International Journal of Computer Applications (0975 – 8887)
Volume 119 – No.4, June 2015

Combined Partitioning Hardware-Software Algorithms

Mehdi Jemai Sonia Dimassi Bouraoui Ouni Abdellatif Mtibaa

Laboratory of electronic and microelectronic
National Engineering School of Monastir University of Monastir,
Monsatir 5000, Tunisia

ABSTRACT

In recent years, meta-heuristics have become increasingly interesting in solving combinatorial problems including hardware-software partitioning. In this paper, we present a comparative study between some algorithms which involve meta-heuristics based on Tabu search, genetic algorithm and Binary Search Trees to address the problem of hardware-software partitioning. In fact, meta-heuristics can produce acceptable solutions within a reasonable time, but they do not guarantee an optimal solution. We have proposed these algorithms to find the efficient hardware-software partition that minimizes the logic area of System on a Programmable Chip while respecting a time constraint. This paper presents an analysis of these meta-heuristics by identifying the fundamental ideas guiding the choice of a meta-heuristic in practice.

Keywords

SOPC; Hardware-software partitioning; Tabu search; genetic algorithm; Binary search trees.

1. INTRODUCTION

Systems on programmable chip (SOPC) are increasingly common in embedded systems. They consist of multiple functions including one or more processors, one or more reconfigurable areas, a signal processor Digital Signal Processor (DSP), various peripherals and memory or analog parts. Many hardware and software techniques must be developed to achieve specific constraints in terms of area, performance, power consumption, etc. In this paper, we present hardware-software partitioning algorithms based on Tabu search, genetic algorithm and Binary Search Trees to minimize the logic area on SOPC. In fact, the two proposed algorithms incorporate the Binary Search Trees respectively into genetic algorithm [1] and Tabu search. The main objective of these algorithms is to balance all design parameters to find a better compromise between the logic area of the application and its execution time.

Combinatorial optimization has become increasingly used in many research fields such as operations research, discrete mathematics and computer science. Indeed, optimization problems [2] are difficult and many practical applications can be formulated as a combinatorial optimization problem [3] such as hardware-software partitioning. Generally, combinatorial optimization problems are easy to identify but difficult to solve. In fact, most of these problems are considered as NP-hard problems and there is no effective algorithmic solution for all data [4]. In this context, many approaches have been developed in operations research and artificial intelligence. These methods can be grouped into two categories as shown in Figure 1: exact method that ensure the completeness of the resolution and approximate methods which lose completeness to gain efficiency.

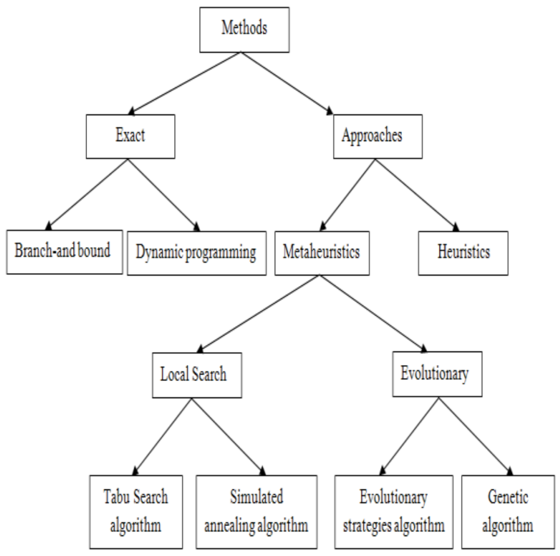

Fig.1: The main methods of resolution

2. RELATED WORK

Today, several factors have resulted in the necessity of co-design, such as the complexity of the structure of modern embedded systems, the requirements of embedded systems market, the emergence of multimedia systems, the requirements on cost, power and the performance. Traditionally, the hardware-software partitioning is done manually. Target systems are generally presented as a task graph, which describe the dependencies between the components of the embedded system. Many approaches have been developed to solve the problem of hardware-software partitioning [5], [6], [7], [8], [9] and they aim to automate the task of partitioning. These approaches include the exact methods and the meta-heuristics.

