Hindawi Publishing Corporation
Advances in Fuzzy Systems
Volume 2012, Article ID 290710, 9 pages
doi:10.1155/2012/290710

Research Article

BDD, BNN, and FPGA on Fuzzy Techniques for Rapid System Analysis

Rahul Dixit and Harpreet Singh

Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA

Correspondence should be addressed to Rahul Dixit, chayaji@hotmail.com

Received 19 February 2012; Accepted 12 September 2012

Academic Editor: Zeng-Guang Hou

This paper looks at techniques to simplify data analysis of large multivariate military sensor systems. The approach is illustrated using representative raw data from a video-scene analyzer. First, develop fuzzy neural net relations using Matlab. This represents the best fidelity fit to the data and will be used as reference for comparison. The data is then converted to Boolean, and using Boolean Decision Diagrams (BDD) techniques, to find similar relations between input vectors and output parameter. It will be shown that such Boolean techniques offer dramatic improvement in system analysis time, and with minor loss of fidelity. To further this study, Boolean Neural Net techniques (BNN) were employed to bridge the Fuzzy Neural Network (FNN) to BDD representations of the data. Neural network approaches give an estimation method for the complexity of Boolean Decision Diagrams, and this can be used to predict the complexity of digital circuits. The neural network model can be used for complexity estimation over a set of BDDs derived from Boolean logic expressions. Experimental results show good correlation with theoretical results and give insights to the complexity. The BNN representations can be useful as a means to FPGA implementation of the system relationships and can be used in embedded processor based multi-variate situations.

1. Introduction

Modern radar systems trace their roots to efforts in several countries to refine existing concepts and theories just prior to WWII. This was the culmination of theory and systems dating back to the 1860s. Simplistically, a transmitted electromagnetic signal reflects from an object back to an antenna where the characteristics of the signal determine properties such as location, speed and direction of the target. We take this simple problem, a show how variable complexity grows, and how the corresponding computational problem also grows.

Consider a simple Time-of-Flight range detector. For a stationary object, the range is simply determined by

$$R = \left(v_p \times t\right) 2, \tag{1}$$

where R = Range; v_p = Speed of propagation in medium (c in air); t = Time to receive the reflected signal.

For a moving target, radar systems use the Doppler Effect where the relative motion between source and observer produces a frequency shift

$$f = (v + v_r\, v - v_s) \times f_0, \tag{2}$$

where f = observed frequency; f_0 = transmit frequency; v = velocity of propagation in medium; v_r = velocity of target relative to medium; v_s = velocity of antenna relative to medium.

Both equations assume colocated transmit and receive (Tx/Rx) antennas. Azimuth and elevation information comes with knowledge of the antenna beam position. So, what started as a simple calculation with 2 variables, becomes a problem with 4 variables. And, as we add moving sensors, propagation media dielectric variations, frequency jitter, multipath echoes, and so forth, the number of variables grows, and the computation problem quickly overwhelms the processor. Modern day military sensors, with video and other multispectra imagers, capture such large amounts of data that embedded processors are rapidly overwhelmed.

The result then is that the embedded processor cannot complete its computations within the prescribed cycle time, and does not return a result or, starts to discard data, resulting in a false answer. Given that outcome, many systems find an "approximate" answer adequate for many mission objectives. In this paper, we employ data from such a video processor and show how Binary Decision Diagrams (BDD) fidelity compares with Fuzzy Neural Network (FNN) result quality and, how the Neural Network Model (NNM) can be used as an indicator of the Field Programmable Gate Array (FPGA) complexity.

The paper describes an environment for the rapid analysis, synthesis and optimization of embedded systems. The simplification strategy involves structure reduction and is carried out through an iterative algorithm aiming at selecting a minimal number of rules for the problem at hand. The selection algorithm allows manipulation of the neuro-fuzzy model or binary decision diagram to minimize complexity and to preserve a good level of accuracy. Since the implementation of these systems is rather complicated, we propose a methodology which automates the entire design flow. Flexibility is achieved by allowing manual intervention which is realized via a modular implementation of algorithms which are being provided.

