

## Mirror, Mirror, on the Wall ... Is the New Release Any Different at All?

Justin E. Harlow III  
 Dept. of Electrical and  
 Computer Engineering, Box 90291  
 Duke University,  
 Durham, NC 27708, USA  
<http://www.ee.duke.edu/>

Franc Brglez  
 CBL (Collaborative Benchmarking Lab)  
 Dept. of Computer Science, Box 7550  
 NC State University  
 Raleigh, NC 27695, USA  
<http://www.cbl.ncsu.edu/>

**Abstract** – *Reduced, Ordered Binary Decision Diagrams (here, simply BDDs) have been adopted as an important data structure for a number of applications, ranging from logic design verification to logic minimization and technology mapping. However, for a number of functions that arise in practical applications, the size of the BDD data structure depends critically on the choice of the variable order: a poor order may lead to a data structure whose size grows exponentially with the number of variables. Finding an order that minimizes the size of BDDs is NP-hard; the development of heuristics for computing better variable orders is an ongoing area of research. In this paper, a Design of Experiments methodology is applied to the study of the behavior of software packages which construct and manipulate BDDs.*

**Keywords:** design of experiments, circuit equivalence classes, benchmarking, BDD.

### 1 INTRODUCTION

Several suites of BDD manipulation subroutines have been widely distributed; among the most popular of these are the three BDD packages [1, 2, 3] in the VIS [4] formal verification system from the University of California at Berkeley. In this paper, we present new results on the use of Design of Experiments (DoE) in studying the variable ordering heuristics employed in these BDD packages and in VIS itself. This paper focuses on a systematic evaluation of the quality of the variable ordering algorithms in VIS, and on drawing statistically significant conclusions about differences in performance between a new VIS release and previously published results from an earlier release. The introductory paper of this session [5] reviews the central ideas of DoE, as they apply to the evaluation and comparison of CAD algorithms.

In earlier papers [6, 7], we presented results which demonstrated that (1) the order and labeling of the source netlist has a dramatic impact on variable ordering algorithms, (2) the static ordering heuristic in VIS was inoperative for combinatorial circuits, (3) repeated application of dynamic variable reordering, even for small, regular circuits with well-known optimal orderings, often cannot find the optimal ordering, and (4) there are statistically significant differences in performance among the three BDD packages, even on small test cases. Recently, the VIS group released VIS 1.3. We have repeated a number of our earlier tests on this release, and now present results comparing the relative performance of the new release with the earlier VIS 1.1.

Justin Harlow was supported in part by National Semiconductor Corporation. Franc Brglez was supported by contracts from the Semiconductor Research Corporation (94-DJ-553), SEMATECH (94-DJ-800), and DARPA/ARO (P-3316-EL/DAAH04-94-G-2080 and DAAG55-97-1-0345).

### 2 EXPERIMENTS

We demonstrated earlier that BDD variable ordering heuristics can be sensitive to the order and labeling of the circuit netlist [6, 7]. In the experiments that follow, we use *netlist isomorphism classes* of combinatorial circuits to study the variability of performance of the ordering heuristics in VIS. The process of random re-ordering and re-labeling of circuit graphs to form isomorphism classes is described in [5].

The discussion of our experiments is organized as follows. First, we review the terminology and procedures used. We then discuss briefly the results of four particular experiments: (2) we examine the sensitivity of static ordering heuristics to variable labeling; (3) we examine whether dynamic reordering can find a known optimal ordering for some simple circuits. The final two experiments study the asymptotic performance of the ordering heuristics, (4) using scalable circuits and (5) using subfunctions of a well-known benchmark circuit. In all cases, we contrast the performance of VIS 1.3 with our previously published results for VIS 1.1.

**1. Experimental Procedure.** In [7] we defined a series of *experimental treatments* for the systematic evaluation of BDD variable ordering heuristics. Figure 1 summarizes the treatments that will be discussed in this paper. Each treatment of a combinatorial circuit consists of an initial (static) variable ordering, construction of a BDD for the function (possibly with dynamic reordering enabled), and zero or more dynamic reordering steps following BDD construction.