The main principle of an exact method is to enumerate implicitly all solutions of the search space. These methods have techniques to detect earliest possible failures and specific heuristics to guide choices. Some of these methods include more traditional methods such as separation techniques and progressive evaluation. The exact methods have allowed finding optimal solutions for reasonably sized problems. Despite progress, particularly in terms of the integer linear programming, the execution time required to find a solution could increase exponentially with the size of the problem, the exact methods typically have difficulty with applications having a large size. Amongst the common approaches we quote the exact algorithms such as branch-and bound [10], integer linear programming [11] and dynamic programming [12]. These exact methods are used to solve problems with a small size in order to find optimal solutions.

A new generation of powerful and efficient approximate methods, often called meta-heuristics [13], [14] have been developed to overcome the drawback of the exact methods They are a very interesting alternative to treat large optimization problems where the optimality is not paramount. Meta-heuristics are represented mainly by methods of neighborhood such as such as simulated annealing algorithms [15], Tabu search and greedy algorithms [16], evolutionary algorithms such as genetic algorithm [17] and evolutionary strategies. They have been extensively used to solve partitioning problem. Now, with these meta-heuristics, we can propose approximate solutions for larger classical optimization problems and for many applications that it was impossible to treat previously [18], [19]. In recent years, the interest in meta-heuristics increases continuously in operations research and artificial intelligence.

In this paper, we have proposed a comparative study between the hardware-software partitioning algorithms based on Tabu search, genetic algorithm and the two proposed algorithms which incorporate the Binary Search Trees (BST) respectively into Tabu search and genetic algorithm [1].

3. BACKGROUND

3.1 Binary search trees

The most used data structure to store ordered data and retrieve information is the trees. They are the largest non-linear structure involved in the computer science. This structure can be adapted to the natural representation of organized and homogeneous information. The trees are used in many computing areas, such as compilation (representation of expressions by the trees syntax), imaging (quaternary trees), algorithmic (sorting methods or management information in tables), or in the fields of artificial intelligence (game trees, decision trees, resolution trees). The BST is interesting because they optimize the access time to information. In computer science, a BST, sometimes called an ordered or sorted binary tree, is a hierarchical data structure with a single reference to root node, each node has at most two child nodes (a left and a right child) and the label of each node is greater than any node in its left sub-tree and less than each node of the right sub-tree.

3.2 Genetic algorithms

Genetic algorithms have been developed by Holland [20]. They are the highly scalable methods based on biological mechanisms related to selection principles, and natural evolution. Genetic algorithms were quickly adapted to a variety of contexts. In a simple genetic algorithm [21], research is regulated by three operators (operator of reproduction, crossover and mutation operator) which are applied sequentially. They are generally used to find a solution, the best solution after a certain number of generations.

The evaluation of the quality (fitness) of an individual can be illustrated with a numerical value, the quality of the genes that make up the individual. More the quality of an individual is higher, more it will have chance to be selected for reproduction. The reproduction is made by crossing two individuals. The generic operators are applied to the two selected individuals, usually cross-over and mutation. The reproduction provides two children (offspring) that are placed in the new population. Reproduction is repeated until we have completed the new population (the population size should remain constant). Then, we replace the old population by the new, and the process is repeated according to the needed number of generations. Finally, the algorithm will return the best individual of the latest generation as the solution of the problem.

3.3 The Tabu Search

The Tabu technique was introduced by Glover [22]. It is a general iterative method from combinatorial optimization and it can be considered as a generalization of local improvement methods. The principal of the Tabu Search is defined as follow: on the basis of any solution x belonging to the set of solutions X, we are moving towards a solution s(x) belonging to the neighborhood S(x) of x. Therefore, the algorithm explores iteratively the space of solutions X. In order to choose the best neighbor s(x), the algorithm evaluates the objective function "f" at each point s(x), and retains the neighbor that enhances the value of "f" or who degrades the least.

4. PROPOSED ALGORITHM

The proposed algorithms are based on the Tabu search, genetic algorithm and the BST. To reduce the logic area on SOPC, the small modules are assigned to the Left Sub-Tree (LST) and the large modules are assigned to the Right Sub-Tree (RST). In fact, the left sub-tree is assigned to the hardware part and the RST is assigned to the software part of architecture. To improve the obtained hardware-software partitioning, genetic algorithm and Tabu search will be applied on the left sub-tree or on the RST according to the time constraint. In such a way we will have the tasks that will migrate from the software part to the hardware part of architecture or the contrary, in order to have the best hardware-software partitioning. The detail of the proposed algorithms is shown in Figure 2.

