Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.6, No.9, 2015

Decoder based on Parallel Genetic Algorithm and Multi-objective Optimization for Low Density Parity Check Codes

Hasna Chaibi Ahlam Berkani My Ahmad Faqihi
SIME Lab, ENSIAS, Mohammed V University, Rabat, Morocco

Abstract

Genetic algorithms are powerful search techniques that are used successfully to solve problems in many different disciplines. This article introduces a new Parallel Genetic Algorithm for decoding LDPC codes (PGAD). The results show that the proposed algorithm gives large gains over the Sum-Product decoder, which proves its efficiency. We also show that the fitness function must be improved by Multi-objective Optimization, for this, we applied the Weighted Sum method to improve PGAD, this new version is called (MOGAD) gives higher performance compared to one.

Keywords: Parallel Genetic Algorithms decoder, Sum-Product decoder, Fitness Function, LDPC codes, Error correcting codes, Multi-objective optimization, Weighted sum method.

1. Introduction

The current large development and deployment of wireless and digital communication encourages the research activities in the domain of error correcting codes. The latter is used to improve the reliability of data transmitted over communication channels susceptible to noise. Coding techniques create codewords by adding redundant information to the user information vectors. Decoding algorithms try to find the most likely transmitted codeword related to the received one as depicted in Figure 1.

Decoding algorithms are classified into two Categories: Hard decision and Soft decision algorithms. Hard decision algorithms work on a binary form of the received information. In contrast, soft decision algorithms work directly on the received symbols (Maini et al. 1994). Soft-decision decoding is an NP-hard problem and was approached in different ways. Recently artificial intelligence techniques were introduced to solve this problem. Among the related works, the decoding of linear block codes using algorithm A* (Han et al. 1991), genetic algorithms (Maini et al. 1994, Azouaoui et al. 2012),(Janikow & Michalewicz 1991) and neural networks (Ja-Ling et al. 2002).

LDPC codes were invented by Robert Gallager (Gallager 1998) in his PhD thesis. Soon after their invention, they were largely forgotten, and reinvented several times for the next 30 years. Their comeback is one of the most intriguing aspects of their history, since two different communities reinvented codes similar to Gallager's LDPC codes at roughly the same time, but for entirely different reasons.

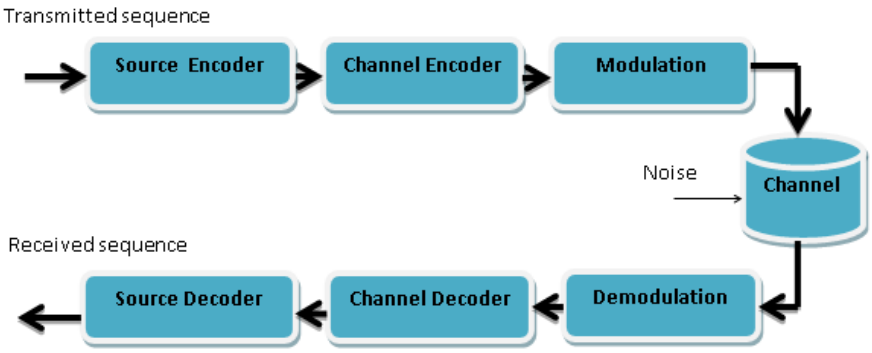

Figure 1. Numerical Communication system model's

Low-density parity-check (LDPC) codes are a class of linear block codes (Othman et al. 2008). The name comes from the characteristic of their parity-check matrix which contains only few 1's in comparison to the amount of 0's. Their main advantage is that they provide a performance which is very close to the capacity for many different channels and linear time complex algorithms for decoding. Furthermore are they suited for

implementations that make heavy use of parallelism.

LDPC codes have emerged as the best error correcting codes with close to the theoretical Shannon limit performance.

When these codes are decoded using Gallager's iterative probabilistic decoding method, also known as the Sum-Product algorithm or Belief propagation algorithm, their empirical BER performance are found to be excellent (Richardson et al. 2001, Richardson & Urbanke 2001, MacKay 1999). This is true when the length of the code vector is large enough.

The LDPC Sum-Product decoding algorithm (Gallager 1998, MacKay & Neal 1997, Wymeersch et al. 2004), makes an estimation of the A Posteriori Probability (APP) of each symbol as a function of the received symbol and the properties of the channel. In this sense, the decoding algorithm does require to know the signal-to-noise ratio in the channel.

In this article we introduces a new Parallel Genetic Algorithm for decoding LDPC codes (PGAD), and we show that the fitness function must be improved by Multi-objective optimization, for this we applied the Weighted Sum method to improve PGAD, this new version is called (MOGAD). In effect, a comparison with other decoder, that are currently the most successful algorithm for LDPC, shows its efficiency, and gives higher performances.