Treatment	Initial Ordering	Dynamic Ordering
0	Natural	None
1	VIS	None
2	VIS	Sift during construction
3	VIS	Sift during construction Sift once after construction
4	VIS	Sift during construction Sift twice after construction
5	VIS	Sift during construction Sift-converge after construction
6	Natural	Sift during construction
7	Natural	Sift during construction Sift once after construction
8	Natural	Sift during construction Sift twice after construction
9	Natural	Sift during construction Sift-converge after construction

Fig. 1. Experimental treatment definitions.

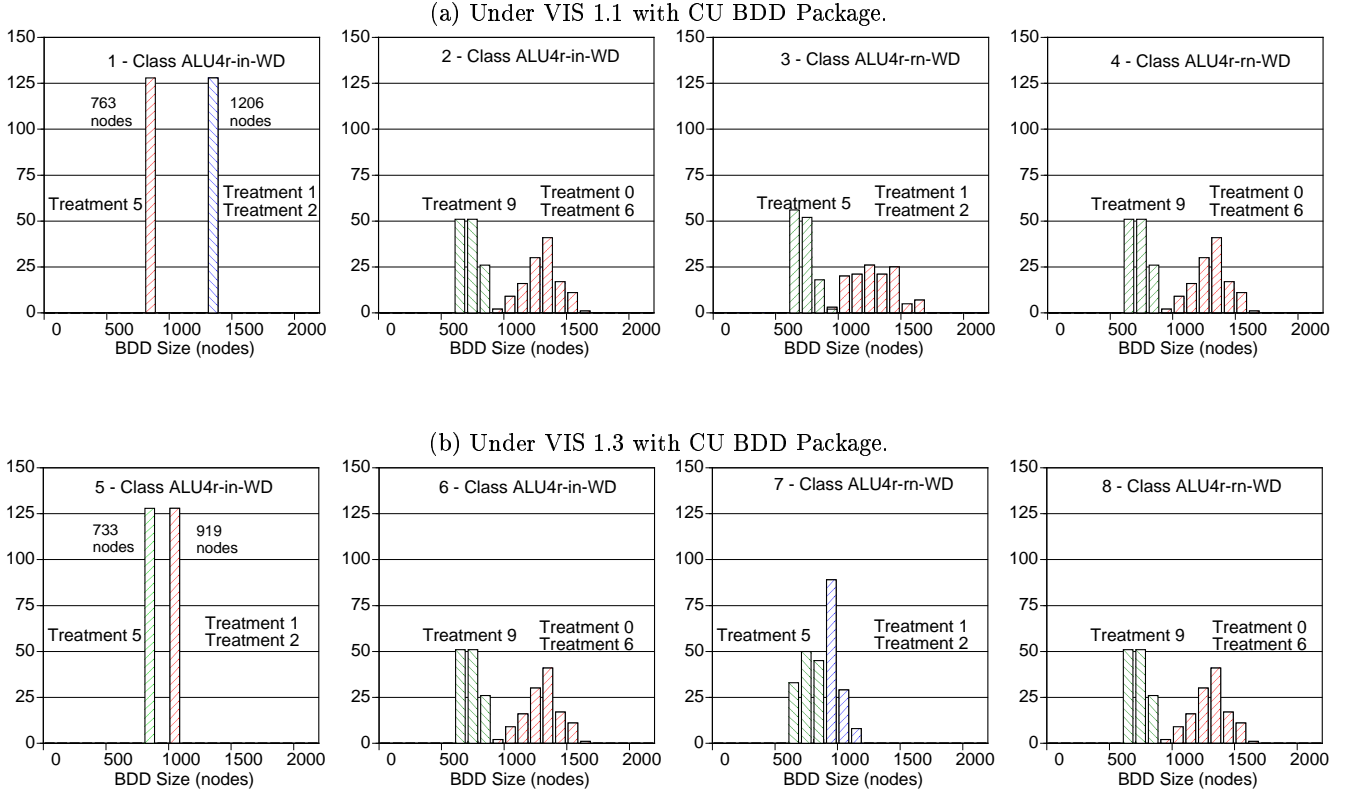


Fig. 2. BDD size (node) statistics under different variable ordering strategies, denoted as “treatments”.