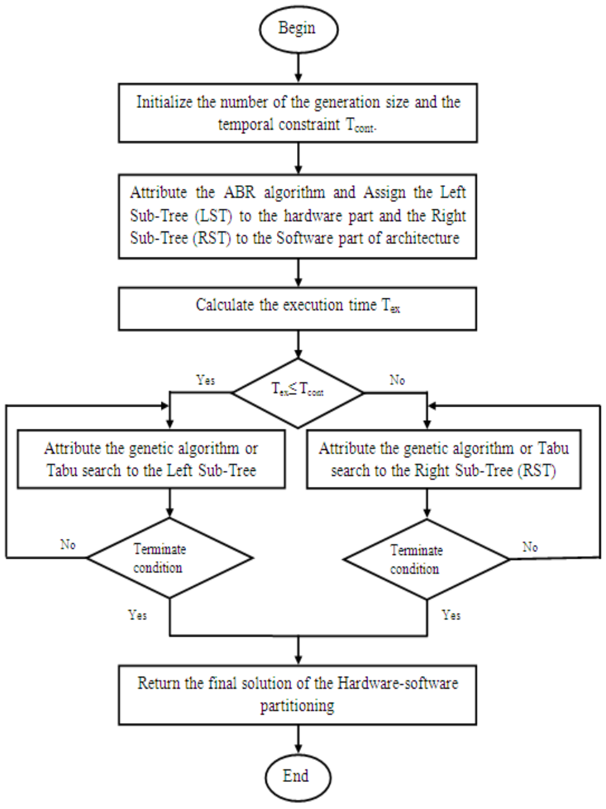
Begin
Initialize the number of the generation size and the temporal constraint T_{cont}.
Attribute the ABR algorithm and Assign the Left Sub-Tree (LST) to the hardware part and the Right Sub-Tree (RST) to the Software part of architecture
Calculate the execution time T_{ex}
$T_{ex} \leq T_{cont}$
Yes
No
Attribute the genetic algorithm or Tabu search to the Left Sub-Tree
Attribute the genetic algorithm or Tabu search to the Right Sub-Tree (RST)
Terminate condition
Terminate condition
No
No
Yes
Yes
Return the final solution of the Hardware-software partitioning
End

Fig.2: Main procedures of the proposed algorithms

5. EXPERIMENTS RESULTS

Our experiments were executed on the H.264 AVC which is the most recent standard for video coding. It has been developed by the ITU-T Video Coding Experts Group as shown in Figure 3 [6]. The H.264 contains an intra-prediction mode with 4×4, 8×8 and 16×16 block sizes. So, the various intra prediction modes influence the computational complexity of H.264 encoder. Recently efficient hardware architectures were proposed for the fast execution of H.264/AVC intra prediction mode selection [23][24].

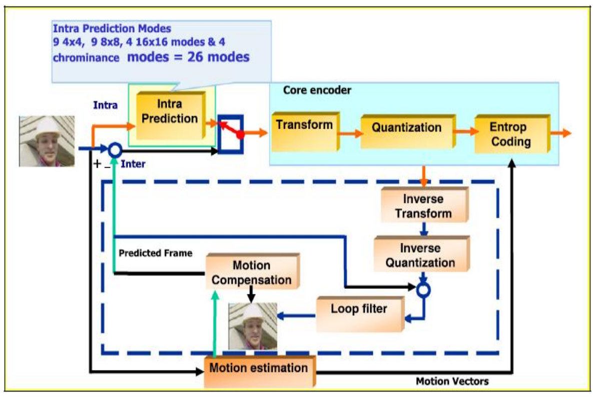

Fig 3: Blocks of H.264.

Figure 4 shows the different blocks in the intra prediction graph which is composed by 28 tasks.