This paper is organized as follows. In section II, we introduce parallel Genetic algorithm; Section III presents PGAD, our decoder and analyses their performances. Section IV presents and analyses the performances of our optimized version by multi-objective optimization. Finally, Section V presents the conclusion and future trends.

2. Parallel Genetic Algorithm

Genetic algorithms (GAs) are powerful search techniques that are used successfully to solve problems in many different disciplines, and are implicitly parallelizable, i.e. many of the operators can be carried out independently of each other's. On multi-processor machines the (usually heavy) workload of calculating function evaluations can be split over each processor, as in the Master-Slave prototype (Paz 1998) .

However, in order to allow for a number of different parallel implementations, perhaps the most straightforward way of parallelizing the genetic algorithm function is to create separate populations evolving independently as separate sub-processes or 'islands'. After each generation the fittest individuals from each 'island' can then 'migrate' to other 'islands' (Figure 2) . If a neighbourhood structure is defined over the set of populations, and once in a while each population sends its best individuals to its neighbours, we say we're running a distributed genetic algorithm. If no swapping of individuals to neighbours is done we have a special case of the distributed model, which we call the partitioned genetic algorithm (Javad et al. 2015) , our work focuses on the last model.

Parallel GAs are particularly easy to implement and promise substantial gains in performance (Pettey et al. 1987, Goldberg 1989, Mühlenbein 1989), and are effective in solving problems of large sizes. Most of these algorithms have been implemented on massive parallel machines and their effectiveness depends on the parallel computing system.

In many of these problems the fitness evaluations for each candidate solution can be calculated independently. This means that each candidate solution can be calculated at the same time, in other words in parallel.

Performing these evaluations in parallel will obviously result in an increase in speed of the algorithm - roughly proportional to the number of processors used. There are, however, reasons for performing GAs in parallel that are believed to give improved performance. If we consider the GA as simply a model of natural systems then some parallel implementations can be viewed as consisting of separate sub-populations evolving independently of each other, with occasional migration allowed between these sub-populations.

There are three main types of parallel GAs (Paz 1998) : global single-population master-slave GAs, single-population fine-grained, and multiple-population coarse-grained GAs .

The most popular parallel Gas consists in multiple populations that evolve separately most of the time and exchange individuals occasionally. This type of parallel GAs is called multi-deme, coarse-grained or distributed GAs (Tongchim 1999, Paz 1998).

The basic idea behind most parallel programs is to divide a task into chunks and to solve the chunks simultaneously using multiple processors. This divide-and-conquer approach can be applied to GAs in many different ways, and the literature contains many examples of successful parallel implementations. Some

parallelization methods use a single population, while others divide the population into several relatively isolated subpopulations. Some methods can exploit massively parallel computer architectures, while others are better suited to multi-computers with fewer and more powerful processing elements.

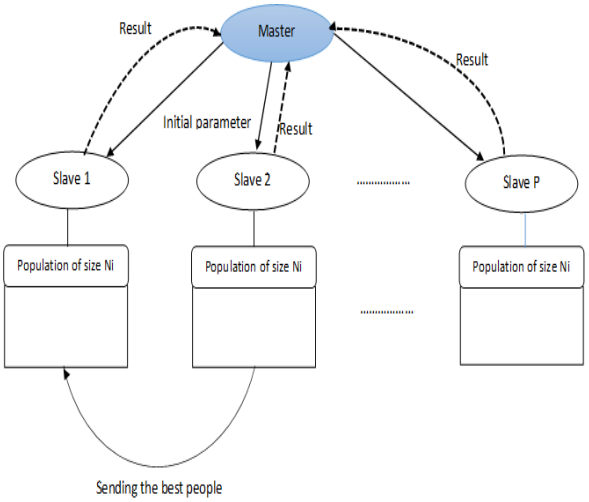

Figure 2. Principle of parallelism islands

3. Parallel Genetic Algorithm Decoder (PGAD)

This work is a parallelization of a new decoder based on Genetic Algorithm, the master computes the syndrome of the received vector, if the syndrome is null, the master machine returns the decoded vector that is equal to the binary decision of the received one, if not, the slaves turns GAs (figure 4) in parallel with an initial population randomly generated for each one. Each process develops independently its population until he decided to gather his best individual which will be a candidate for the decision step (figure 3).