2. Logic Analysis

An extension of the Time-of-Flight system is a typical track-while-scan (TWS) radar system, which gets a discrete (temporally separated) sequence of detections ("hits") on a given target. Each detection is of a finite, typically quite short,z duration. From each detection a given collection of attributes about that hit is extracted. The attributes typically include detection time, position, and reflected intensity, as well as various statistical moments and other mathematical features of the detection. The goal of these radar detections is typically to identify (detect and classify) the target of interest and its trajectory. By successfully tracking the target in motion, knowledge of the target can usually be substantially improved, since repeated detections of the moving target provide a greater volume of information and allow aggregate, track-based, statistics to also be used.

If we expand the problem to a multidimensional target identification problem, let us consider an example of a cluttered video image, and we need to find and classify the target in the scene. The image attributes include distance (to target it is called *distance*), aspect ratio of target (called *aspect*), vertical pixel height of target (called *vert*), area (in pixels it is called *area*) covered by target, target luminosity (called *target lum*), dark area in the image (called *dark area lum*), the surrounding luminosity (called *surround lum*), and edge pixels (called *edg pts*). The object is to classify the target (type) and assess the time taken to complete this chore. Reproduced from [1–3] is a table, Table 1, showing rows of such data. The data is just for illustration purpose only and is used in calculations to illustrate the comparison between full neuro-fuzzy analysis versus what target classification confidence we can achieve using Boolean or fuzzy techniques. The details of neuro-fuzzy techniques are not reproduced here [4], and, as background, the ANFIS (ANFIS = adaptive neurofuzzy inference system. As deployed in MatLab the Sugeno fuzzy model is where all output membership functions are singleton spikes and the implication and aggregation methods are fixed. The implication method is simply multiplication, and the aggregation operator just includes all of the singletons. The Sugeno method is ideal for acting as an interpolative supervisor of multiple linear controllers that apply different operating conditions of a dynamic nonlinear system. A Sugeno system is also suited for modeling nonlinear systems by interpolating multiple linear models) toolkit in MatLab and VeriMax analysis using MiniTab are employed. The architecture and learning procedure underlying ANFIS (adaptive-network-based fuzzy inference system) is a fuzzy inference system implemented in the framework of adaptive networks. By using a hybrid learning procedure, ANFIS can construct an input-output mapping based on fuzzy if-then rules and stipulated input-output data pairs.

And, as is shown [5, 6], we see that the Boolean and fuzzy approaches increase the linguistic rules as the number of input vectors increases. A neuro-fuzzy approach allows a more compact and computationally efficient representation and lends itself to adaptive schemes. And, where there is no intuitive knowledge of input vector behaviors or relationships to the output, these adaptive techniques in turn help create the entire fuzzy network for us.

Looking at Table 1, for this example, we see that there are 9 target types, and that there is little intuitive correlation between input vectors and target type or time taken to classify the target. Heuristic methods and expert knowledge (described in [7]), exploit *a-priori* knowledge to make inferences and help cluster the vectors and reduce system size by eliminating some inputs.

Using the Verimax analysis in MiniTab, we calculate, Table 2 [3], the correlation between vectors and see that vert (number of vertical pixels) and area (area covered by target in pixels) information is strongly correlated (95%).

There are other inferences that can also be made. But, none that allows a simple determination of logical connections. Employing the Neuro-fuzzy ANFIS function from the MatLab toolkit, we see in Table 3, that it is possible to get upto 87% confidence in target classification, relying on the full data set.

Where the terms as used by MiniTab are [8] as follows.

CORR: Correlation between the original output and the estimated output from the fuzzy neural system using the data from each method.

TRMS: total Root mean square for the distance between the original output and the estimated output using the same testing data through the fuzzy neural system

$$\text{TRMS} = \frac{\sum_{i=1}^{n} \sqrt{(x_i - y_i)^2}}{n-1}, \tag{3}$$

Table 1: Data reproduced from [1]. Data representing the target classification, and the various extracted attributes of the visual scene.

