Application of Iterative Consensus to Multiple-Output Functions

Abstract—A limitation in the application of the technique of iterative consensus to multiple-output functions is discussed. A brief example is given, and a solution to the limitation is offered.

Index Terms—Iterative consensus, minimization, multiple-output functions, prime implicants.


1) Each row of the table is compared with each row above it in the table.

2) If any row is found to be included in another row (to have 1's wherever the other row has 1's and 0's wherever the other row has 0's), the included row is removed from the table.

3) If any two rows have a consensus, the consensus term is compared with all other rows of the table and then added to the bottom of the table if it is not included in any other row. Two rows have a consensus if there is only one column in which one row has a 1 and the other row has a 0. The consensus row has a dash in the column in which the two original rows differ and in any columns in which both the original rows have dashes. It has a 0 in any column in which either of the original rows has a 0 and a 1 in any column in which either of the original rows has a 1.

4) This process terminates when every row has been compared with all rows lower down the table. The rows which remain in the table correspond to all prime implicants. The consensus of any pair of rows either must appear as a row of the table or must be included in some row of the table.

For multiple-output functions the terms of the sum-of-products formulas are represented by identifier and tag elements. The algorithm is expanded to include multiple-output functions by addition of a fifth rule and with the statement that the algorithm developed for single-output functions can be directly extended to multiple-output functions by carrying out the algorithm, making no distinction between the tag and identifier portions of the rows in checking for inclusion and consensus terms. The fifth rule is quoted.

5) If two rows have identical identifier portions and differ in their tag positions, a new row is formed having the same identifier and dashes wherever either of the original rows have dashes in their tags. The original rows are then removed from the table.

An exception not covered by the above rules occurs whenever the identifier portion of one row includes the identifier portion of a second row, but where the tag element of the included row has a 0 in those columns where the including row has a blank. Consider the following fully specified 2-output function.

\[ F_1(w, x, y, z) = w'x'y' + x'y'z' \]

\[ F_2(w, x, y, z) = w'x'y'z' \]

In tabular form these are written as

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row</td>
<td>w</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Application of the algorithm provides no additions or changes to the prime implicants. A brief study of Karnaugh maps for these functions reveals 0000 -- is also a prime implicant (in fact it is required for 2-level minimization of the functions).

The difficulty exhibited by this example may be resolved by the following additional rule.

If the identifier portion of one row may be absorbed by the identifier portion of a second row, the tag elements associated with the absorbed identifier must be altered to reflect its application to those functions to which the absorbing row applies. In the above table Row 3 becomes 0000 -- and Rows 1 and 2 remain unchanged.

John M. Mage
Air Force Institute of Technology
Wright-Patterson AFB
Ohio

References


Computation of the Hadamard Transform and the R-Transform in Ordered Form

Abstract—This correspondence describes a method of computing the Hadamard transform and the R-transform. This method produces the transform components in the order of the increasing sequency of the Walsh functions represented by the rows of a symmetric Hadamard matrix for which the sequency of each row is larger than the sequency of the preceding row.

Index Terms—Hadamard transform, invariance with cyclic data shifts, pattern recognition, R-transform, Walsh-Fourier transform.

Walsh functions and the Walsh-Fourier transform have been discussed in recent articles [1]-[3]. Algorithms for computing discrete Walsh-Fourier or Hadamard transforms are described in [2] and [3]. The R-transform described in [4] may also be considered in terms of a Hadamard matrix. Its major property is that transformed data is independent of cyclic shifts of the input pattern. One form of computation of a one-dimensional R-transform is based on a flow graph which is similar to the flow graph used for the Cooley-Tukey FFT algorithm. This computation is the same as that shown for the fast Walsh-Fourier transform in [3] (Fig. 2), except that the absolute value is taken for each computation which involves both +1 and -1 multiplying factors. The fast Hadamard transform described in [2] provides the transform components in an "ordered" form, where these components are in the order of the sequency of the Walsh functions represented by the rows of a symmetric Hadamard matrix for which the sequency of each row is larger than the sequency of the preceding row. The algorithms described in [3] and [4] provide the components in a natural form (not in order of sequency). The new form of computation described here can be used to compute the Hadamard transform and the R-transform in an "ordered" form. It is for use with sampled data where the number of samples considered is an integral power of 2. The fast Hadamard transform of [2] is not suitable for use with the R-transform since it is not based on a computation starting with pairs of the N input samples which are separated by N/2 samples as required for invariance of the R-transform under cyclic permutation of the input samples.