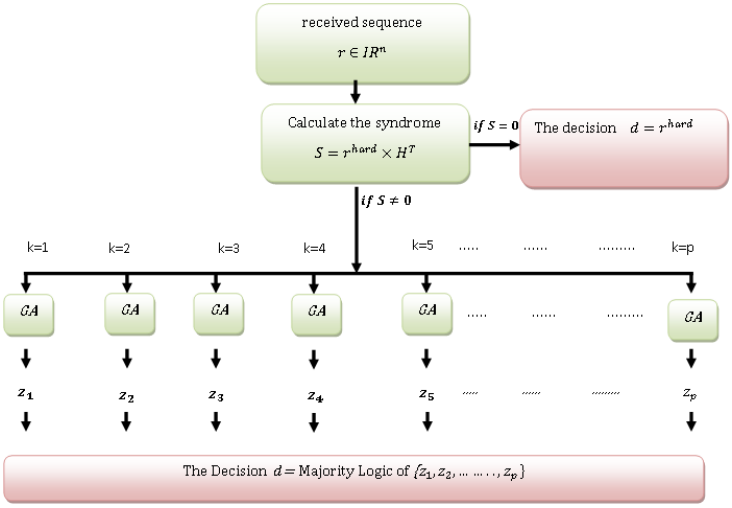

Figure 3. Parallelization of the Genetic Algorithm Decoder.

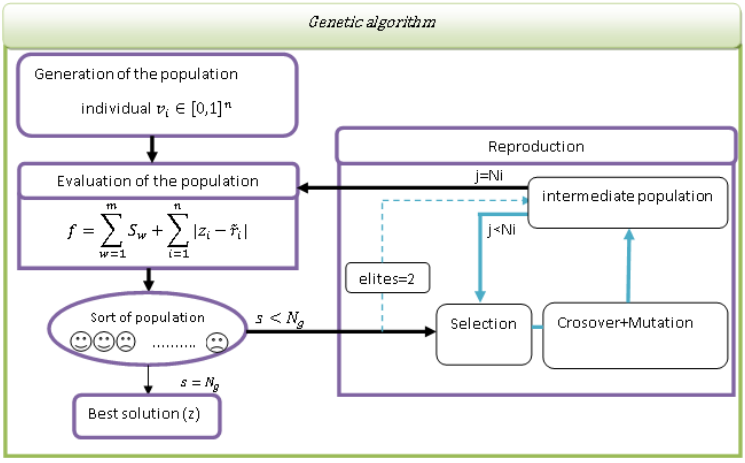

Figure 4. The proposed Genetic Algorithm.

3.1 The PGAD decoder

Let $\boldsymbol{C}$ a low-density parity-check (LDPC) code, and let $(\mathbf{r_i})_{\mathbf{1 \le i \le n}}$ be the received sequence over a communication channel with noise variance $\boldsymbol{\sigma = N_0/2}$, where $\boldsymbol{N_0}$ is noise power spectral density.

Let $\boldsymbol{N_i}$, $\boldsymbol{N_e}$, $\boldsymbol{N_g}$ and $\boldsymbol{P}$ denote, respectively, the population size, the number of elite members, the number of generations and the number of processor.

Let $\boldsymbol{p_c}$ and $\boldsymbol{p_m}$ be the crossover and the mutation rates.

Let $\boldsymbol{U = [0,1]}$, $\tilde{\boldsymbol{r}} \in \boldsymbol{U}$ is the received vector transformed into $\boldsymbol{[0,1]}$ interval using the logistic function (3).

$$\tau: IR \rightarrow U \qquad (1)$$

$$\tau: r \rightarrow \tilde{r} \qquad (2)$$

$$\tilde{r}_i = \frac{1}{1+e^{-2r_i}} \qquad (3)$$

The decoding-based on Parallel Genetic Algorithm is depicted on Figure 3 and Figure 4.The steps of the decoder based on PGA are as follows:

Step1: The master machine calculates de syndrome of de received vector (eq.4).

$$\boldsymbol{S = r^{hard} * H^T} \qquad (4)$$

where $\boldsymbol{r^{hard}}$ is the hard decision of $\boldsymbol{r}$ (eq.5), and $\boldsymbol{H^T}$ is the transpose of the matrix $\boldsymbol{H}$.

$$r_i^{hard} = \begin{cases} 1 & if \quad r_i > 0 \\ 0 & Otherwise \end{cases} \qquad (5)$$

If $\boldsymbol{S_i = 0} \quad \forall \boldsymbol{i} \in \boldsymbol{[0, n-k]}$, then a valid code vector $\boldsymbol{d}$ is obtained by $\boldsymbol{d = r^{hard}}$. Otherwise the sub-Genetic Algorithms start running in parallel way on each processor separately as given below:

Step2. Generate an initial random population containing $\boldsymbol{Ni}$ soft vectors $\boldsymbol{v_i}$ of $\boldsymbol{n}$ components $(\boldsymbol{v_i} \in \boldsymbol{[0,1]})$.

Step3. Compute the fitness of each individual in the population:

The fitness function is the sum of the syndrome weight of de candidate, and the distance between the candidate vector and the received one (eq.6).

$$fitness = \sum_{j=1}^{m} S_j + \sum_{i=1}^{n} |z_i - \tilde{r}_i| \qquad (6)$$

Where

$$z_i = \begin{cases} 0 & if \quad \tilde{r}_i < v_i \\ 1 & Otherwise \end{cases} \qquad (7)$$

And $S = z * H^T$ (8)

$\boldsymbol{z}$ is the solution candidate of GA.