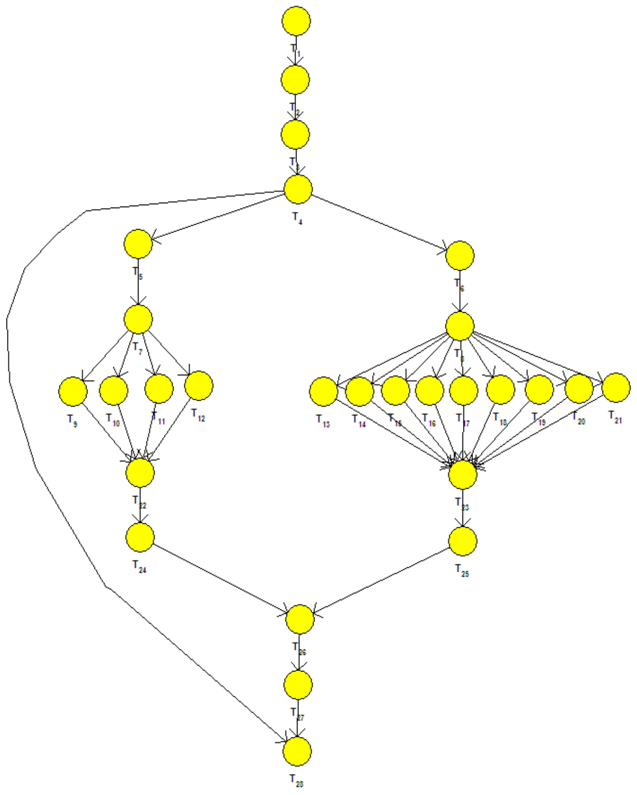

Fig 4: Intra prediction graph.

For the simulated data in Figure 5, Figure 6 and Figure 7, each algorithm was executed under Windows-7 on Acer-PC (Intel Core 2 Duo T5500; 1.66 GHz; 1GB of RAM) and has been written in JAVA language.

In our experiments, we have applied the algorithms based on Tabu search, genetic algorithm and two proposed algorithms that incorporate the BST respectively into Tabu search and genetic algorithm. [1].

The algorithms are implemented using the following parameters: for the genetic algorithm, the initial population consists of 10 individuals, and each individual contains 28 nodes, the probability of crossover is 0.6 and that of mutation is 0.1. For the case of Tabu search, the choice of initial solution is arbitrary. The size of the First In First Out list (FIFO) is the half of the number of tasks

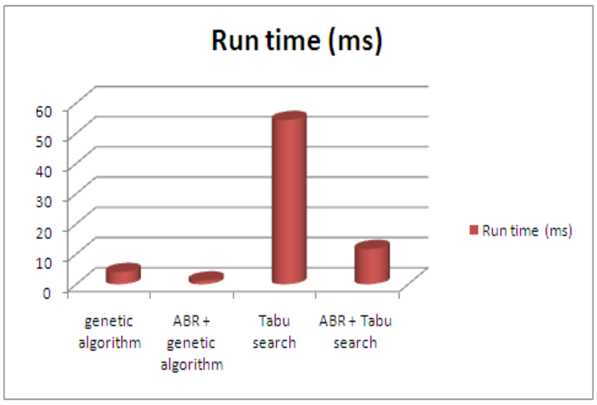

Fig 5: Run time

Figure 5 shows that the combined algorithms require less time to reach the final solution. Indeed, the first combined algorithm (BST + genetic algorithm) has a gain of 69% compared to the algorithm based on genetic algorithm and the other combined algorithm (BST + Tabu search) has a gain of 78% compared to the algorithm based on Tabu search

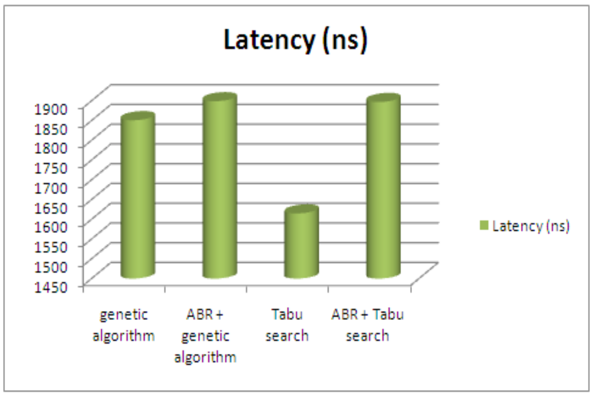

Fig 6: Latency

Figure 6 shows that the combined algorithms are slow in terms of application execution time. Thus the combined algorithm (BST + genetic algorithm) has a loss of 2.52% compared to the algorithm based on genetic algorithm and the combined algorithm (BST + Tabu search) has a loss of 14.86% relative to the algorithm based on Tabu search.

