C – Minimizer Algorithm: A New Technique for Data Minimization

Divya Nasa
M.Tech(CSE)
USICT, GGSIPU
Delhi, India

Udayan Ghose
Associate Professor
USICT, GGSIPU
Delhi, India

ABSTRACT
Digital gates are the basic components of the digital circuits. To reduce the cost of the circuit, number of these gates must be reduced, and hence a method is needed to do the desired. There are some methods such as Karnaugh maps, in which visualization becomes difficult when number of variables become more than five, and Quine McCluskey method, which overcomes the drawback of Karnaugh maps but it becomes complex when large number of variables are used. Here, in this paper, an approach, Consummate minimizer(C-minimizer) has been proposed which overcomes the limitations of such conventional methods and produces a minimized expression containing minimizing elements. The same can be used in minimizing the patterns in large data sets.

Keywords
Minimization, C-minimizer algorithm, Karnaugh Map, Quine McCluskey method.

1. INTRODUCTION
Minimizing the number of gates for the Boolean expression to be used in a circuit leads to reduction in the cost of the circuit. There are some methods which are used for the minimization of Boolean expression such as Karnaugh maps and Q-M method. Veitch[1] proposed a method "Map method" for boolean minimization and then a little modification was made by Karnaugh[2] in the above method, which was then named as "Karnaugh Map method" . Karnaugh maps use a graphical representation of Boolean equation so as to produce a simplified minimized expression. It searches for patterns to produce an output. This method holds the property of differing one variable in adjacent cells but suffers from a drawback of not able to effectively handle more than four variables[3]. Another method is Q-M, which is a systematic method for calculating the simplified expression. After Karnaugh Map, came another method which was a sequential approach, proposed by Quine[4] , which got optimized by McCluskey[6], and known as Quine McCluskey method. According to [8], while designing a relational database, Karnaugh map can prove to be a good method for finding all candidate keys out of the database but it breaks as the number of variables shoots to five or more than five. The advantage of Q-M method is that, it can holds for more than four variable where karnaugh map breaks[5]. The main hindrance in this method is the increase of complexity as the number of variables increases, because for this method the complexity is directly proportional to the number of variables used and hence becomes much complex for large number of variables. A proposed ''minimization approach'' is also a systematic method to compute minimized expression containing minimizing elements. This algorithm which is explained in the next section has been implemented using java and verified experimentally.

2. DATA MINIMIZATION
There are a set of related problems in the fields of data mining, knowledge discovery, and pattern recognition. If Artificial Neural Networks' (ANN's) are used in data mining process, it is not known that how many neurons should be in the hidden layer or the output layer. Thus if ANNs are used for clustering as a preliminary method for finding patterns, heuristic methods should be developed to determine the number of clusters. This is just another view of the problem in data mining of knowing how many patterns are there in the data and how these patterns can be discerned. There is a related problem in k-nearest-neighbors clustering in which an appropriate data structure is needed to efficiently find the neighbors of a given input vector. Indeed, if k-nearest-neighbors algorithm is to be used for classifying an input vector, the training input vectors need to be clustered first, and an ANN might have been used for this process. The problem of knowing the number of patterns and minimizing them is an interesting area in data mining. Also the Karnaugh maps can be used to minimize the patterns in data mining problems. Similarly the Q - M method can be used for this purpose. But the limitations of these methods still exist in the case of data minimization problem when multidimensional data sets are used.

3. C-MINIMIZER ALGORITHM

3.1 Algorithm
Step 1 : Input number of variables and truth table.

Step 2 : Initialize objects according to number of rows in truth table.

Step 3 : if F==0 then PUSH that element in el_zero_list

else if F==1 then PUSH that element in el_one_list

Step 4 : repeat step 3 till all elements are finished.

Step 5 : if SOP

then compare 1st element of el_one_list with every element of el_zero_list & make corresponding entries in the el_row_list.

else if POS

then compare 1st element of el_zero_list with every element of el_one_list & make corresponding entries in the el_row_list.

Step 6 : POP first node from el_next_list & compare every character of this node to every node of el_prev_list

if that character is contained in corresponding node

then copy that node in el_next_list.

else

append that character in string of that node, increase cost by 1 & then append it to el_next_list.

if cost in arriving node > encountered cost

Then Go to Step 8.

Step 7 : el_prev_list = el_next_list

Empty el_next_list

Go to Step 6 till all comparisons are done.

Step 8: for each node in el_prev_list having cost<= initial encountered cost and node.checked =false,

if weight of any character in node can give remainder zero when divided by weight of every node of el_row_list

then Go to Step 9.

Go to Step 7.

Step 9: minimized element = string in node of step 8.