$\boldsymbol{m}$ and $\boldsymbol{n}$, denote, respectively, the number of rows of the parity check matrix $\boldsymbol{H}$, and the code vector length.

Step4. The population is sorted in ascending order of candidates' fitness value defined by (eq.6).

Step5. The best two candidates $(\boldsymbol{Ne = 2})$ of each generation are inserted in the next one.

Step6. The other $\boldsymbol{Ni - Ne}$ members of the next generation are generated as follows:

Substep6.1. *Selection operation:* a selection operation that uses the tournament selection method is applied in order to identify the best parents$(\boldsymbol{v_1, v_2})$, on which the reproduction operators are applied.

Substep6.2.*Crossover operation:* Create new vectors $(\boldsymbol{v'_1, v'_2})$ "children", with a given probability rate $\boldsymbol{p_c = 0.95}$. We use Ring Crossover (RC) (Kaya et al. 2011).

Substep6.3.*Mutation operator:* To complete the new generation, children are mutated by introducing random changes with a given probability rate $\boldsymbol{p_m = 0.01}$ to single parent.

The best member from the last generation for each GA run is returned as the candidate for the next step.

Step7. Decision

The $\boldsymbol{P}$ vectors are a set of possible solutions of the $\boldsymbol{P}$ runs of GA, So, the process of decision stage generates the final solution, i.e, a decoded vector $\boldsymbol{d}$.

This process applies the majority logic, a procedure which performs a component wise decision over the $\boldsymbol{z}$ candidate vectors, setting each final component $\boldsymbol{d_i}$ as the bit state of higher frequency.

3.2 Simulation Results and Discussions related to PGAD:

In order to prove the effectiveness of PGAD, we do intensive simulations.

The simulations where made with default parameters outlined in Table 1. The performances are given in terms of BER (bit error rate) as a function of SNR (Signal to Noise Ratio Eb/N0).

Table 1

Default parameters

Simulation parameter	***Parameter value***
Pc (crossover rate)	0.95
Pm (mutation rate)	0.01
Ng (generation number)	25
Ni (population size)	500
Ne (elite number)	2
Channel	AWGN
Modulation	BPSK
Minimum number of bit errors	100
Minimum number of bloc	300
P(GA runs)	15
Default code	Regular LDPC(60,30)
Type of crossover	Ring Crossover (RC)
Type of selection	Tournament

Comparison with Sum-Product Decoder

Our new decoder has been compared with the Sum-Product Decoder for regular LDPC(60,30), LDPC(75,45) and

LDPC(96,48) codes. The results are given in Figure 5, figure 6 and Figure 7:

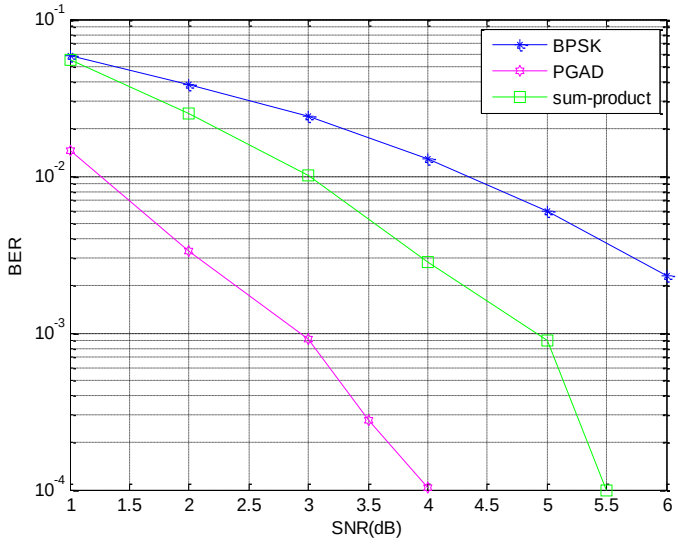

Figure 5. Performances of PGAD compared to sum-product decoder for a regular LDPC(60,30) code.

The figure 5 shows that the PGAD provides good performances compared to sum-product decoders for regular LDPC(60,30) code. The gain between the PGAD and sum-product decoder is 1.5 dB at10^{-4}.

Figure 6 compares the performances of PGAD with sum-product decoder for regular LDPC(75,45) code. We remark that the PGAD is better than sum-product decoder. The gain between the PGAD and sum-product decoder is 2.5 dB at 10^{-3}.