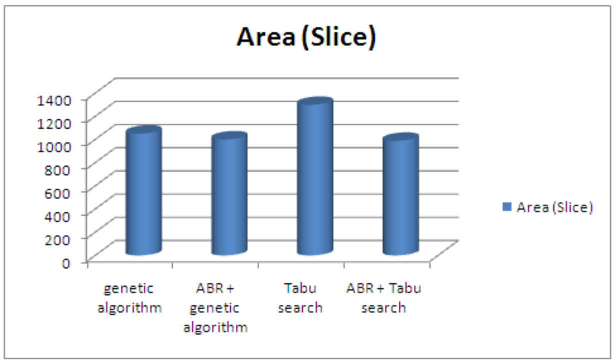

Fig 7: Used resources

Figure 7 shows that the proposed algorithms are more efficient in terms of area, because the combined algorithm (BST + genetic algorithm) has a gain of 4.75% compared to the algorithm based on genetic algorithm and the combined algorithm (BST + Tabu search) has a gain of 23.86% compared to the algorithm based on Tabu search.

According to the results presented above, we are unable to conclude the perfect efficiency of our method. So, to evaluate our method, we have introduced the following metric A_v:

$$Av = \frac{L}{A_{max} - A_L} \qquad \textbf{(1)}$$

A_{max}: all nodes of the graph are implemented to the hardware part of the architecture.

A_L: the logic area consumed by the graph

L: the whole latency of the graph

Therefore, based on the above equation, a partitioning algorithm is classified to be good if it decreases the value of A_v.

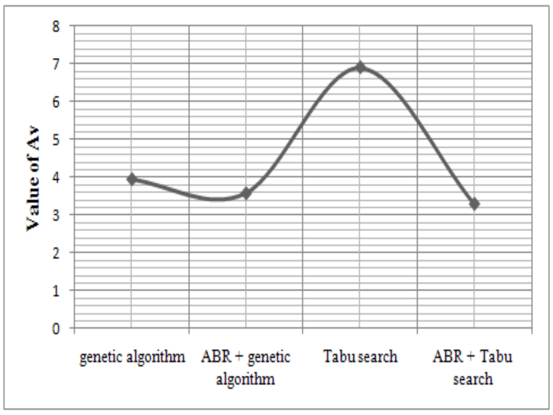

Fig 8: The metric A_v

Based on the above design results shown in figure 8, the proposed algorithms are the bests in terms of A_v value. Indeed, the combined algorithm (BST + genetic algorithm) provide a gain of 9.64% compared to the algorithm based on genetic algorithm and the combined algorithm (BST + Tabu search) has a gain of 52.46 % compared to the algorithm based on Tabu search.

6. CONCLUSION

Meta-heuristics are a set of approximate methods adaptable to a large number of combinatorial problems. They allow providing approximate solutions of good quality for a large number of classical optimization problems and for real applications having a large size. So whatever meta-heuristic used, it has become possible to solve combinatorial optimization problems having a large size. Therefore, these methods are currently being developed. In this context, we have presented a new approach based on Tabu search, genetic algorithm and Binary Search Trees to solve the problem of hardware-software partitioning in order to reduce the logic area on SOPC. The proposed algorithm [1] that incorporates the BST into Genetic Algorithm improves the complexity and the run time of the original genetic algorithm. The BST is used to reduce the search space and to have an optimized data access time. In fact, instead of performing the search in the whole BST, it will be done, on the left sub-tree or on the right sub-tree according to the time constraint. In this paper we have made a comparative study between the proposed algorithms and the algorithms based on genetic algorithm and Tabu search. The design result shows that our approach provides better design results in terms of the logic area. The prospects and future works include the apply of the proposed approach to intelligent algorithms

7. REFERENCES

[1] Sonia Dimassi, Mehdi Jemai, Bouraoui Ouni and Abdellatif Mtibaa,"Hardware-software partitioning algorithm based on Binary Search Trees and Genetic Algorithm to optimize logic area for SOPC", Published in Journal of Theoretical and Applied Information Technology (JATIT), Vol.66, No.3, August 2014.

[2] C.H. PAPADIMITRIOU, K. STEIGLITZ, Combinatorial optimization – algorithms and complexity. Prentice Hall, 1982.

[3] C.C. RIBEIRO, N. MACULAN (Eds.), Applications of combinatorial optimization. Annals of Operations Research 50, 1994.

