7. DECOMPOSITION OF FUZZY RELATIONS.

Multi valued relation is introduced in [13] as a table in which for certain combination of input variables values one of several specified output values can be selected. For instance, in Figure 15g in cell for $z = 1$, $G = 0$ there are two values, 0 and 1. It means that any ternary value other than value 2 can be taken for this combination of input variable values. This is called a generalized don't care and it generalizes a standard don't care concept where any set of values of a given output is allowed for given input combination. Thus, the generalized don't cares of a ternary signal are: {0,1}, {1,2} and {0,2}. The standard don't care is {0,1,2}. Let us observe that the generalized and standard don't cares correspond to the following values in fuzzy logic: {0,1} = x'_i or $x_i x'_i$ (when an undecided shape is between the one from Fig.2b and the one from Fig. 3a). {1,2} = $x_i x'_i$ or x_i (when an undecided shape is between the one from Fig. 3a and the one from Fig. 2a). {0,2} x'_i or x_i (when an undecided shape is between the one from Fig. 2b and the one from Fig. 2a). {0,1,2} when the shape of x_i is irrelevant. There are several ways to specify the initial fuzzy relations A graphical method is illustrated in Figure 15a. The OR relations among groups of terms denote that the choice of any of the groups of terms pointed by the two arrows originating from word OR can be made. Thus the function from Fig. 15 is specified by the expression: *F(x, y, z) = yz CHOICE-OF[x' y' z z ' OR (z z'x x' y' + z z' y y' x')] + xz.* In general, a fuzzy relation can be specified by an arbitrary multi-level decision unate function on variables G_I, each of these variables denoting Max of terms for a sum-of-products form of fuzzy relation. Such unate function uses functors AND and OR and variables G I corresponding to Max groups of terms. The above fuzzy relation is specified by the unate decision function: A AND B AND (C OR D) = (A AND B AND C) OR (A AND B AND D) where: A = *yz* , B = *xz*, C = *x' y' z z'* , D = *(z z' x x' y' + z z' y y' x').* Thus, every fuzzy relation corresponds to a set of sum-of-products fuzzy functions among whichwe can freely choose.

Example 8. Given is a fuzzy relation F_r *(x, y, z)* = *yz* + CHOICE-OF[x' y' z z' OR (z z' x x' y' + z z' y y' *x'*)] + *xz* , illustrated also in the map from Figure 16a. This is modification of Example in which more choices of fuzzy terms are given to the optimization tool. We specify that the tool has a freedom of choice between the groups of terms C = *x' y' z z'* or D = *(z z' x x' y' + z z' y y' x'),* which ever simplifies the final solution more.

For this fuzzy relation the map of ternary relation from Figure 15e is created by the operation of Maxing the ternary maps of functions xz (Fig. 15b), *yz* (Fig. 15d), and the map of the ternary relation corresponding to fuzzy relation [CHOICE-OF *x' y' z z'* OR (*z z' x x' y'* + *z z' y y' x'*)] (Fig. 15c). Observe that there are two entries, 0 and 1 in the cell $x = 0$, $y = 1$, $z = 1$ in Fig. 15e; this cell is called a generalized don t care and thus Fig. 15 stores a ternary relation, not a ternary function. The characteristic patterns found for Ashenhurst-like decomposition are encircled in Fig. 15e. Other patterns found are 011 and 0(0,1)0. The last pattern corresponds to either pattern 000 or to pattern 010. Thus, in any case there are three patterns, and the decomposition exists. Ternary function G after decomposition is shown in Figure 15f and ternary relation *H* is shown in Figure 15g . In general, both *G* and *H* can be relations in our approach, so our decomposition decomposes a relation to relations. Interestingly, sometimes also a function can be decomposed to relations. As we see, there is a choice of 0 and 1 in cell z=1, *G* = 0 in Figure 15g. Choice of value 0 (Fig. 15g*, H* = *GZ*) leads to the simpler solution from Figure 15h. Alternately, the choice of value in Fig. 15g leads to the more complex solution from Figure 15i, which was found earlier in Example 7, when function *F* was assumed instead of relation F_r . Transforming, when possible, a fuzzy function to a fuzzy relation, has thus a similar effect as replacing some of cares of a function by don't cares - it can be better minimized.