Step 10: Delete el_one_list nodes containing the minimized element.

Step 11: repeat from Step 6 till el_one_list is empty.

Step 12: output = sum of minimized elements.

Step 13: End

3.2 Brief Explanation of Algorithm

In the algorithm, F is the Boolean function which is to be minimized. Here, rows of the truth table are segregated on the basis of output of given function as '0' or '1' and are put in the linked list el_zero_list or el_one_list respectively. Small characters (a,b,c,d….) are used to indicate '0' and capitals(A,B,C,D…) are used to indicate '1'. el_row_list is the list of nodes having strings, which contains those characters which are different in nodes getting compared. Cost in "step6" refers to the number of variables in the el_prev_list and el_next_list on the basis of which minimized element will be computed, whereas encountered cost is the highest cost encountered yet.

By using this algorithm both SOP (Sum Of Products) and POS(Product Of Sum) forms of any Boolean expression can be computed easily. SOP is formed by combining the minterms, while POS is formed by combining the maxterms[7].The above written algorithm has been explained by taking an example of four variables and the results are verified using Karnaugh map in the following section.

3.3 Example of SOP

The algorithm can take input as follows:

F= Σ or Π 0 ,1, 2 ,8 , 9 + d (5 , 7 ,10)

Number of variables = 4

According to the given Boolean function, truth table is generated as shown in Table 1.

Table 1: Truth Table

Elements	A	B	C	D	F(output)
Z_0	0	0	0	0	1
Z_1	0	0	0	1	1
Z_2	0	0	1	0	1
Z_3	0	0	1	1	0
Z_4	0	1	0	0	0
Z_5	0	1	0	1	0
Z_6	0	1	1	0	0
Z_7	0	1	1	1	X
Z_8	1	0	0	0	1
Z_9	1	0	0	1	1
Z_{10}	1	0	1	0	X
Z_{11}	1	0	1	1	0
Z_{12}	1	1	0	0	0
Z_{13}	1	1	0	1	0
Z_{14}	1	1	1	0	0
Z_{15}	1	1	1	1	X

Now, segregation of the elements of truth table(represented as 'Z') is done. Elements with value '1' and '0' are put in the linked list named el_one_list and el_zero_list respectively and d(don't care) are neglected as shown in Figure 1.

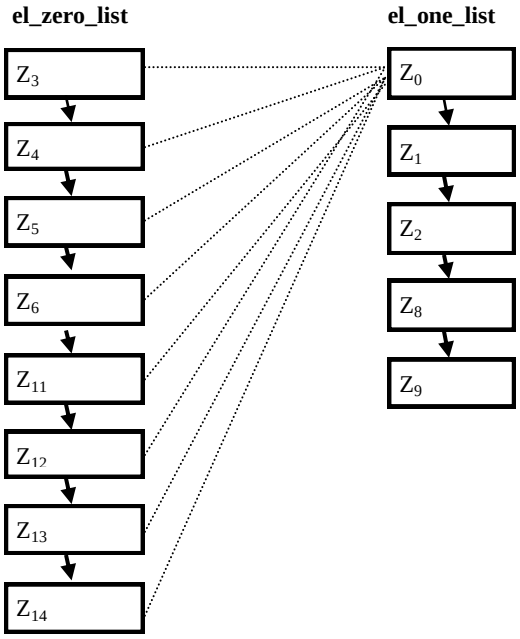

Figure 1: Separation of elements of truth table

Taking first element i.e. Z_0 from el_one_list and a prime weight starting from 2,3,5,7.....so on is assigned to all the variables in minterm Z_0 and is then compared with Z_3, Z_4,Z_5, Z_6, Z_{11}, Z_{12}, Z_{13}, Z_{14} one by one and comparisons are saved in another linked list named el_row_list as per algorithm.

The node of this linked list will contain a comparison string and weight of this string is calculated by multiplying the weight of each character contained in the string. This is shown in the following diagrams in the form of a matrix for understanding only, but the implementation has been done using linked lists named el_prev_list & el_next_list as per algorithm, hence reducing the space complexity.

