An Algorithm for the Construction of Decision Diagram by Eliminating, Merging and Rearranging the Input Cube Set

Prof. Sudha H Ayatti
Department of Computer Science & Engineering
KLS GIT, Belagavi, Karnataka, India

Prof. Sharada M Kori
Department of Computer Science & Engineering
KLS GIT, Belagavi, Karnataka, India

Prof. Veena K Lalbeg
Department of Computer Science & Engineering
KLS GIT, Belagavi, Karnataka, India

Prof. Kavya Agalatakatti
Department of Computer Science & Engineering
KLS GIT, Belagavi, Karnataka, India

Abstract- This proposed work is DDS are a data structure that allows compact representation of discrete functions Boolean functions. The construction of DDS in terms of memory and time is considered problems we proposed method of eliminating, merging and reordering the set of cubes in matrix specification that results in the reduction of both memories occupied and time complexities of the construction of DDs. First we employ elimination algorithm followed by merging and then again elimination algorithm and reordering the set of cubes in this way, the number of operations on the nodes is reduced. This reduction results in a decrease both in the number of temporary nodes and construction time the experiments show that the total number of created nodes is reduced on average by 35% and construction time is decreased by 49%.

Keywords: -cubes, decision diagram.

I. INTRODUCTION

DDS provide compact representation of discrete functions. Due to its versatility in the areas of simulation of logical functions, VLSICAD, DDS and programming packages for their construction and manipulation are now days a standard part of many related CAD systems.

Binary DDS were originally inverted for hardware verification to efficiently store a large number of states that share many commonalities. DDS have been used in symbolic model checking circuit synthesis, polynomial algebra, finite state machine, integer linear programming prime generation, matrix representation data compression, program analysis and many others areas.

II. RELATED WORK

A Binary decision diagram is a directed acyclic graph with a single root node and two terminal nodes which represent the constants 0 and 1. This graph represents a Boolean function over a set of input decision variables each nonterminal node T in the graph is labeled with an input decision variable and has exactly two outgoing edges a higher edge and a low edge. We specifically use a variant of binary decision diagrams called reduced ordered binary decision diagram or reduced-ordered binary decision diagrams.

III. GENERAL DESCRIPATION

The prime purpose of this paper is that the binary decision diagram is well known for its simplicity and efficiency. The proposed technique depends on principle of regrouping which means placing the nodes closer to each other having maximum matching's, because of this it will minimize the number of transitions.

The general constraints are:-

1. The binary decision diagram must be saved in excel file as considered as input.

2. The binary decision diagram must be represented as connection matrix.
3. First we need to apply elimination algorithm followed by merging algorithm and finally once again we need to apply elimination algorithm.

IV. SPECIFIC REQUIREMENTS

Functions requirements of the end product:-

1. User friendly.
2. Graphical representation of space and time complexity.
3. Analysis by applying various algorithms.
4. List of reduction in time and memory to construct reduced ordered binary decision diagrams in terms of percentage.

V. SOFTWARE SYSTEM REQUIREMENT

The software requirement specification is produced at the calculation of the analysis task. The function and performance allocated to software as part of system engineering are defined by establishing a complete information description as functional representation a representation of system behavior, an indication of performance requirements and design constraints, appropriate validation criteria, graphical user interface is required to provide user interface.

VI. HARDWARE SYSTEM REQUIREMENTS

Processor: Pentium 4
RAM: 128MB
Hard disk: 200MB

ELIMINATION RULE

```
For all V V (i)
{
    If (id (low (v)) =id (high (v))
    {
      Remove V from v (i)
Redirect all incoming edges of v to low
       (v)
      Remove V;
}
 Else {
      Key (v) = (id (low (v), id (high (v)) ;
  } }
```

MERGING RULE

```
Old key = (0, 0)
For all V V (i) sorted by key (v)
  {
   If key (v) = old key
  {
  Remove V from V (i)
Redirect all incoming edges of V to old node
  Remove V
}
Else
  {
    Old node = V
     Old key = key (v)

}
}
```