8. EXPERIMENTAL RESULTS.

We decomposed correctly all functions from [3,6] and from other papers on fuzzy logic and the computer times were negligible. The decomposer from can be set to any fixed number of values in all intermediate signals, so it is set to the value of three for ternary logic that corresponds to fuzzy logic. The decomposer from [14] decomposes to arbitrary-valued intermediate signals, in order to maximally decrease the total circuit's complexity and decrease the recognition error. It requires then encoding the signals that have more than three values to ternary vectors which is done by hand. For instance an intermediate signal with values 0, 1, 2 and 3 is encoded to two ternary signals as follows: 0 = [00], 1 = [01], 2 = [02] and 3 = [1X], where X means any of values 10, or 11, or 12. Thus, our encoding method introduces the don't cares and in general the relations to the MV data for decomposition. It proves thus that the concept of decomposing relations, introduced by us for Machine Learning and circuit design applications in program GUD-MV [12,13], is also useful for fuzzy logic. Currently we keep looking for more fuzzy logic benchmarks, especially large ones, but unfortunately all examples from books and conference proceedings that we were able to find are too small for the power of our decomposers. Perhaps the answer to this problem is to create large fuzzy data on our own. We intend to generate them automatically as the results of image processing procedures that create fuzzy features for pattern recognition experiments. Next our Constructive Induction approach to Machine Learning based on uniform approach to the decomposition of binary multi valued and fuzzy functions will be used in the final stage of pattern recognition instead of a Gaussian Classifier that we currently use [15,16]. Currently we are able to generate automatically multi-valued functions and relations from robot data (image and sensors) [24].

9. **CONCLUSION**

The new method of converting fuzzy functions to multiple-valued functions for decomposition allows not only for Ashenhurst-like, but also for Curtis-like decompositions. By converting fuzzy functions to multiple-valued functions we eliminate the time-consuming conversion to the canonical form. The need for special and complex methods like Kandel's decomposition method does no longer exist, and any existing MV decomposer can be used. Thus, various decomposers lead to different kinds of fuzzy functions decompositions. Our method can be expanded to arbitrary shape of fuzzy literals, and not only the literals x discussed above. Such an extension leads to multi-valued encodings of these fuzzy functions with logic radices higher than 3. In addition, our method can be used with no modification to relations. Several decomposers [12 -- 24] can be used for this task, again leading to different decompositions that can be evaluated and compared.

REFERENCES

[1] D. R. Luginbuhl and A. Kandel (1983) "A comparison of fuzzy switching functions and multiple-valued switching functions," Proc. IEEE International Symposium of Multiple-Valued Logic, 264 – 272.

[2] X. Zhiwei (1984) "Multivalued logic and fuzzy logic - their relationship, minimization, and application to fault diagnosis," IEEE Transactions on Computers, C-33(7), 679 - 681.

[3] A. Kandel and J. M. Francioni (1983) "Decomposable fuzzy valued switching functions," Fuzzy Sets and Systems, 9, 41- 68.

[4] L. A. Zadeh (1972) "A Fuzzy Set Theoretic Interpretation of Linguistic Hedges," J. Cybern., 2, 3, 4–34.

[5] M. Sakawa (1993) "Fuzzy Sets and Interactive Multiobjective Optimization," Plenum Press, NY,NY.

[6] A. Kandel (1974) "On the Minimization of Uncompletely Specified Fuzzy Functions," Information and Control, 26, 141-153.

[7] P. N. Marinos (1969), "Fuzzy logic and its application to switching systems," IEEE Transactions on Computers, C-18(4), 343-348.

[8] M. Karnaugh (1953) "The map method for Synthesis of Combinational Logic Circuits," Transactions AIEEE, 72, pt. I, 593 – 598.

[9] G. W. Schwede, and A. Kandel (1977) "Fuzzy maps," IEEE Trans. SMC-7(9), 699 – 674.

[10] E. W. Veitch, (1952) "A Chart Method for Simplifying Truth Functions," Proc. ACM, 127 – 133.

[11] M. Mukaidono (1984) "An Improved Method for Minimizing Fuzzy Switching Functions," Proc. IEEE International Symposium of Multiple-Valued Logic, 196 – 201.

[12] S. Grygiel and M. Perkowski (2001) ``Labeled rough partitions - a new general purpose representation for multiple-valued functions and relations", Journal of Systems Architecture*,* Vol. 47, 29-59.

[13] M. Perkowski, M. Marek-Sadowska, L. Jozwiak, T. Luba, S. Grygiel, M. Nowicka, R. Malvi, Z. Wang, and J. Zhang (1997) "Decomposition of Multi-Valued Functions and Relations," Proc. IEEE International Symposium of Multiple-Valued Logic.

[14] C. Files, R. Drechsler, and M. Perkowski (1997) "Functional Decomposition of MVL Functions using Multi-Valued Decision Diagrams," Proc. IEEE Intern. Symp. Multiple-Valued Logic,

[15] M. A. Perkowski, S. Wang, W. K. Spiller, A. Legate, and E. Pierzchala (1990) "Ovulo-Computer: Application of Image Processing and Recognition to Mucus Ferning Patterns," Proc. of the Third IEEE Symposium on Computer-Based Medical Systems, 52-59, Chapel Hill, North Carolina.

[16] M. A. Perkowski, S. Wang, W. K. Spiller, A. Legate, and E. Pierzchala (1990) "A Simple and Portable PC-based Image Processing System and its Application in Ovulometry," Proc. Imaging West Conference, 758-763, Pasadena, CA.