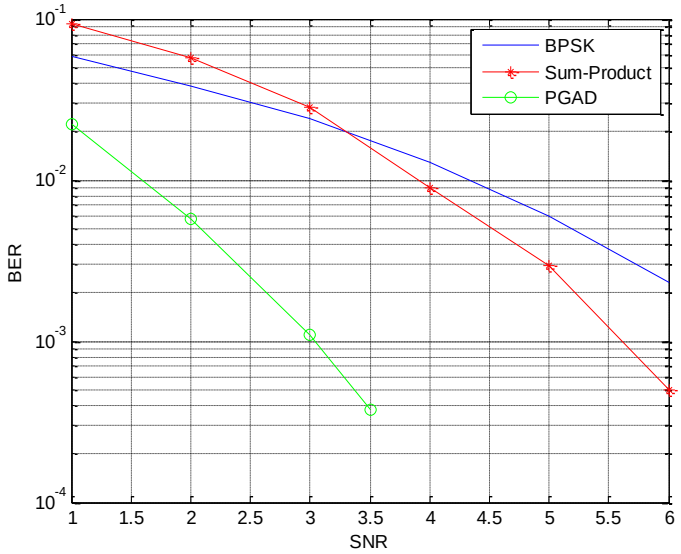

Figure 6. Performances of PGAD compared to sum-product decoder for a regular LDPC(75,45) code.

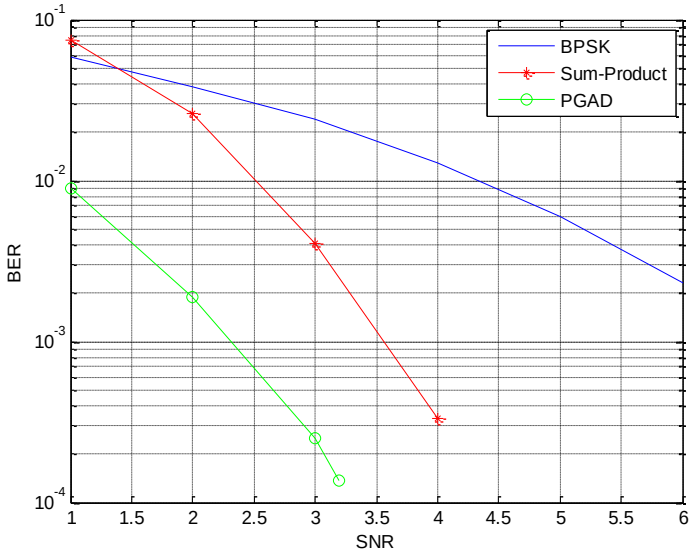

Figure 7. Performances of PGAD compared to sum-product decoder for a regular LDPC(96,48) code.

Figure 7 compares the performances of PGAD with sum-product decoder. We remark that the PGAD is better than sum-product decoder for regular LDPC(96,48) code. The gain between the PGAD and sum-product decoder is 1.5 dB at 10^{-3}.

4. The Multi-objective optimization to Improve fitness function

In this section we show that the fitness function (eq.9), must be improved using multi-objective optimization.

Based on a comparison between the PGAD decoder where the fitness is equal to the first part of fitness (Syndrome Weight (SW): f_1) and where it is equal to the second part of the fitness (Distance between the Candidate vector and the Received vector (DCR): f_2) (figure 8), we remark that:

For all SNR when the fitness is equal to SW, the performances are always better than the case when the fitness is equal to DCR.

$$\boldsymbol{fitness} = \boldsymbol{\alpha} \times \sum_{\boldsymbol{w=1}}^{\boldsymbol{m}} \boldsymbol{S_w} + \boldsymbol{\beta} \times \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} |\boldsymbol{z_i} - \tilde{\boldsymbol{r}}_{\boldsymbol{i}}| \qquad (\mathbf{9})$$

$$f_1 = \sum_{w=1}^{m} S_w$$

$$f_2 = \sum_{i=1}^{n} |z_i - \tilde{r}_i|$$

We also note that the SW as fitness gives better results than both functions in the fitness. Then, we can deduce that the SW affects much more the performances than the DCR.

We also note that the performances presented by the DCR are very degraded compared to those given by the SW. Nevertheless, the SW has managed to mitigate their effects, and as a result, the performances of the two functions together are closer to those presented by the SW than the ones presented by the DCR.

Whereby, when we trace the performances of both functions, we gave the SW and DCR the same importance by factoring theme to equal coefficients. This is not just because, we valorise the same way two things that do not have equal importance.

It is very remarkable that the SW is more important than the DCR (figure 8), which pushes us to introduce coefficients (α and β) for the two functions (f_1 and f_2) (eq.9).