In all cases, the `print_partition_stats` command in VIS is used as an evaluator, and the MDD (Multivalued Decision Diagram) node count is used as the cost function. We use the  $t$ -test to determine whether the means of two distributions are significantly different;  $|t| \geq 1.98$  indicates a significant difference of means at the 95% confidence level.

**2. Sensitivity to Variable Labeling.** We apply a set of treatments to the isomorphism classes `ALU4r-in-WD` (128 reordered instances of a 4-bit ALU circuit) and `ALU4r-rn-WD` (128 relabeled, reordered instances of the same circuit) to demonstrate the importance of randomizing both the order and labeling of netlists [7]. Figure 2a summarizes the results. In Figure 2a-1, all circuits achieve identical static orderings, and thus identical node counts, when the variables are not randomly relabeled (763 and 1206 nodes, respectively). In [6], this was determined to be a feature of *static ordering* in VIS 1.1: before applying the static variable ordering algorithm itself, an internal hash table ordering overrides the variable order provided in the input data file. Figure 2a-3 shows the results of identical treatments applied to the same circuit class, but with the variables not only reordered but also randomly relabeled. Now, the static ordering algorithm sees a different variable order since the random relabeling of nodes, for each circuit instance, has confounded and randomized the ordering induced by the hash table in VIS. As emphasized in [5], the randomization process of *re-order and re-label* is an essential part of a good experimental design. Without it, experiments run the risk of generating data that does not at all reflect the true behavior of the algorithm whose performance we are testing.

Figure 2b shows the results of the same procedures under VIS 1.3. Although identical results are obtained when random static orders are forced (Figure 2b-6 and 2b-8), there are obvious differences in performance when VIS static ordering is enabled (Figure 2b-5 and 2b-7).

If we were to compare the results shown in Figure 2a-1 for VIS 1.1 (723 and 1206 nodes) with Figure 2b-5 for VIS 1.3 (733 and 919 nodes) and declare an ‘improvement’, it would be premature, since the experiment has been confounded again by the hashing of input data into an order that is the same for all input representations. The only way to deal with such problems is to randomize node labels, even though all netlists remain isomorphic and logically identical.

Randomizing the node labels for each circuit instance and repeating the experiment again induces distributions, as shown in Figure 2b-7, which captures the correct behavior under VIS 1.3 of the algorithms labeled as Treatment 1, 2, and 5. Comparing the distributions for Treatment 1, 2, and 5 under VIS 1.1 and VIS 1.3, using the data of Figures 2a-3 and 2b-7, we can now correctly conclude the following: (1) VIS 1.3 is significantly better only for Treatment 1, 2 ( $t = 19.4$ ), while (2) VIS 1.1 is significantly better for Treatment 5 ( $t = 4.03$ ). The conclusions based on this experiment using the results in Figure 2b-5 would clearly be wrong!

**3. Effectiveness of Dynamic Reordering.** In [7] it was demonstrated that, even under the “strongest” available reordering treatment, the results were far from optimal. Here, we repeat the experiment under VIS 1.3, and similar results are obtained. Figure 3 shows the effects of successive dynamic reordering for a 15-bit carry circuit with a known