Target no. Type	Distance m	Aspect ass (sin)	Vert Pixels	Area (Pixels)	Target lum Scene	Dark area lum Dark	Surround lum Grass	Edgepts Pts	Search time Search time (s)
1	4007	0.71	10	141	14	17	29	9571	14.6
1	2998	0.82	11	225	21	10	27	8927	15.2
2	3974	0.71	13	173	20	24	28	9138	12.4
3	5377	0.05	5	49	18	23	30	8970	29.8
2	1013	0.52	50	2708	19	5	34	8706	2.8
4	3052	0.00	11	100	12	18	30	8755	6.4
5	5188	0.41	9	76	18	23	28	9053	26.7
6	3679	0.12	10	96	12	20	26	8620	10.0
2	860	1.00	54	3425	9	1.5	40	8961	2.7
4	1951	0.85	16	332	15	11	27	8572	2.8
3	3992	0.79	11	154	20	19	26	9194	11.9
6	1041	0.74	24	1645	11	4	35	9074	2.5
7	2145	0.98	17	553	8	5	18	8280	3.7
3	1998	0.76	19	659	20	10	22	8739	8.1
2	4410	0.00	11	101	22	18	29	9404	12.4
1	2893	0.42	16	320	12	7	23	8670	2.5
5	1933	0.98	13	368	15	12	23	8606	4.8
1	1850	0.96	28	876	3	4	9	8464	2.8
8	1045	0.09	26	985	19	10	12	8613	12.3
2	1933	0.95	22	867	16	11	27	8376	2.8
7	4206	0.00	9	79	26	29	38	9506	15.1
1	5722	0.88	7	73	38	40	46	9044	25.6
4	4920	0.42	8	61	20	21	36	8618	12.1
6	4206	0.81	9	142	18	12	21	9152	8.0
5	2348	0.94	9	198	18	21	30	8504	5.5
1	3992	0.88	11	217	15	14	26	9078	7.8
9	4410	0.96	11	247	16	8	19	9397	9.6
8	2321	0.83	15	458	22	21	47	8365	5.1
5	3661	0.76	9	84	17	25	23	8807	7.5
3	3670	0.00	13	192	14	15	27	8483	6.1
7	1671	1.00	19	893	15	13	31	8959	3.5
4	4345	0.81	8	63	15	12	20	9021	12.3
2	3662	0.57	10	203	26	25	44	8702	5.4
5	633	0.71	50	4403	20	5	39	8741	2.5
3	492	0.07	57	3045	20	16	23	8992	2.2
4	1497	0.78	16	560	10	7	20	9014	5.8
5	1041	1.00	33	1613	17	5	32	8486	2.6
1	2891	0.99	19	486	12	12	35	9021	12.1
7	5147	0.93	5	81	18	27	34	9075	34.9
6	1648	0.59	18	648	23	7	37	9070	2.7
8	948	0.73	35	1463	18	5	38	8790	3.7
7	3662	0.41	12	188	19	25	39	8524	5.8
6	2900	0.00	17	340	20	10	49	8791	4.1
2	5136	0.00	10	79	25	16	27	8941	10.6

where x_i is the estimated value and y_i is the original output value.

STD: Standard deviation for the distances between the original output and the estimated output using the same testing data through the fuzzy neural system.

MAD: Mean of the absolute distances between the original output and the estimated output using the same testing data through the fuzzy neural system.

EWI: The index value from the summation of the values with multiplying the statistical estimation value by its equally weighted potential value for each field.

Table 2: Reproduced from [1]. Verimax analysis using MiniTab tool to calculate covariance between input vectors.

	Distance	Aspect	Vert	Area	Target lum	Dark area	Surround	Edgepts
Distance	1	−0.24	−0.79	−0.72	0.37	0.72	0.06	0.41
Aspect	−0.24	1	0.08	0.12	−0.24	−0.25	−0.08	−0.10
Vert	−0.79	0.08	1	0.95	−0.19	−0.58	0.07	−0.19
Area	−0.72	0.12	**0.95**	1	−0.14	−0.53	0.15	−0.13
Target lum	0.37	−0.24	−0.19	−0.14	1	0.62	0.52	0.25
Dark area	0.72	−0.25	−0.58	−0.53	0.62	1	0.32	0.22
Surround	0.06	−0.08	0.07	0.15	0.52	0.32	1	0.04
Edgepts	0.41	−0.10	−0.19	−0.13	0.25	0.22	0.04	1

Table 3: Correlation and other analysis of data from [1–3].