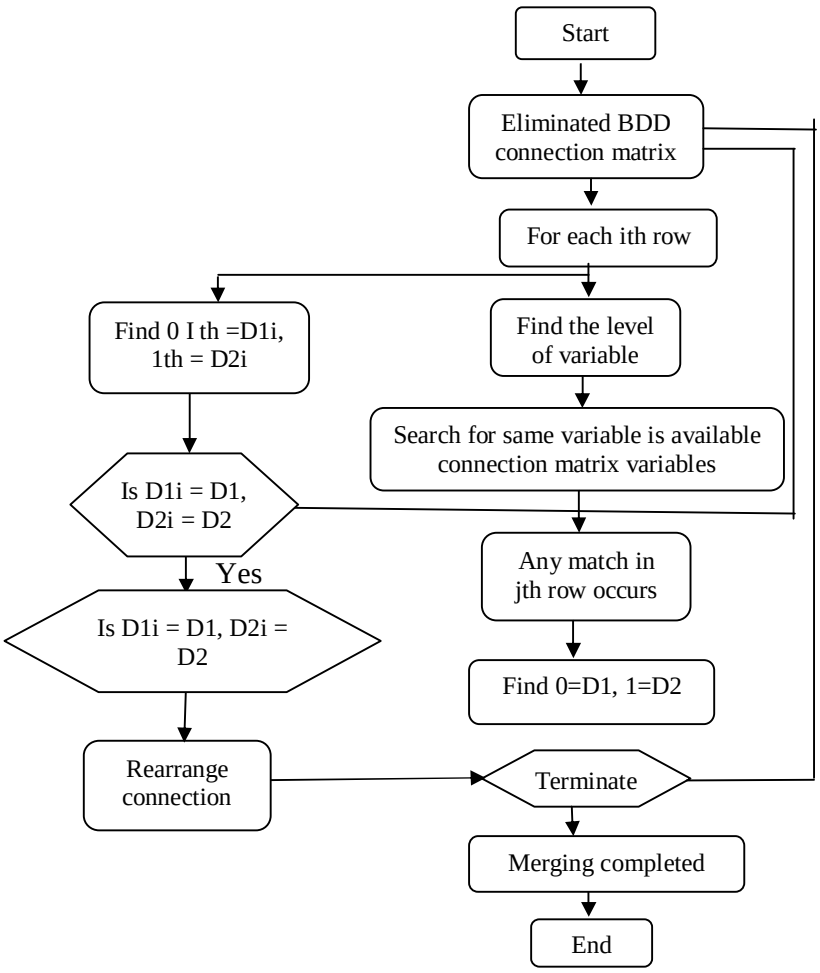

Fig1. Merging algorithm

VII. REARRENGING THE INPUT CUBE SET

1. Determine the matching matrix M for the Input cube set.
2. Compute the total matching vector TM
3. Determine the cube K with the maximum tank.
4. Write into as output file all cubes J for Which mk (0) starting from the cube with the maximum Mkj to the cube with the Minimal mkj the corresponding tmj is set to zero.
5. Repeat step 3 and 4, until the vector TM Contains non zero elements