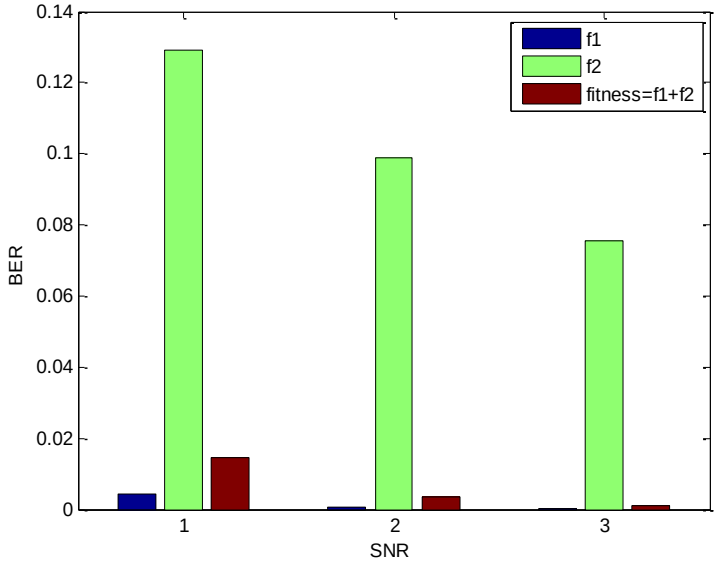

Figure 8. Performances of PGAD decoder for a regular LDPC (60, 30) codes for tree fitness.

The coefficients α and β must have specific values to have the best possible performances.

Therefore, we are in front of a multi-objective optimization problem.

4.1 Multi-objective Optimization Problem:

Most real optimization problems are described using a number of objectives or criteria to be optimized simultaneously. While for the problems include a single objective, the optimum sought is clearly defined, it remains to be formalized for multi-objective optimization problems. Indeed, for a problem to be two contradictory goals, the sought optimal solution is a set of points corresponding to the best possible compromise to resolve our problem

In the case, one extreme solution not necessary be the best solution for other objectives if not satisfy both objective functions and the optimal solution of one objective. Therefore different solutions will produce trade-offs between different objectives and a set of solutions is required to represent the optimal solutions of all objectives. The multi-objective optimization problems may also have one or more constraints including inequality, equality or variable bounds to be satisfied. However in real engineering applications usually more than one constraint is involved in the problem. In multi-objective optimization values of objective functions create a multidimensional space called objective space. Each decision variable on variable space corresponds to a point in objective space.

Our contribution is the development of genetic algorithms for finding the best performance using the multi-objective optimization. We choose a weighted sum method for multi-objective optimization to improve the fitness function. This method is the simplest approach and probably the most widely used classical method. This method scalarizes the set of objectives into a single objective by multiplying each objective with a user supplied weight.

4.2 Weighted Sum Method for multi-objective optimization (MOGAD) Applied to our decoder:

The weighted sum strategy converts the multi-objective problem of minimizing the vector of criteria functions, into a scalar problem by constructing a weighted sum F of all the objectives.

$$F_i = \sum_{i=1}^{N_i} w_j f_{ij} \qquad j \in [1, q] \qquad (10)$$

Where $\sum_{j=1}^{q} w_j = 1$ (11)

$\mathbf{f_{ij}}$ is the actual value of the $\mathbf{N_i}$ actions in terms of the q criterion and $\mathbf{w_j}$ is the weight or importance of the q criterion.

The problem can then be optimized using a standard unconstrained optimization algorithm. The problem here is in attaching weighting coefficients to each of the objectives. The weighting coefficients correspond directly to the relative importance of the objectives (Kim & Weck 2009) ,(Das & Dennis 1997).

4.3 The proposed algorithm

For the rest of this section we use the following notation.

Let $\boldsymbol{N_i}$, $\boldsymbol{N_e}$, $\boldsymbol{N_g}$ and $\boldsymbol{P}$ denote, respectively, the number of actions (the population size), the number of elite members, the number of generations and the number GA executions (runs) .

m, n and q are respectively the number of rows of the parity check matrix H, the code vector length and number of criterion. (In our case we have two criterions).

Let $\boldsymbol{p_c}$ and $\boldsymbol{p_m}$ be the crossover and the mutation rates.

Let $\boldsymbol{v_i}$ be an individual called action, $\mathbf{F_i}$ is the performance of each action, $\mathbf{f_1}$ and $\mathbf{f_2}$ are the criterion.

The algorithm based on weighted sum method which will run parallely is given below:

Step1:The algorithm begins creating an initial population of $\boldsymbol{N_i}$ vectors $\boldsymbol{v_i}$ with real components ($\boldsymbol{v_i} \in [\mathbf{0}, \mathbf{1}]$).

Step2: Evaluate the individuals (action) for each of criterion (we have two criterion $\mathbf{f_1}$ and $\mathbf{f_2}$,$q = 2$):

-For k from 1 to $\boldsymbol{N_i}$:

Substep2.1: Calculate the criterion f1: $f_{1k} = \sum_{j=1}^{m} S_j$

Substep2.2: Calculate the criterion f2: $f_{2k} = \sum_{i=1}^{n} |z_i - \tilde{r}_i|$

Substep2.3: Normalization of $\boldsymbol{f_{1k}}$, $\boldsymbol{f_{2k}}$ to preserve the proportionality between the values.