	CORR	TRMS	STD	MAD	EWI	ERR
2 factor	0.77	5.73	3.60	5.35	14.92	13.44
3 factor	0.56	6.65	5.81	6.21	19.11	20.52
4 factor	0.46	7.59	14.39	7.09	29.62	16.55
5 factor	0.22	21.11	32.05	19.70	73.63	58.87
6 factor	0.71	6.92	10.46	6.46	24.13	18.03
7 factor	0.84	3.31	4.83	3.09	11.39	8.05
Original	**0.87**	2.84	4.40	2.65	10.02	7.38

ERR: The error rate is

$$\text{ERR} = \sum_{i=1}^{n} \left(\frac{E(i) - O(i)}{E(i)} \times 100 \right), \tag{4}$$

where n is the number of testing data, $E(i)$ is the estimated output, and $O(i)$ is the actual output.

This forms our baseline. We now compare this analysis to what we can get if the data in Table 1 is converted to Boolean. This Boolean data is presented in Table 4.

Verimax analysis when we employ Boolean relationships to determine the correlation functions shows correlation between the original output and the estimated output to be 82%. This compares well with the FNN analysis and is a reasonable approximation of the original data. It is this acceptable degradation that provides an opportunity to use Binary Decision Diagrams and Boolean techniques, to reduce the number of variables and to simplify the computational problem. The general correlation similarities between vectors are approximately preserved. And, again, *vert* and *area* vectors have high correlation. Thus, using the premise that an approximate answer is better than no-answer. The simple Boolean quantization of the raw data, provides a reasonable estimate as compared to the full data set, fuzzy inferences.

3. Boolean Decision Diagram Techniques

The analysis techniques used above show that reasonable computation reduction can be achieved by data quantization implying that using full-precision data as baseline, we still achieve approximately 82% fidelity in the correlation data using Boolean quantization of the data. This opens the door to using Boolean Decision tools for rapid, reasonable fidelity and analysis of the system.

Let us again start with the data in Table 4—which is the Boolean representation of the raw data. We find that we have 8 input variables and 1 output vector called search time, that is, the time required to correctly classify the target type. As shown in [2, 7], the correlation matrix for the Boolean data also shows that, because it has such high correlation between adjacent vectors, we can simply eliminate the *vert* or *area* vectors and, using a-priori knowledge for this specific problem, that *target lum* and *edge pts* vectors also can be combined in an OR function to some extent. Finally, that *distance* and *target lum* also can be OR combined to further reduce the matrix. We can then develop a 4-variable matrix to develop a set of linguistic rules.

Table 5 shows such a reduced vector representation of the original raw data. And, this is specific for this problem, other problems, and, using specialized knowledge [7] we can use the analysis techniques shown to reduce the input vector dimension.

Using Boolean Decision Diagram techniques, this is represented as

$$f = \sum(0, 1, 2, 3, 5, 6, 7, 10, 13, 14). \tag{5}$$

From this, we can create the K-map (Figure 1) of the above linguistic rules:

And, we can re-write f as:

$$f = B\overline{C}D + \overline{A}\,\overline{B} + C\overline{D} + \overline{A}C. \tag{6}$$

The complexity of circuit and systems design increases rapidly. Therefore, in seeking efficient algorithms and data structures, we choose, binary decision diagrams (BDDs). These have been used in a wide variety of applications and

Table 4: This Boolean data is presented in Table 4.

Target no. Type	Distance m	Aspect ass (sin)	Vert Pixels	Area (Pixels)	Target lum Scene	Dark area lum Dark	Surround lum Grass	Edgepts Pts	Search time Search time (s)
1	1	1	0	0	0	0	0	1	0
1	1	1	0	0	1	0	0	0	0
2	1	1	0	0	1	1	0	1	0
3	1	0	0	0	0	1	1	1	1
2	0	1	1	1	1	0	1	0	0
4	1	0	0	0	0	0	1	0	0
5	1	0	0	0	0	1	0	1	1
6	1	0	0	0	0	1	0	0	0
2	0	1	1	1	0	0	1	1	0
4	0	1	0	0	0	0	0	0	0
3	1	1	0	0	1	0	0	1	0
6	0	1	0	0	0	0	1	1	0
7	0	1	0	0	0	0	0	0	0
3	0	1	0	0	1	0	0	0	0
2	1	0	0	0	1	0	0	1	0
1	0	0	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0	0
2	0	1	0	0	0	0	0	0	0
7	1	0	0	0	1	1	1	1	0
1	1	1	0	0	1	1	1	1	1
4	1	0	0	0	1	1	1	0	0
6	1	1	0	0	0	0	0	1	0
5	0	1	0	0	0	1	1	0	0
1	1	1	0	0	0	0	0	1	0
9	1	1	0	0	0	0	0	1	0
8	0	1	0	0	1	1	1	0	0
5	1	1	0	0	0	1	0	0	0
3	1	0	0	0	0	0	0	0	0
7	0	1	0	0	0	0	1	1	0
4	1	1	0	0	0	0	0	1	0
2	1	1	0	0	1	1	1	0	0
5	0	1	1	1	1	0	1	0	0
3	0	0	1	1	1	0	0	1	0
4	0	1	0	0	0	0	0	1	0
5	0	1	1	0	0	0	1	0	0
1	0	1	0	0	0	0	1	1	0
7	1	1	0	0	0	1	1	1	1
6	0	1	0	0	1	0	1	1	0
8	0	1	1	0	0	0	1	0	0
7	1	0	0	0	1	1	1	0	0
6	1	0	0	0	1	0	1	0	0
2	1	0	0	0	1	0	0	1	0

