Sort $v_t^1, v_t^2, ..., v_t^{N_s}$ in an increasing order. From the best test solution to the worst test solution, if the test solution is a non-tabu solution or it is a tabu solution but the objective value is better than the best value of all iterations (aspiration level), then choose this test solution as the current best solution and go to step 4; otherwise, try the next test solution. If all test solutions are tabu solutions, then go to step 2.
4. Set $s_b = s_c$ and $v_b = v_c$. Insert the swapped indices of the current best solution to the tabu list. Set the inserting point of the tabu list $t_l = t_l + 1$. If $t_l > T_s$, set $t_l = 1$. If $i < I_m$, set $i = i + 1$ and go to step 2; otherwise, record the best codevector index assignment and terminate the program.

EXPERIMENTAL RESULTS

The test materials for these experiments consisted of 200 words recorded from one male speaker. The speech is sampled at a rate of 16 kHz and 13-dimensional cepstrum coefficients (including energy) are computed over 20 ms-wide frames with 5 ms frame shift. A total of 20,030 analyzed frames are used to generate 32 codevectors and 64 codevectors for the experiments of codevector indices assignment.

Experiments were carried out to test the performance of the proposed tabu search approach, standard parallel genetic algorithm, binary switching algorithm and the average distortion of the random assignment using Eq. 1 for 32 codevectors and 64 codevectors. The performance is measured in terms of mean squared error using Eq. 2. The distribution of the codevector probability is set to a uniform distribution.

The first algorithm of the tabu search approach is referred to as TABU. The number of test solution, tabu list size, maximum number of iterations and the probability threshold are 400, 100, 500 and 0.95 respectively. The second algorithm of the tabu search approach for 400 test solutions and 200 test solutions are referred to as TS-400 and TS-200. The standard parallel genetic algorithm and binary switching algorithm are referred to as PGA and BSA. The tabu list size and the maximum number of iterations are 10 and 500. For the standard parallel genetic algorithm, the parameter values used for the group population size P, the number of groups G, the predefined number of generations, the survival rate P_s, the crossover rate P_c, the mutation rate P_m, the number of individuals for selection M, the number of top best for communication B and the number of generations for communication R are 50, 8, 500, 0.5, 0.4, 0.1, 3, 1 and 50 respectively.

One example of the relationship between the best solution of current iteration and the best solution of all iterations in the second algorithm for 0.01 bit error probability is illustrated in

Fig. 1. Table 1 shows the mean squared error of 32 codevectors for TS-400, TS-200, TABU, PGA and BSA with 0.01 bit error probability for 10 runs. Table 2 shows the mean squared error of 64 codevectors for TS-400, TS-200 and BSA with 0.01 bit error probability for 10 runs.

CONCLUSIONS

From these experiments, the robustness of the tabu search strategy compared with the standard parallel genetic algorithm and the binary switching algorithm in mean squared error for non-redundant VQ channel coding in the preliminary experiments is demonstrated. The second algorithm of the tabu search approach is not only better than the first algorithm of the tabu search approach in the optimizational performance, but also less memory is needed in the second algorithm. One of the benefit of using tabu search approach compared with parallel genetic algorithm is that only a few parameters are adjusted in the tabu search approach.

REFERENCES

Gray, R. M. (1984) Vector Quantization, *IEEE Magazine*, 4–29.

Farvardin, N. (1990) A Study of Vector Quantization for Noisy Channels, *IEEE Trans. on Information Theory*, 36(4), 799–809.

Wu, H. S. and Barba, J. (1993) Index Allocation in Vector Quantization for Noisy Channels, *IEE Electronics Letters*, 29(15), 1317–1319.

Zeger, K. and Gersho, A. (1990) Pseudo-Gray Coding, *IEEE Trans. on Communications*, 38(12), 2147–2158.

Farvardin, N. and Vaishampayan V. (1991) On the Performance and Complexity of Channel-Optimized Vector Quantizers, *IEEE Trans. on Information Theory*, 37(1), 155–160.

Kumazawa H., Kasahara M. and Namekawa T. (1984) A Construction of Vector Quantizers for Noisy Channels, *Electronics and Engineering in Japan*, 39–47.

Knagenhjelm P. (1993) How Good Is Your Index Assignment, *IEEE International Conference on Acoustics Speech and Signal Processing*, II-423 – II-426.

Potter, L. C. and Chiang, D.-M. (1995) Minimax Nonredundant Channel Coding, *IEEE Trans. on Communications*, 43(2/3/4), 804–811.

Pan, J. S., McInnes, F. R. and Jack, M. A. (1996a) Application of Parallel Genetic Algorithm and Property of Multiple Global Optima to VQ Codevector Index Assignment for Noisy Channels, *Electronics Letters*, 32(4), 296–297. 29(3), 511–518.