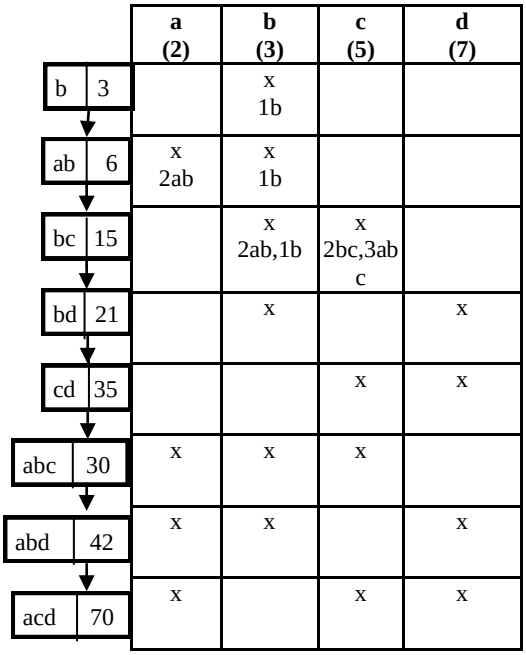

Figure 2: Matrix I

After comparison the list el_row_list in "Fig. 2" contains b-3, 3 is the weight of b, ab-6 ,6 comes after multiplication of weight of a & b and till acd- 70 this follows. Taking first node b-3 from el_row_list , and comparing it with nodes of el_prev_list. Since initially el_prev_list is empty, hence entry for each character is made in el_next_list as 1-b.

Now make el_prev_list = el_next_list , and then empty el_next_list and delete this node of el_row_list.

Again by doing the same thing taking node ab-6 and comparing each character of this node (a,b) with nodes of el_prev_list and making corresponding entries(2ab,1b) in el_next_list. This is how the whole process goes. When node 2ab is formed, cost in arriving node= 2 but encountered cost=1. Hence making condition in "Step 6" true and therefore by following the algorithm, weight of nodes of el_prev_list are divided by the weights of nodes of el_row_list to get minimized element if there. Repeating these steps for every node in el_row_list till minimized element is obtained. Here obtained minimized element is "bc". Now delete all the nodes of el_one_list having value of second variable and third variable as '0'. Repeating the process for rest of the nodes left in el_one_list shown below in Figure 3.

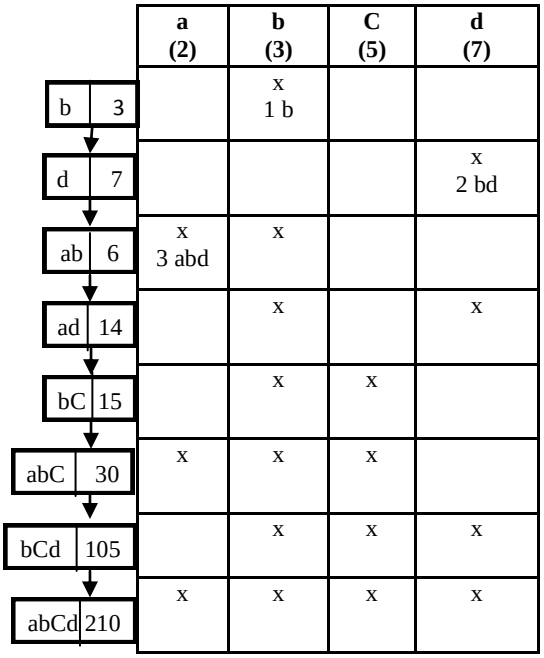

Figure 3: Matrix II

From Figure 3 the matrix minimized element is "bd". Deleting the corresponding nodes from el_one_list , making it empty and hence given minimized expression is "F = bc + bd".

Now, verification of this output using Karnaugh maps, taking the same function F is done and drawing Karnaugh map for that function leads to the following diagram as shown in Figure 4.

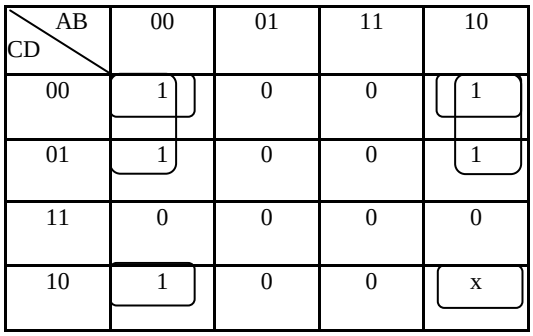

Figure 4: Karnaugh Map I

In Figure 4 combinations of 1's are made, two groups, including don't cares if needed, are formed as shown. So, minimized SOP form is 'bc+bd', which matches that of the above algorithm.

3.4 Examples of POS

Now again the same function F= Π 0 ,1, 2 ,8 , 9 + d (5 , 7 ,10) is taken and POS form will be calculated with the same procedure as of SOP with a little difference that instead of el_one_list, el_zero_list is to be considered. The solution can be shown with the help of the following figures and will be verified using Karnaugh map.

	a (2)	b (3)	C (5)	D (7)
C 3			x 1C	
D 6				x 2 CD
aC 15	x 3 aCD		x	
CD 21			x	x
aCD 42	x		x	x

Figure 5: Matrix III

From Figure 5 minimized element is CD which is expressed in POS as (c+d).

	a (2)	B (3)	c (5)	d (7)
B 3		x 1B		
aB 6	x 2aB	x 1B		
Bc 15		x	x	
Bd 21		x		x
Bad 42	x	x		x

Figure 6: Matrix IV

From Figure 6 minimized element is B which is expressed in POS as (b). So total minimized element or minimized POS form is (b)(c+d).