were intensively studied. And, the Boolean Decision Diagram as Figure 2.

Having this Boolean Decision Diagram, we can apply the analysis tools [9–11] readily available for such analysis. This shows how a complex video scene, once it is translated into desired attributes, can then be analyzed using BDD techniques to yield fast approximate solutions to the data classification problem. Research exists [12] on how to generate FPGA using BDDs, and in this paper we acknowledge that numerous researchers have used BDD synthesis for FPGA

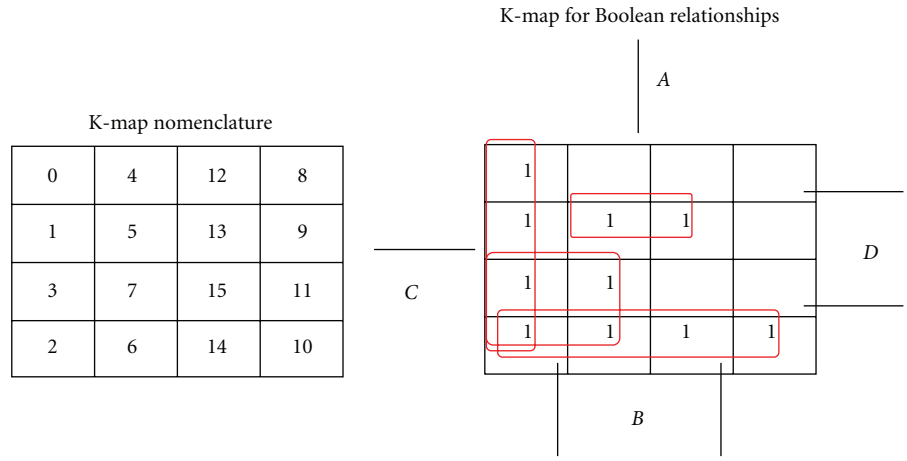

Figure 1: K-map of data represented by Table 5.

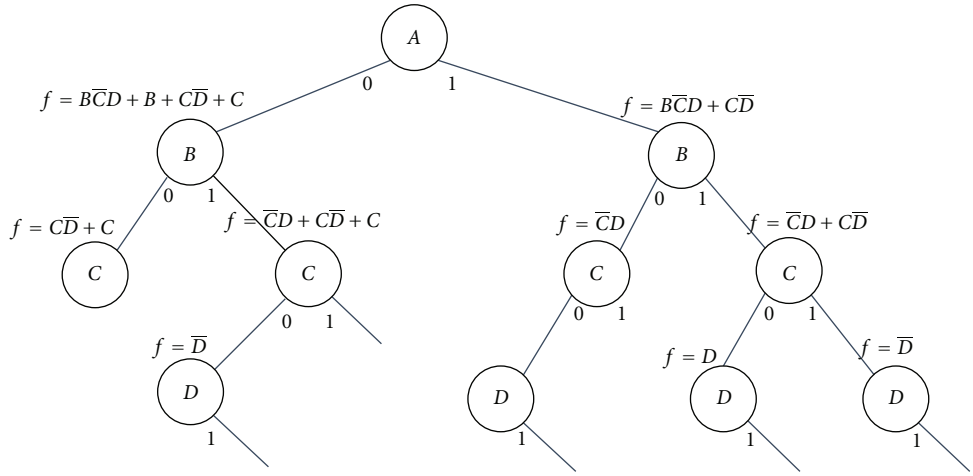

Figure 2: Boolean Decision Diagram for data represented by Table 5.