Subsubstep2.3.1: Calculate : $f'_k = \sum_{i=1}^{q} f_{ki}$

Subsubstep2.3.2: For i from 1 to q calculate: $f''_{ki} = \frac{f_{ki}}{f'_k}$

Substep2.4: Normalization of the weights (the sum of the weights $(w_i) = 1$)

-For i from 1 to q : $w'_i = \frac{w_i}{\sum_i^m w_i}$

Substep2.5: Implementation of the weighted sum method

$$\boldsymbol{F_k} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{q}} \boldsymbol{w_i f''_i} \quad (12)$$

Therefore an only criterion for each action(individual) k.

step3: The population is sorted in ascending order of member's fitness defined in (eq.12) .

The best two members $(\boldsymbol{Ne} = \mathbf{2})$ of each generation are inserted in the next one.

step4: The other $\boldsymbol{Ni} - \boldsymbol{Ne}$ members of the next generation are generated as follows:

Substep4.1.Selection operation: a selection operation that uses the Tournament Selection method is applied in order to identify the best parents$(\boldsymbol{v_1}, \boldsymbol{v_2})$, on which the reproduction operators are applied.

Substep4.2. Crossover operator: Create new vectors $(\boldsymbol{v'_1}, \boldsymbol{v'_2})$ "children", with a given probability rate $\boldsymbol{p_c} = \mathbf{0.95}$. We use ring crossover.

Substep4.3. Mutation operator: To complete the new generation, children are mutated by introducing

random changes with a given probability rate $\boldsymbol{p_m = 0.01}$ to single parent.

The best member from the last generation is returned as the candidate of the decision step.

We use the same decision method as PGAD decoder.

4.4 Simulation Results and Discussions related to MOGAD:

In order to prove the effectiveness of MOGAD, we simulated our decoder in de same conditions as PGAD in section 3.

The simulations where made with default parameters outlined in Table 1.

The figure 9 shows the performance of MOGAD for a regular LDPC(60,30) compared to PGAD and sum-product decoders. This figure shows that the MOGAD provides good performances compared to PGAD and sum-product decoders. The gain between the MOGAD and PGAD decoder is 1 db in 10^{-4}.

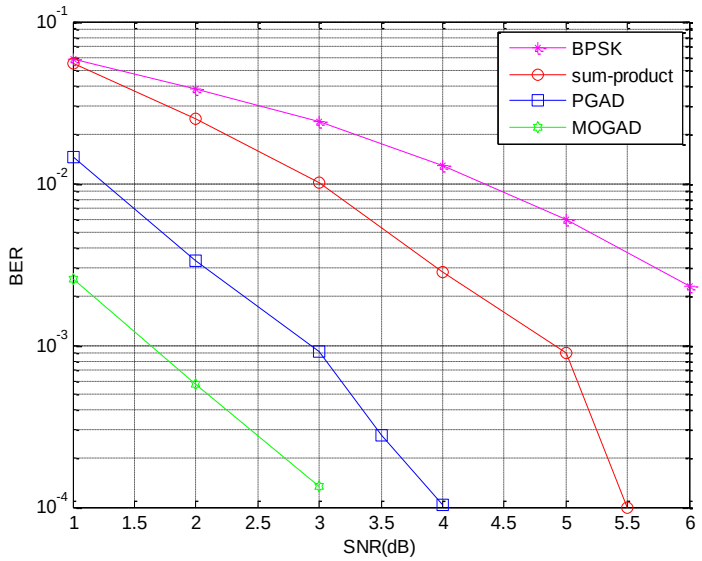

Figure 9. Performance of MOGAD decoder compared to Sum-product and PGAD decoders for a regular LDPC(60,30).

5. Conclusion

In this paper, we have proposed a new decoder based on parallel GA for LDPC codes. The simulations applied on some LDPC codes, show that the proposed algorithm is an efficient one. The comparison between our PGAD and sum-product decoder shows that our decoder is better in terms of performances. we have shown that the fitness function must be improved by multi-objective, for this purpose, we applied the weighted sum in PGAD decoder which gives better performances compared to our decoders. The obtained results will open new horizons for the artificial intelligence algorithms in the coding theory field.

References

Kaya, Y. Uyar, M. and Tekin R. (2011), "A Novel Crossover Operator for Genetic Algorithms: Ring Crossover," presented at CoRR.

Han, Y. S. Hartmann, C. R. P. and Chen, C.-C. (1991), "Efficient maximum likelihood softdecision decoding of linear block codes using algorithm A*", Technical Report SUCIS- 91-42, School of Computer and Information Science, Syracuse University, Syracuse, NY 13244.