Pan, J. S., McInnes, F. R. and Jack, M. A. (1996b) VQ Codevector Index Assignment Using Genetic Algorithms for Noisy Channels, To appear in *Proc. of The Fourth International Conference on Spoken Language Processing.*

Glover, F. (1989) Tabu Search, Part I, ORSA Journal on Computing, 1(3), 190–206.

Random	*0.12900*				
Seed	*TS-400*	*TS-200*	*TABU*	*PGA*	*BSA*
1	*0.057086*	*0.057020*	*0.057052*	*0.057117*	*0.057919*
2	*0.056864*	*0.056864*	*0.056948*	*0.057323*	*0.059184*
3	*0.057010*	*0.057115*	*0.057144*	*0.057281*	*0.057068*
4	*0.057035*	*0.056940*	*0.057049*	*0.057010*	*0.057985*
5	*0.057003*	*0.057041*	*0.057020*	*0.057371*	*0.057776*
6	*0.057010*	*0.056940*	*0.056864*	*0.057040*	*0.057383*
7	*0.056864*	*0.057176*	*0.057094*	*0.057020*	*0.057131*
8	*0.057020*	*0.057003*	*0.057010*	*0.058332*	*0.057333*
9	*0.057010*	*0.056958*	*0.057010*	*0.057118*	*0.058730*
10	*0.056918*	*0.057003*	*0.057041*	*0.057901*	*0.057305*
Average	*0.056982*	*0.057006*	*0.057023*	*0.057351*	*0.057781*

Table 1: Mean squared errors for ten runs of tabu search approaches, standard parallel genetic algorithm and binary switching algorithm for 32 codevectors

Random	*0.156945*		
Seed	*TS-400*	*TS-200*	*BSA*
1	*0.060059*	*0.060306*	*0.060556*
2	*0.060283*	*0.060363*	*0.061885*
3	*0.060305*	*0.061113*	*0.061695*
4	*0.060608*	*0.060442*	*0.061042*
5	*0.061188*	*0.059880*	*0.061724*
6	*0.060549*	*0.061163*	*0.060211*
7	*0.060990*	*0.060976*	*0.061355*
8	*0.060144*	*0.060617*	*0.061461*
9	*0.060487*	*0.060678*	*0.061382*
10	*0.060623*	*0.060559*	*0.060817*
Average	*0.060524*	*0.060610*	*0.061213*

Table 2: Mean squared errors for ten runs of tabu search approaches and binary switching algorithm for 64 codevectors

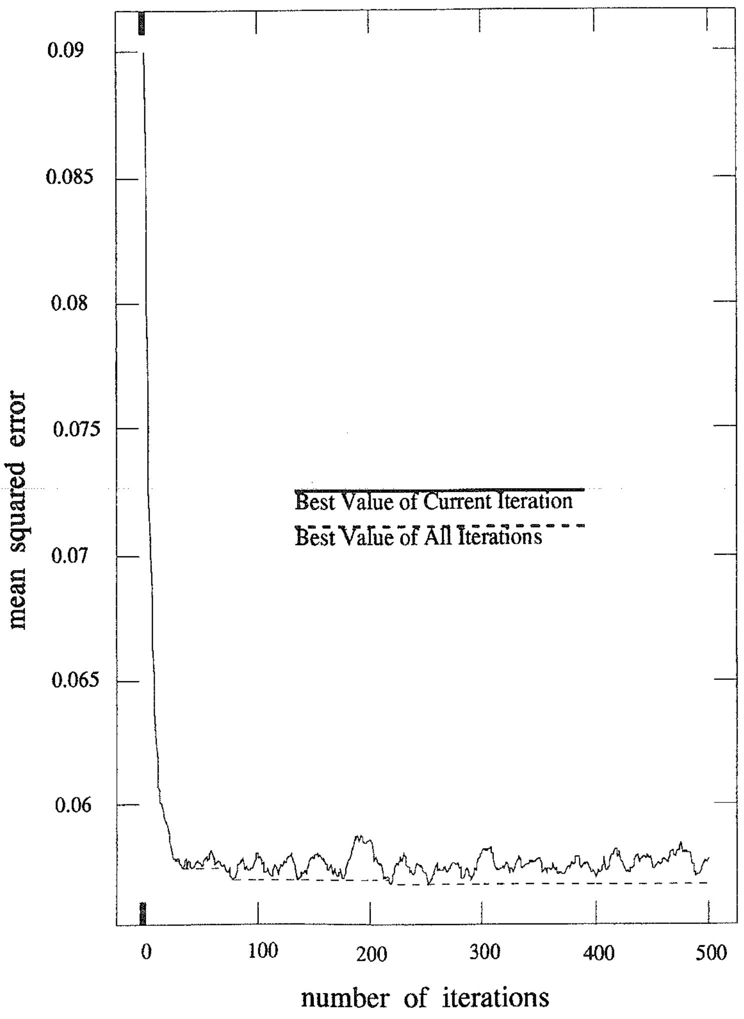

Figure 1: Mean squared error for best solution of all iterations and best solution of current iteration