performance optimization. Additionally, recent publications [13–16] show that such order-reduction models are created and changed throughout the development process process and [13, 14] argue that if one looks at the idea of model checking as a constraint checking problem, then the BDD-based symbolic model checking indeed is capable of verifying systems with a large number of states. This is in agreement with our main thesis as well, that is the analysis of abstract models using such order-reduction analysis will have great impact in reducing computational load on embedded processors. The authors of [15, 16] note that it is indeed possible to reduce (replace) the initial set of input variables with their linear combinations and that such path reduction (order reduction) can be compared using spectral techniques, or weighted autocorrelation techniques, to compare and approximate accuracy degradations.

4. Proposed Algorithm

The proposed algorithm to implement the above technique is illustrated in this section. The input space is reduced and the number of rules is decreased, and the simulation results illustrate the approach is practicable, simple, and effective and the performance index is not significantly degraded. The video scenes, in this illustrative example, are first converted recognition attributes as required by the classification algorithm. And, using FNN and ANFIS techniques, a baseline answer fidelity index is generated. Then, the data streams are converted to Boolean and checked to see that resultant fidelity degradation is acceptable. This assessment can be rule-set or can rely on expert a-priori knowledge to judge when the quantization is acceptable. Then, using Factor and Cluster techniques, the problem size is reduced first by combining or eliminating the input vectors, and then, by selecting orthogonal data rows, such that a full set of outcomes is represented. Finally, use BDD techniques, to draw K-map diagrams and to develop a Boolean equation to solve the problem.

The synthesis, optimization, and implementation of embedded systems is rather complicated, and we propose a methodology which automates the entire design flow. Flexibility is achieved by allowing manual intervention which is

Table 5: Original visual scene data, after Boolean quantization, and after factor reduction.

Target type	A	B	F	G	Output
1	1	1	0	0	1
1	0	1	0	1	1
2	1	1	1	0	1
3	1	0	1	1	1
2	0	1	0	1	0
4	1	0	0	1	1
5	1	0	1	0	1
6	1	0	1	0	1
2	0	1	0	1	0
4	0	1	1	1	0
3	1	1	0	0	1
6	0	1	0	1	0
7	0	1	0	1	0
3	0	1	0	1	1
2	1	0	1	0	1
1	0	1	0	1	0

realized via a modular implementation of algorithms which are being provided. This is represented in the following flow (see Figure 3).

Assume there are n by p matrix with p-1 input variables and one output.

(1) Using the raw data, and FNN, ANFIS function from the MatLab toolkit, calculate the correlation matrix, R, between variables of data set. This correlation is the benchmark correlation between variables and can be used later to decide the validation of reducing procedure.

(2) Review input vector correlation data, to determine which variables can be reduced (or combined).

(3) Use iterative convergence to reduce/combine input vectors. Each time verifying that the resultant data is reasonably correlated to the original data.

(4) Convert remaining data stream to Boolean. Typically this binary representation is of a reduced set of input variables.

(5) Then, using Factor and Cluster techniques, the problem size is reduced first by combining or eliminating the input vectors, and then, by selecting orthogonal data rows, such that a full set of outcomes is represented. This can be performed, based on heuristic, a-priori, or correlation analysis.

(6) Repeat MiniTab/VeriMax analysis to validate that the resultant correlation value is within acceptable range as compared to the full-fidelity FNN correlation value. This step can be automated via a rule set, or can include judgment, depending on confidence in the factoring algorithms.

(7) Next, use BDD techniques, to draw K-map diagrams and to develop a Boolean equation to solve the problem.

(8) Create the BDD to represent the Boolean relationship.

(9) Verify the diagram with representative cases.

(10) Use FPGA design techniques, to convert the BDD to code.

(11) The system is now ready to take a raw stream of input data and yield a high efficiency solution to the target classification problem.