Now, POS form is calculated using Karnaugh map and output will be compared to the above computed output. In the Figure 7 combinations of zero's are made, two groups, including don't cares if needed, are formed as shown. One group is of size eight and second is of size four. So the minimized POS form would be (b)(c+d) which matches with the above computed output.

CD \ AB	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	0	X	X	0
10	1	0	0	X

Figure 7: Karnaugh Map II

4. RESULT AND CONCLUSION

The algorithm can be used to compute SOP or POS in minimized form of the given Boolean expression. It has been implemented and tested for multiple random variables or dimensions in a large data set. In Table 2, average computation time (in milliseconds), taken in useful pattern, for minimization of fifteen variables has been shown. This result has also been seen graphically, which has been shown in the Figure 8. According to [9], run time of Q-M method for higher number of variables has been computed as 109 ms, 703 ms , 5921 ms, 54140 ms, 491250 ms for 7,8,9,10, 11 number of variables respectively. But from Table 2, it has been observed that C-minimizer take 379.66 ms, 806 ms, 1716 ms, 3564.5 ms, 7316.33 ms for 7,8 , 9, 10, 11 number of variables respectively. Now, comparing these values it can be seen that for 7 and 8 number of variables , the values are nearly equal but for higher number of variables the difference between values of run time changes drastically. This shows that performance of C- minimizer is good for higher number of variables. This algorithm can be used in various applications in the fields of various industries and service sector such as networking, bioinformatics, image compression techniques, educational data, population data etc. to find out relevant information by minimizing the data set into a sizeable and informative data

Table 2: Results

Number of Variables	Average Value(ms)
3	15
4	16
5	32
6	166.33
7	379.66
8	806
9	1716
10	3564.5
11	7316.33
12	14976
13	30599.5
14	62415.6
15	127265

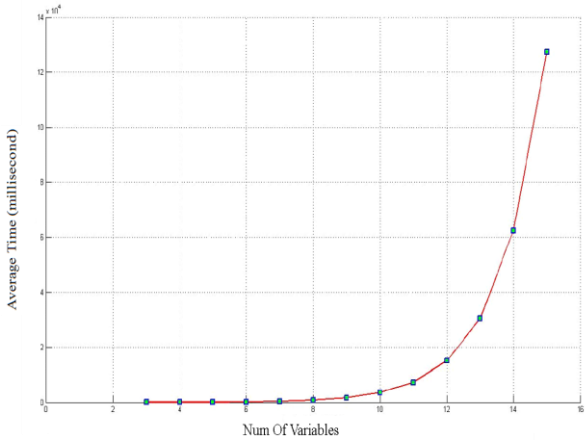

Figure 8: Graph between number of variables and average time of computation

5. REFERENCES

[1] E.W. Veith, "A Chart Method for Simplifiying Truth Functions." , Proc. of the ACM, pp. 127-133, 1952.

[2] M. Karnaugh, 'A Map Method for Synthesis of Combinational Logic Circuits', Trans. AIEE, Comm. And Electronics, Vol.72(1), pp. 593-99, 1953.

[3] K. J. Dean, "An Extension of the use of Karnaugh Maps in the minimization of logical functions", IEEE in Radio and Electronic Engineer , vol. 35, pp 294-296, may 1968.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[4] W.V. Quine, "The problem of simplifiying Truth Functions", Am. Math. Montly, vol.59(8), pp.521-31,1952.

[5] P. W. Chandana Prasad, A. Beg, A.K. Singh," Effect of Quine-McCluskey simplification on boolean space complexity", Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA) Monash University, Sunway campus, Malaysia, July 2009.

[6] E. J. McCluskey, "Minimization of Boolean Functions," Bell System Technical Journal, vol. 35, no. 5, pp. 1417-1444,1956.

[7] M. E. Holder," A Modified Karnaugh Map Technique" IEEE Transactions on Education, vol. 48, No. 1, February 2005.

[8] Z. Yi-Shun," Determining all candidate keys based on Karnaugh Map", International Conference on information management, innovation management and industrial engineering, 2009.

[9] T. K. Jain , D. S. Kushwaha and A. K. Misra," Optimization of the Quine-McCluskey Method for the Minimization of the Boolean Expressions" Fourth International Conference on Autonomic and Autonomous Systems.