[17] A. Mishchenko, B. Steinbach, and M. Perkowski (2001) ``An Algorithm for Bi-Decomposition of Logic Functions," Proceedings of Design Automation Conference, DAC 2001*,* June 18-22, Las Vegas, 103 - 108.

[18] A. Mishchenko, B. Steinbach, and M. Perkowski (2001)``Bi-Decomposition of Multi-Valued Relations," Proc. 10-th International Workshop on Logic and Synthesis, IWLS'01 , 35 – 40.
[19] M. Perkowski, and S. Grygiel (2001) `` Decomposition of Relations: A New Approach to Constructive Induction in Machine Learning and Data Mining - An Overview" Proc. Workshop National Institute of Telecommunications and Polish Association for Logic and Philosophy of Science, 91 - 111, Warsaw, Poland.
[20] M. Perkowski, R. Malvi, S. Grygiel, M. Burns, and A. Mishchenko (1999) ``Graph Coloring Algorithms for Fast Evaluation of Curtis Decompositions," *Proc. DAC'99.*
[21] B. Steinbach, M. Perkowski, and Ch. Lang (1999) ``Bi-Decomposition of Multi-Valued Functions for Circuit Design and Data Mining Applications," Proc. ISMVL'99, 50 – 58.
[22] C. Files, and M. Perkowski (1998) ``Multi-Valued Functional Decomposition as a Machine Learning Method," Proc. ISMVL'98.
[23] C. Files, and M. Perkowski (1998) ``An Error Reducing Approach to Machine Learning Using Multi-Valued Functional Decomposition," Proc. ISMVL'98.
[24] U. Wong and M. Perkowski (2002) "A New Approach to Robot's Imitation of Behaviors by Decomposition of Mutiple-Valued Relations," Proc. 5th International Workshop on Boolean Problems, Freiberg University of Mining and Technology, Freiberg, Germany, 265-279.

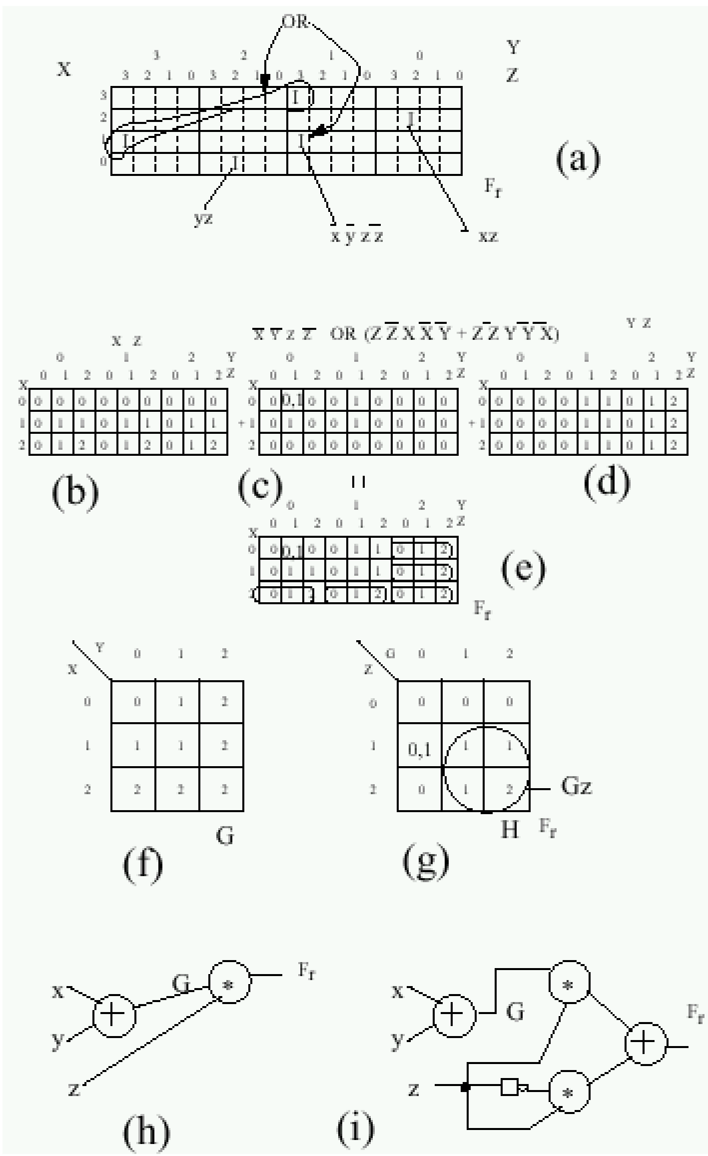

Figure 15. Stages of decomposition of a fuzzy relation to Example 8. (a) Original fuzzy relation F_r. (b) – (e) stages of creating a ternary relation corresponding to fuzzy relation F_r, (f) ternary function G from decomposition, (g) ternary function H from decomposition, (h) (i) Two realizations of fuzzy relation F_r , corresponding to two realizations of ternary relation H.