5. Analysis and Conclusion

Binary Decision Diagrams are generated by applying Shannon expansion [16] repeatedly to a logic function to yield a multivalued decision diagram. This is usually a (very) large matrix, generally unsolvable by embedded processors having finite memory and limited cycle-time before a decision must be rendered. Cluster decomposition (i.e., circuit partitioning) is also possible, but, generally this leads to several smaller matrices that must be solved simultaneously and also requires a fuzzy-logic arbitration algorithm to decide on the outcome. Further, when such subcircuits have fanouts, [16], the number of BDD nodes may increase exponentially thereby defeating the intent of Order Reduction algorithm. The study in [15] also confirms that the number of paths (order) is a function of the weighted autocorrelation coefficients and that it is possible to minimize the number of paths by constructing an ordered set of basis vectors of high weighted autocorrelation value. This is the key to use auto-correlation analysis to find relations between input vectors and output parameter, also to convert dynamic data to Boolean, that is, a 1-bit A/D approach, to create both a reduced-order circuit, and to simplify analysis. This technique of both order reduction (linearly combining input vectors) and input data-resolution reduction (converting from full-dynamic range data to Boolean data) still yields adequate quality results.

In summary, a high number of input parameters can result in a large number of rules that are computationally difficult to handle. Hence, it is imperative to develop the techniques, which can reduce the input parameters such that the original system and reduced systems have approximately the same behavior. In this paper, we have taken multi-variate streams of as-available data from a video scene analyzer and have created a technique for order reduction, and of digitization, such that the problem can reasonably be reduced to a Boolean problem, and, that this Boolean problem can then be solved using well-known techniques for Boolean Decision Diagrams. In the context of the example chosen for this paper, data streams from a video recorder are selected and analyzed for target classification efficiency. It is shown that an original 8 input-parameter data set with an 87% confidence full fuzzy neural net analysis, the reduced order problem was converted to a 4-variable problem, and the technique was able to generate a reasonably accurate 82% accurate

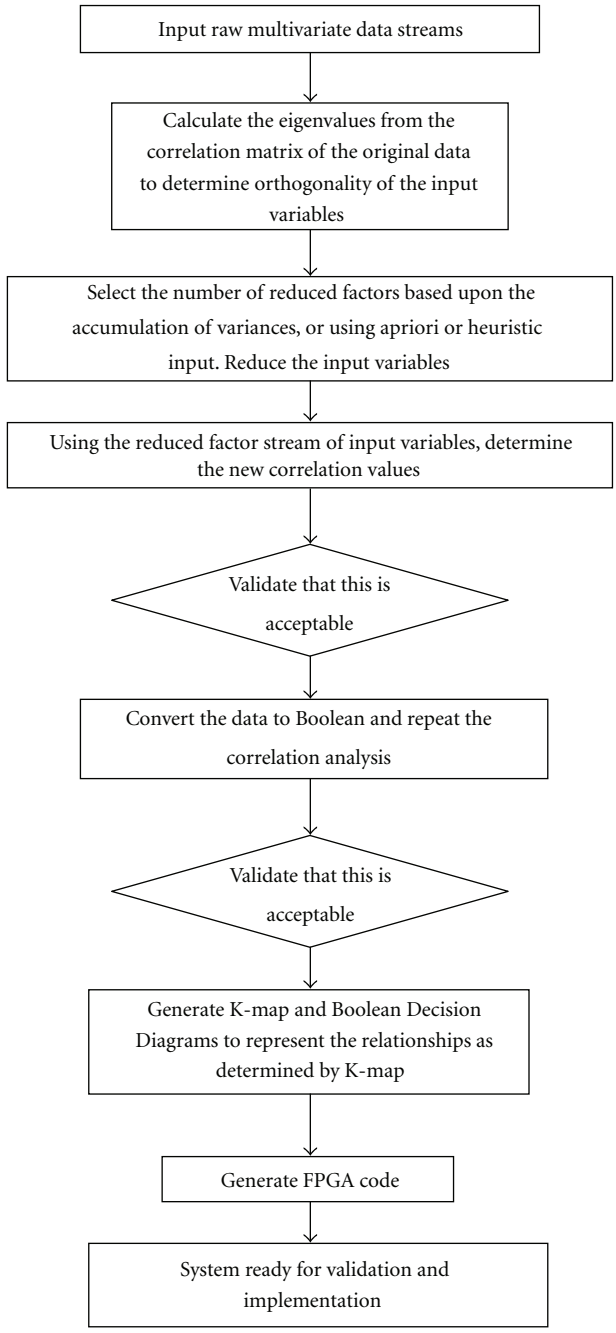

Figure 3: Flowchart for the proposed algorithm.

solution. And, that such a reduced-order problem can then be converted to a Boolean relationship. Existing research [9–12] shows how to convert this relationship to a FPGA design, such that, for this example, such video data can be quickly analyzed and a reasonably accurate answer generated. So while it is possible to have larger and larger processors and more and more complex code, often in many military applications, an approximate answer is adequate. And, that answer can be calculated using order-reduction techniques outlined herein, and then completely solved using BDD techniques.