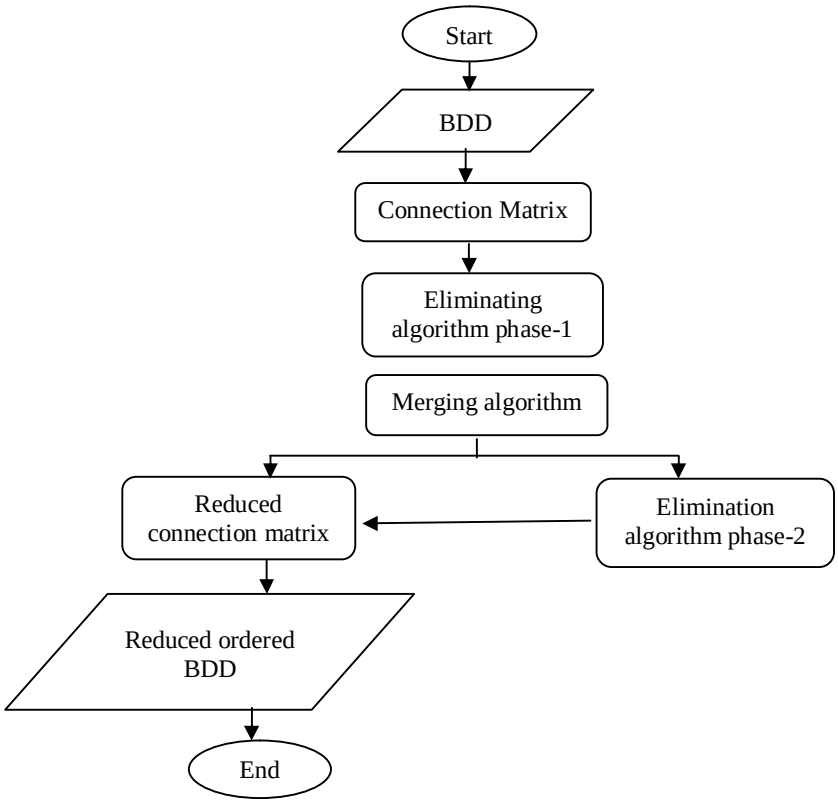

Fig2. Data flow diagram

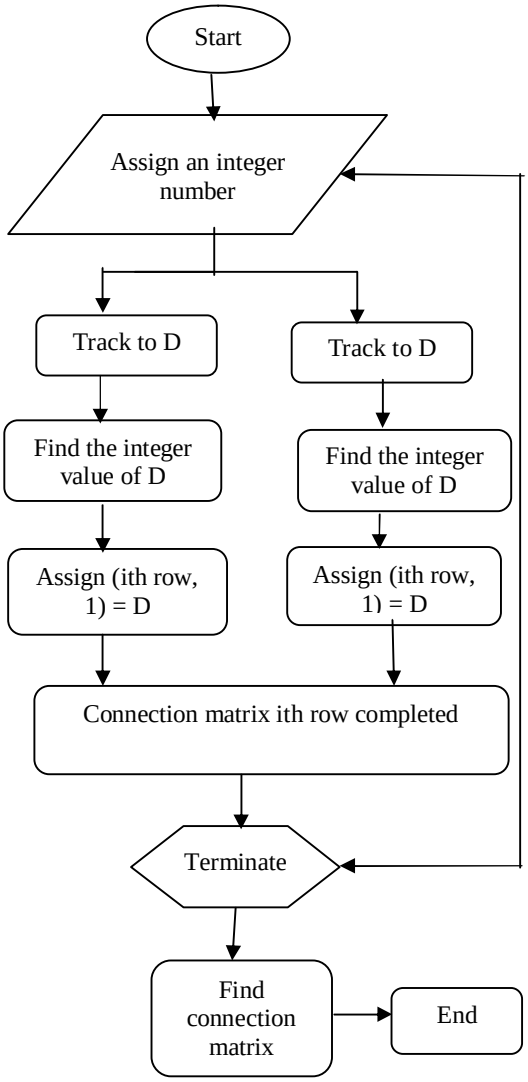

Fig 3. Connection matrix

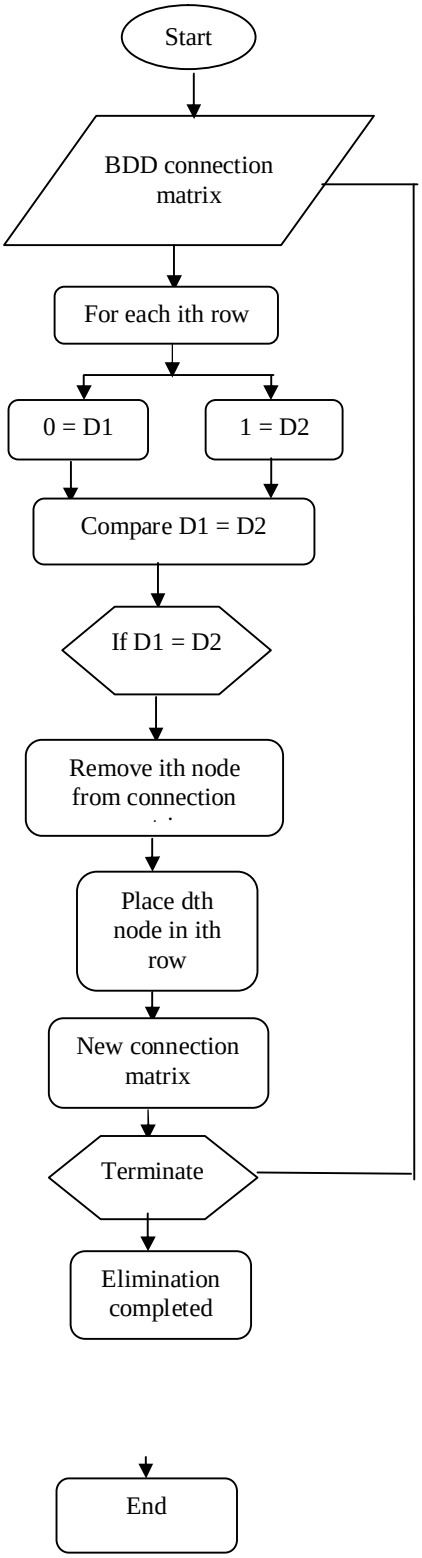

Fig4. Elimination

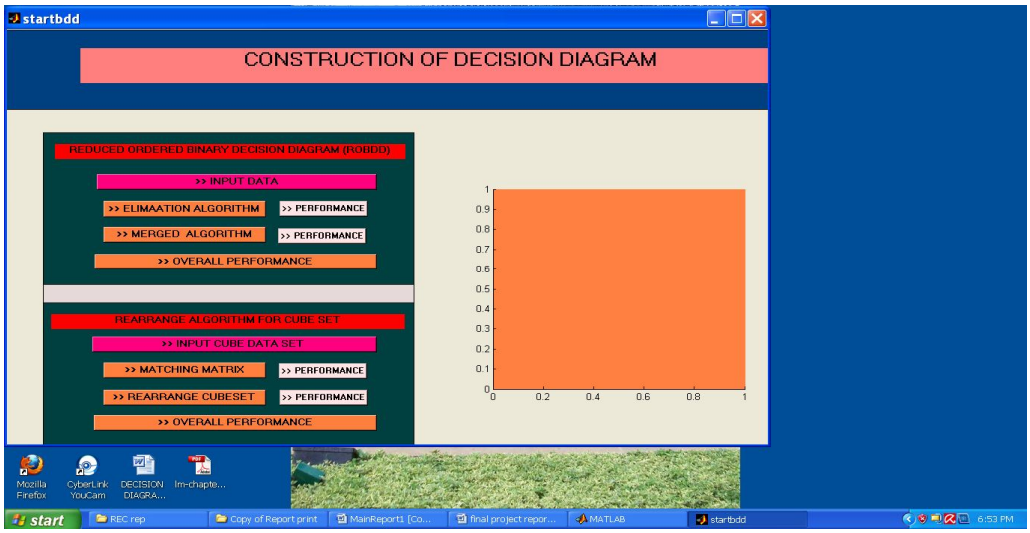

Figure B.1 Main form

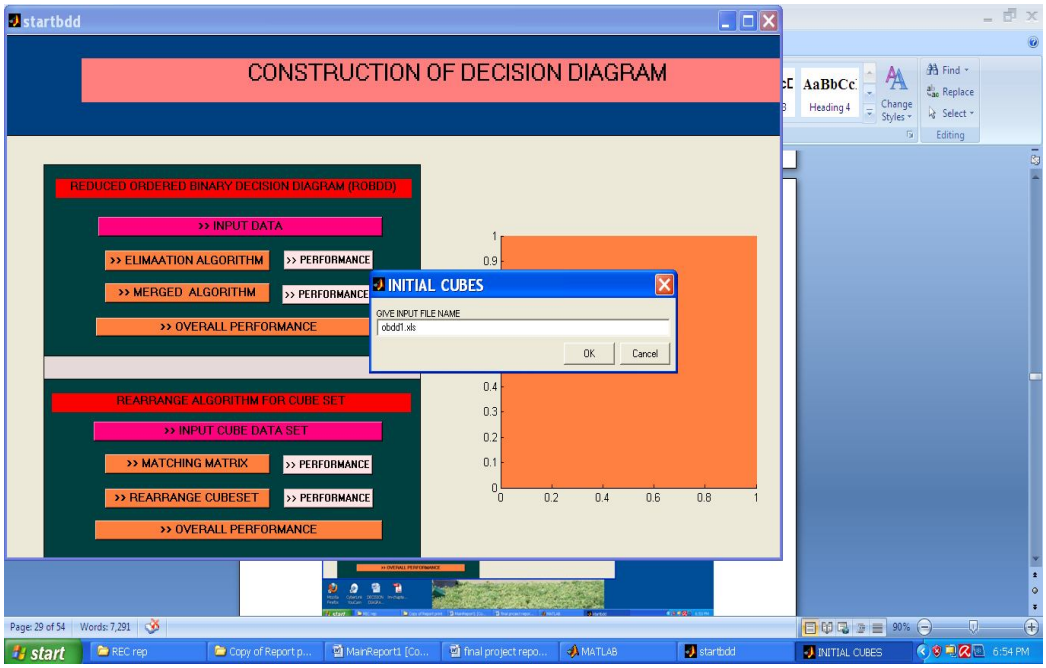

Figure B.2 applying the input data

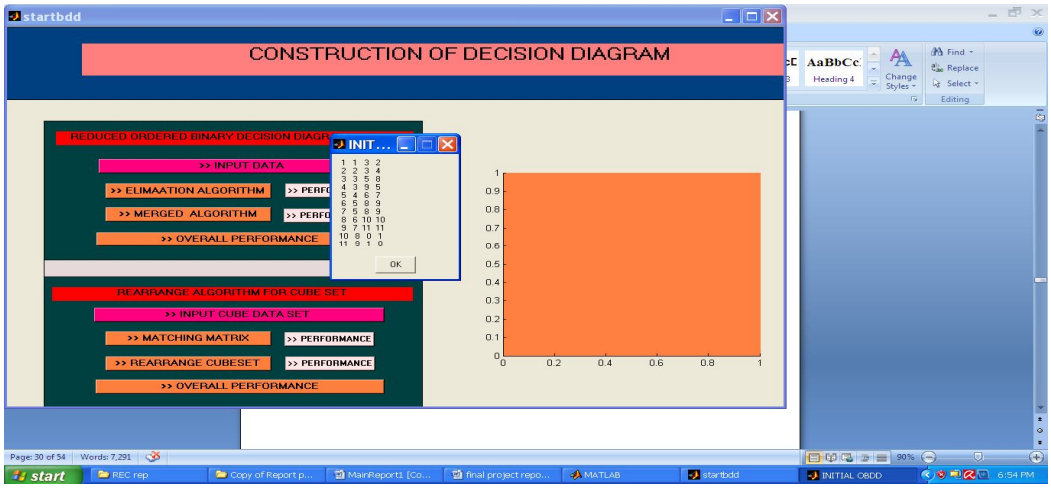

Figure B.3 original data from the file

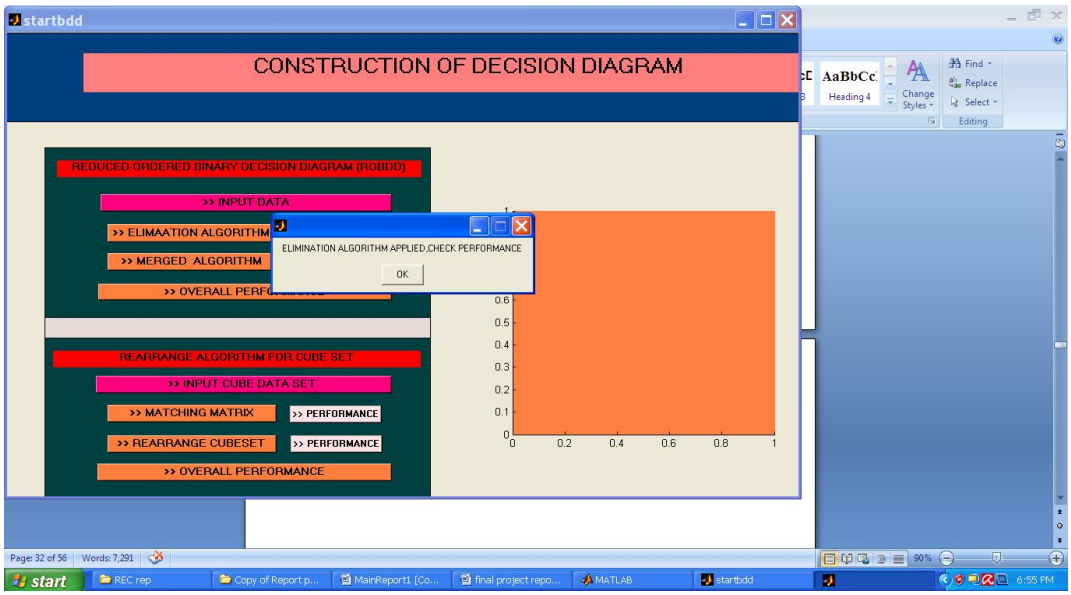