Maini, H.S. Mehrotra, K. G. Mohan, C. Ranka, S. (1994) "Genetic Algorithms for Soft Decision Decoding of Linear Block Codes", Journal of Evolutionary Computation, Vol.2, No.2, pp.145-164. Azouaoui, A. Belkasmi, M. and Farchane, A. (2012)"Efficient Dual Domain Decoding of Linear Block Codes Using Genetic Algorithms", Journal of Electrical and Computer Engineering, vol. 2012, Article ID 503834, 12 pages.

Ja-Ling Wu. , Yuen-Hsien Tseng, and Yuh-Ming Huang, (2002)"Neural Networks Decoders for Linear Block Codes", International Journal of Computational Engineering Science, vol.3, No.3, pp.235-255, 2002.

Wymeersch, H. Steendam, H. and Moeneclaey, M. (2004) "Log-domain decoding of LDPC codes over GF(q)",IEEE Communications Society, vol. 45(2), pp. 399-431.

Vansnick, J.C. (1990), "Measurement theory and decision aid", in Banae Costa (ed.), Readings in Multiple Criteria Decision Aid, Springer-Verlog, Berlin, 81-100.

Tongchim, S. (1999) "Coarse-Grained Parallel Genetic Algorithm for Solving the Timetable Problem", Proc.of the 3 rd Annual Nat.Symp.on Computational Science and Engineering. Bangkok, Thailand, 1999.

Pettey, C. Leuze, M. Grefenstette, J. (1987) "A parallel genetic algorithm," In Proceedings of the Second International Conference on Genetic Algorithms, pp. 155-161, Los Altos, CA.

Goldberg, D. E. (1989) "Sizing populations for serial and parallel genetic algorithms," in Proceedings of the Third International Conference on Genetic Algorithms, pp. 70-79, San Mateo, CA, 1989.

Mühlenbein H. , (1989) "Parallel genetic algorithms, population genetics and combinatorial optimization," Proceedings of the Third International Conference on Genetic Algorithms, pp. 416-421, San Mateo, CA.

Paz, E. C. (1998) "A Survey of Parallel Genetic Algorithms," Calculateurs Paralleles, Reseaux et Systems Repartis, v10, pp. 141-171.

Gallager, R. G. (1998) "Low-Density Parity-Check Codes". The MIT Press, Sep. 1963.

Richardson, T. Shokrollahi, A. and Urbanke, R. (2001) "Design of capacity approaching irregular Low-Density Parity-Check codes," IEEE Trans. Inform.Theory, 47,619-637.

Richardson, T. and R. Urbanke, (2001) "The capacity of Low-Density Parity-Check codes under message-passing decding," IEEE Trans. Inform.Theory, 47,599-618.

MacKay D.J.C., (1999) "Good error-correcting codes based on very sparse matrices" IEEE Trans. Inform.Theory, 45,399-431.

MacKay D.J.C, and R.M. Neal, (1997)"Near Shannon Limit Performance of Low-Density Parity-check Codes" IEE Elect. Lett., 33, 457-458.

Kim, I.Y, Weck, O.L. (2009), "adaptive Weighted Sum Method for Multiobjective Optimization." A/AA, (2004-4322).

Das, I., and Dennis, J. E., (1997) "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems" Structural Optimization, Vol. 14, pp. 63-69.15.

Janikow C I, Michalewicz I. (1991) "An Experimental Comparison of Binary and Floating Point Representation in Genetic Algorithms". In Proc 4th Int Conf Genetic Algorithms. 31～36.

Javad V. , Marzuni S.S.M, Farzai S.,(2015) "Comparing performance of parallel grouping genetic algorithm with serial grouping genetic algorithm for clustering problems" in International Journal of Mechatronics, Electrical and Computer Technology, Vol. 5(15) Apr. 2015, PP. 2198-2206.

Othman O. Khalifa, khan S., Zaid M., and Nawawi M., (2008.)"Performance Evaluation of Low Density Parity Check Codes", International Journal of Computer Science and Engineering, pp. 67-70, Spring.

The IISTE is a pioneer in the Open-Access hosting service and academic event management. The aim of the firm is Accelerating Global Knowledge Sharing.

More information about the firm can be found on the homepage:
http://www.iiste.org

CALL FOR JOURNAL PAPERS

There are more than 30 peer-reviewed academic journals hosted under the hosting platform.

Prospective authors of journals can find the submission instruction on the following page: http://www.iiste.org/journals/ All the journals articles are available online to the readers all over the world without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. Paper version of the journals is also available upon request of readers and authors.

MORE RESOURCES

Book publication information: http://www.iiste.org/book/

Academic conference: http://www.iiste.org/conference/upcoming-conferences-call-for-paper/

IISTE Knowledge Sharing Partners

EBSCO, Index Copernicus, Ulrich's Periodicals Directory, JournalTOCS, PKP Open Archives Harvester, Bielefeld Academic Search Engine, Elektronische Zeitschriftenbibliothek EZB, Open J-Gate, OCLC WorldCat, Universe Digtial Library , NewJour, Google Scholar