From this study, we may decide the best heuristic and fuzzy techniques and also when the usage of fuzzy or heuristic method is better than the other. These are of significant advantage in satellite sensors where power and compute capacities are limited, and where such approximations can then cue more detailed analysis if warranted.

References

[1] T. J. Meitzler, L. Arafeh, H. Singh, and G. R. Gerhart, "Fuzzy logic approach for computing the probability of target detection in cluttered scenes," *Optical Engineering*, vol. 35, no. 12, pp. 3623–3636, 1996.

[2] T. J. Meitzler, H. Singh, L. Arefeh, E. Sohn, and G. R. Gerhart, "Predicting the probability of target detection in static infrared and visual scenes using the fuzzy logic approach," *Optical Engineering*, vol. 37, no. 1, pp. 10–17, 1998.

[3] T. J. Meitzler, E. Sohn, H. Singh, A. Elgarhi, and D. H. Nam, "Predicting search time in visually cluttered scenes using the fuzzy logic approach," *Optical Engineering*, vol. 40, no. 9, pp. 1844–1851, 2001.

[4] J. S. Jang, "ANFIS: adaptive network based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–684, 1993.

[5] Y. Xu and N. S. Chaudhari, "Application of binary neural networks for classification," in *2003 International Conference on Machine Learning and Cybernetics*, vol. 3, pp. 1343–1348, November 2003.

[6] E. C. C. Tsang, D. M. Huang, D. S. Yeung, J. W. T. Lee, and X. Z. Wang, "A weighted fuzzy reasoning and its corresponding neural network," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, pp. 605–610, October 2002.

[7] R. Dixit and H. Singh, "Heuristic approaches for embedded processor system size reduction," in *Proceedings of the International Conference on Data Mining, WorldComp'11*, pp. 192–197, Las Vegas, Nev, USA, July 2011.

[8] R. L. Gorsuch, *Factor Analysis*, Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 2nd edition, 1983.

[9] H. M. F. Madeira and J. Barbera, "Incremental evaluation of BDD-represented set operators," in *Proceedings of the 13th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'00)*, pp. 308–315, 2000.

[10] S. F. Mekhamer, M. E. El-Hawary, M. M. Mansour, M. A. Moustafa, and S. A. Soliman, "Fuzzy and heuristic techniques for reactive power compensation of radial distribution feeders: a comparative study," in *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (IEEE CCECE'02)*, vol. 1, pp. 112–121, May 2002.

[11] Y. Q. Zhang and F. L. Chung, "A fuzzy neural network tree with heuristic backpropagation learning," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '02)*, vol. 1, pp. 553–558, May 2002.

[12] P. Kraipeerapun, C. C. Fung, and K. W. Wong, "Quantification of vagueness in multiclass classification based on multiple binary neural networks," in *Proceedings of the 6th International Conference on Machine Learning and Cybernetics (ICMLC'07)*, vol. 1, pp. 140–144, chn, August 2007.

[13] A. Vakili, "Analyzing temporal properties of abstract models," in *Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering*, Lawrence, Kan, USA, November 2011.

[14] T. K. Nguyen, J. Sun, Y. Liu, and J. S. Dong, "A symbolic model checking framework for hieracrhical systems," in *Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering*, pp. 633–636, Lawrence, Kan, USA, November 2011.

[15] O. Keren, I. Levin, and R. S. Stankovic, "Determining the number of paths in decision diagrams by using autocorrelation coefficients," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 30, no. 1, pp. 31–44, 2011.

[16] H. Nakahara, T. Sasao, and M. Matsuura, "Multi-terminal multi-valued decision diagrams for characteristic function representing cluster decomposition," in *Proceedings of the 42nd IEEE International Symposium on Multiple-Valued Logic (ISMVL'12)*, pp. 148–1153, Vancouver, Canada, May 2012.