Figure B.4 applying the elimination algorithm

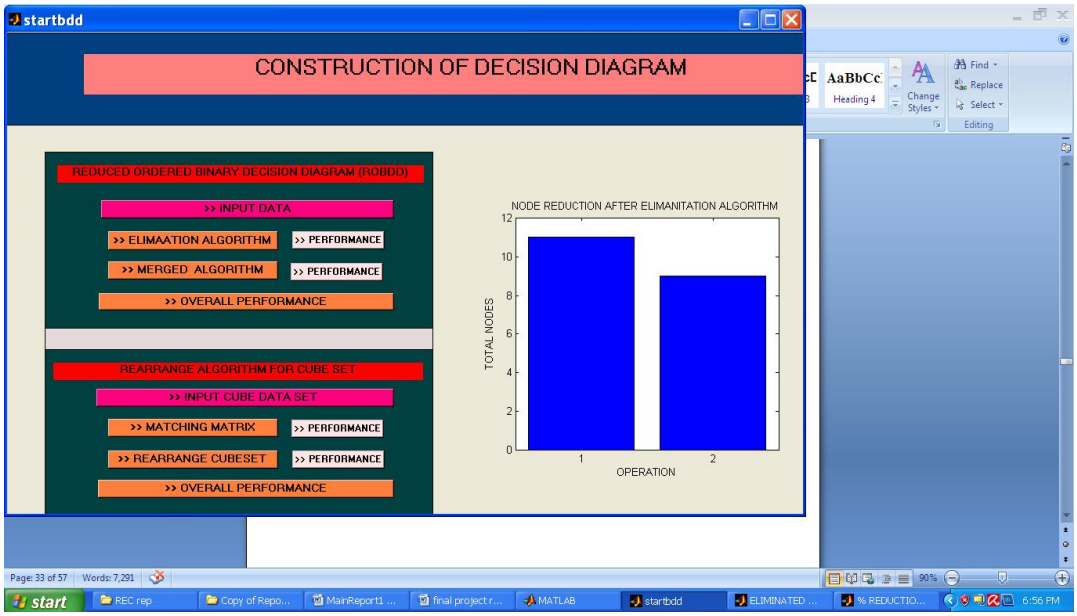

Figure B.5 Performance check after applying the algorithm

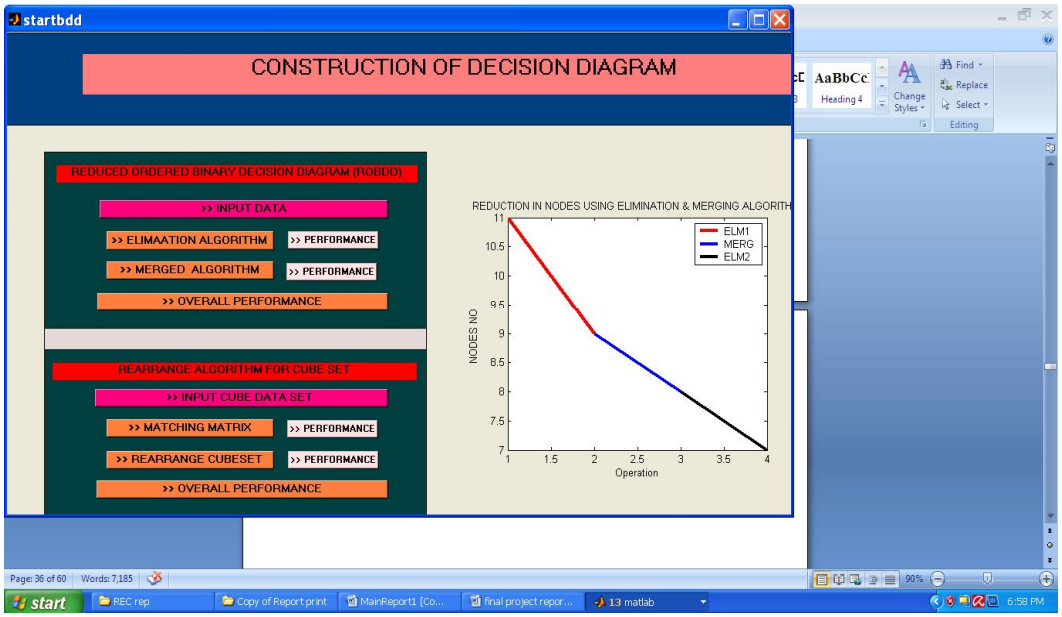

Figure B.6 Overall performance

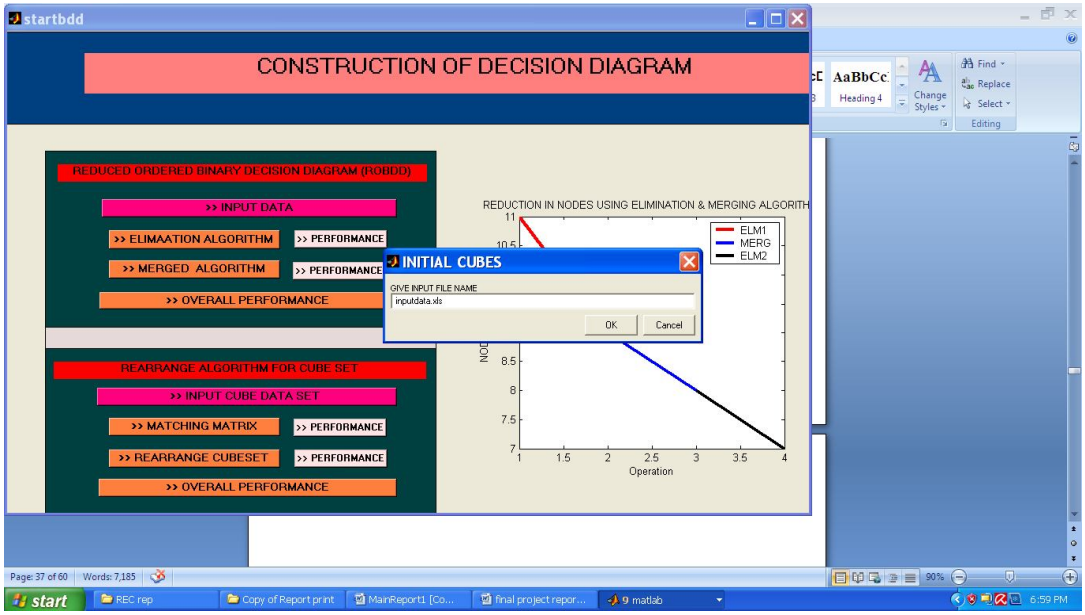

Figure B.7 Input cube set data

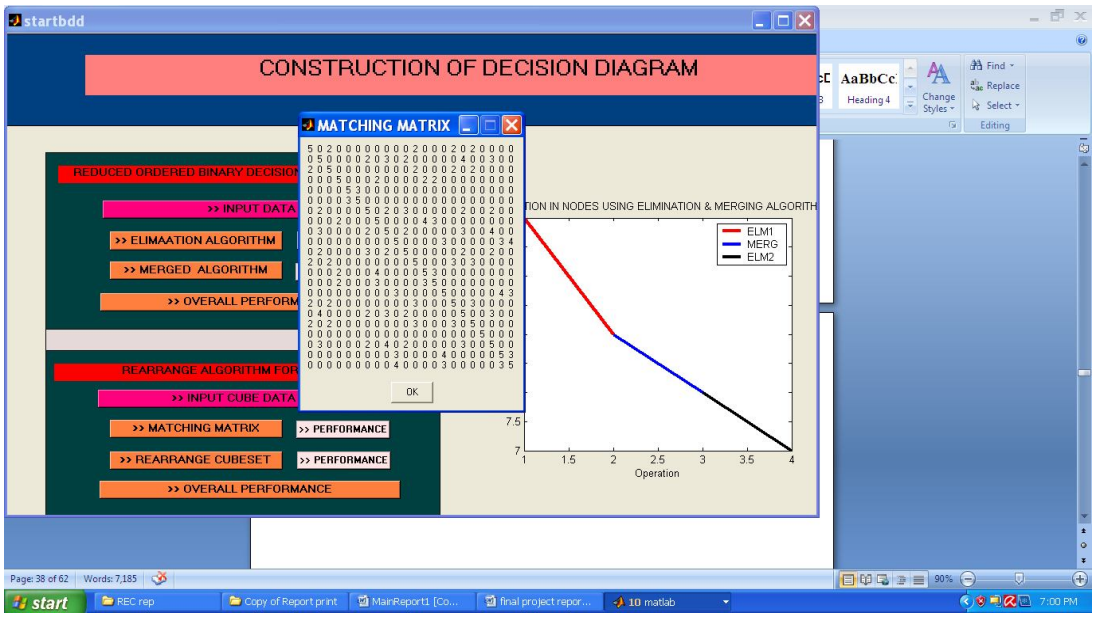

Figure B.8 matching matrix

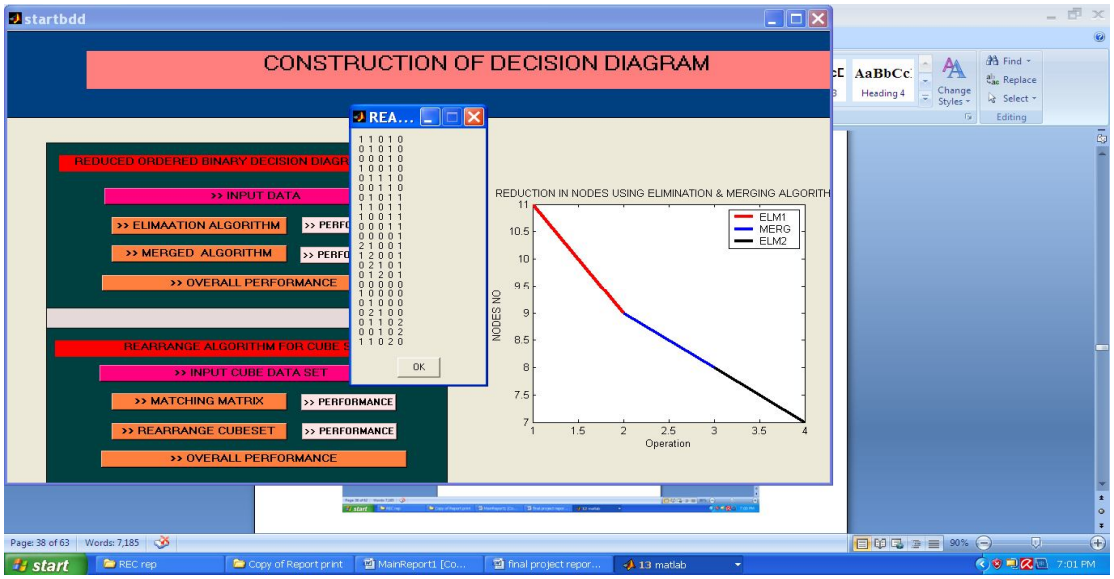