[4] M.R. GAREY, D.S. JOHNSON, Computers and intractability: a guide to the theory of NP-completeness, W.H. Freeman and Company, New York, 1979.

[5] Bouraoui Ouni, Ramzi Ayadi and Abdellatif Mtibaa, "Combining Temporal Partitioning and Temporal Placement Techniques for Communication Cost Improvement" Advances in Engineering Software, Elsevier Publishers, Volume 42, no 7, July 2011, pp : 444-451.

[6] Bouraoui Ouni. , Ramzi Ayadi, and Abdellatif Mtibaa. "Temporal partitioning of data flow graph for dynamically reconfigurable architecture", Journal of Systems Architecture, vol 57, no 8, September 2011, pp 790-798

[7] Bouraoui Ouni and Abdellatif Mtibaa, "Optimal placement of modules on partially reconfigurable device for reconfiguration time improvement", Microelectronics International published by Emerald Group Publishing Limited, volume 29, Issue 2, 2012, Pages 101-107

[8] Ramzi Ayadi, Bouraoui Ouni and Abdellatif Mtibaa, "A Partitioning Methodology that Optimizes the Communication Cost for Reconguable Computing Systems" International Journal of Automation and Computing (IJAC), Institute of Automation and

Springer-Verlag Publishers, Volume 9, N° 3, June 2012, pp 280-287.

[9] Mehdi Jemai, Sonia Dimassi, Bouraoui Ouni and Abdellatif Mtibaa, "Optimization of logic area for System on Programmable Chip based on hardware-software partitioning", International Conference on Embedded Systems and Applications (ICESA) Hammamet-Tunisia,March 2014.

[10] K. Chatha and R. Vemuri, "Hardware-software partitioning and pipelined scheduling of transformative applications," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, no. 3, pp. 193–208, 2002.

[11] S. Banerjee, E. Bozorgzadeh, and N. D. Dutt, "Integrating physical constraints in hw-sw partitioning for architectures with partial dynamic reconfiguration," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 11, pp. 1189 –1202, nov. 2006.

[12] J. Wu and T. Srikanthan, "Low-complex dynamic programming algorithm for hardware/software partitioning," Information processing letters, vol. 98, no. 2, pp. 41–46, 2006.

[13] C.R. REEVES (Ed.) Modern heuristic techniques for combinatorial problems, Blackwell Scientific Publications, Oxford, 1993.

[14] E.H.L. AARTS, J.K. LENSTRA (Eds.), Local search in combinatorial optimization, John Wiley & Sons, 1997.

[15] J. Henkel and R. Ernst, "An approach to automated hardware/ software partitioning using a flexible granularity that is driven by high-level estimation techniques," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 9, no. 2, pp. 273–289, 2001.

[16] P. Eles, Z. Peng, K. Kuchcinski, and A. Doboli, "System level hardware/software partitioning based on simulated annealing and tabu search," Design Automation for Embedded Systems, vol. 2, no. 1, pp. 5–32, 1997.

[17] R. Dick and N. Jha, "Mogac: a multiobjective genetic algorithm for hardware-software cosynthesis of distributed embedded systems," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 10, pp. 920–935, 1998.

[18] G. LAPORTE, I.H. OSMAN, Metaheuristics in combinatorial optimization, Annals of Operations Research 63, J.C. Baltzer Science Publishers, Basel, Switzerland, 1996.

[19] I.H. OSMAN, J.P. KELLY (Eds.), Meta-heuristics: theory and applications, Kluwers Academic Publishers, Boston, 1996.

[20] Holland J.H. 1975. Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor.

[21] Goldberg D.E., 1987. Genetic algorithms in search, optimization, and machine learning, Addison Wesley.

[22] Glover F., 1986. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, 13, p. 533-549.

[23] Li-Wei Kang and Jin-Jang Leou."An error resilient coding scheme for H.264 video transmission based on data embedding+ ". Published in: Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on Volume 3, pp 257-260, 2004.

[24] Liangbao Jiao, Jing Zhou and Rui Chen. "Efficient Parallel Intra-prediction Mode Selection Scheme for 4x4 Blocks in H.264". Published in: Intelligent Computation Technology and Automation (ICICTA). IEEE International Conference on Volume: 2, pp 527 – 530, 2011.