Figure B.9 Overall performance after applying the algorithm

VII. CONCLUSION

There are secured technologies available at present that minimizes memory and processing time but they are not cost efficient. The resulting binary decision diagram and its types will be cost effective with high data availability and efficiency, this application can be installed an any PC with the software MATLAB 6.5.

REFERENCES

[1] K.S. Brace, R.L. Rudell, and R.E. Bryant, “Efficient Implementation of a BDD Package,” Proc. Design Automation Conf., pp. 40-45, 1990.

[2] R.E. Bryant, “Graph-Based Algorithms for Boolean Function Manipulation,” IEEE Trans. Computers, vol. 35, no. 8, pp. 677- 691, Aug. 1986.

[3] R. Drechsler and B. Becker, “Ordered Kronecker Functional Decision Diagrams—A Data Structure for Representation and Manipulation of Boolean Functions,” IEEE Trans. CAD, vol. 17, no. 10, pp. 965-973, Oct. 1998.

[4] R. Drechsler and B. Becker, Binary Decision Diagrams: Theory and Implementation. Kluwer Academic Publishers, 1998.

[5] G.D. Hachtel and F. Somenzi, Logic Synthesis and Verification Algorithms. Kluwer Academic Publishers, 1996.

[6] D.M. Miller, “Spectral Transformation of Multiple-Valued Decision Diagrams,” Proc. 24th Int’l Symp. Multiple-Valued Logic, pp. 89-96, 1994.

[7] D.M. Miller and R. Drechsler, “Implementing a Multiple-Valued Decision Diagram Package,” Proc. 28th Int’l Symp. Multiple-Valued Logic, pp. 52-57, 1998.

[8] S. Minato, Binary Decision Diagrams and Applications for VLSI CAD. Kluwer Academic Publishers, 1996.

[9] R.S. Stankovi_c and J. Astola, Spectral Interpretation of Decision Diagrams. Springer-Verlag, 2003.

[10] S. Stojkovi_c, “UDDP—Universal Decision Diagram Package,” Acta Electrica et Informatica, vol. 5, no. 1, pp. 34-41, 2005.