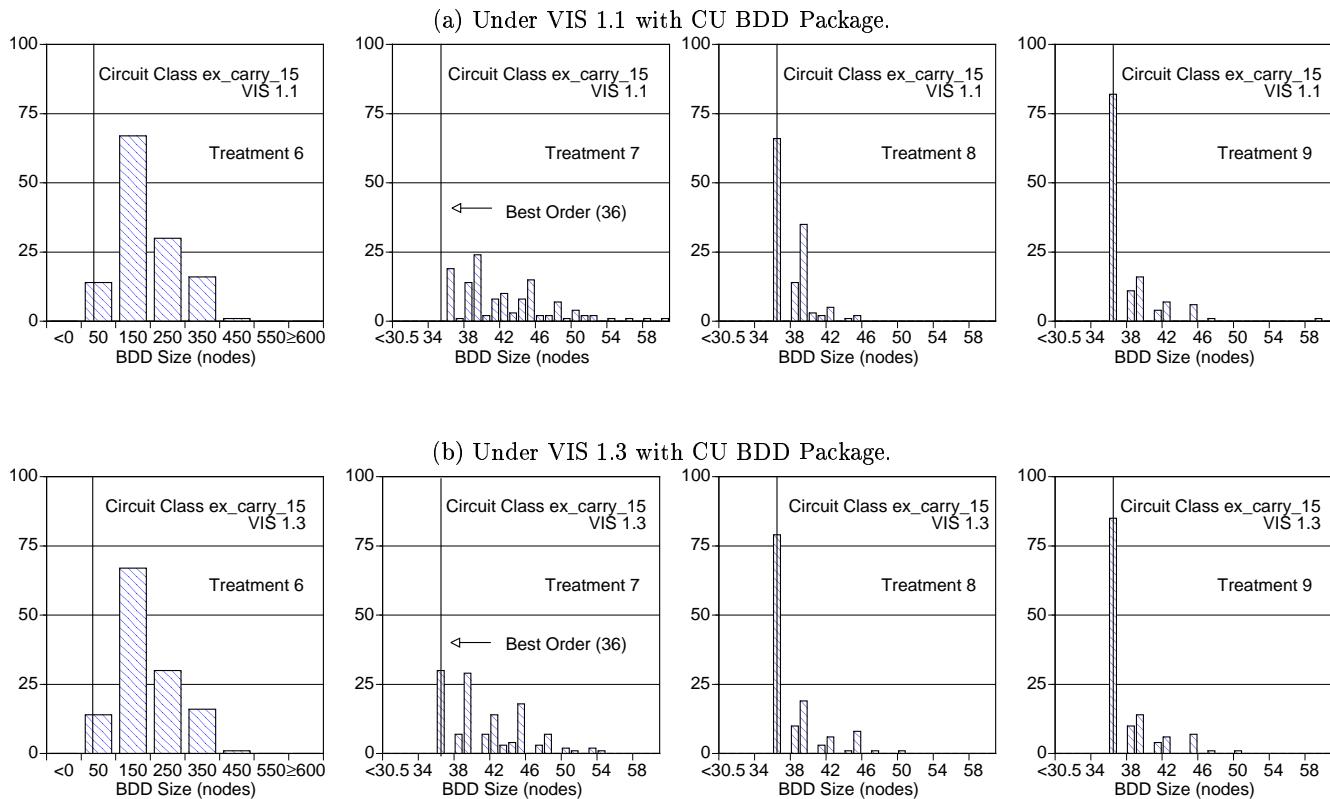


Fig. 3. BDD size distributions for isomorphic and logically equivalent classes of 128 15-input carry circuits.

optimal order. Inspection of the histograms for treatments 7 through 9 for VIS 1.3 suggests small improvements over VIS 1.1, but the  $t$ -test proves that these differences are not statistically significant. The  $t$  score is  $-0.54$  for treatment 8 and  $+0.288$  for treatment 9. However, observe that about 25% of all mutants in this class achieve non-optimal orderings, even with exhaustive sifting (treatment 9) under either release.

#### 4. Asymptotic Performance of Dynamic Reordering.

Figure 4 shows a test of asymptotic performance for the two releases of VIS, using classes of scalable circuits (*warp*, a chain of multiplexors, and *ex\_carry*, a modified carry chain) with known optimal orders. As expected, the BDD size increases exponentially with the number of circuit inputs; however, note that the distributions of BDD sizes (summarized as box plots) are almost entirely *above* the best-order line in all cases. These results are similar to those shown in Figure 2 for the ALU circuit. The  $t$ -tests for the 31 and 63 input *ex\_carry* circuits give scores  $-3.98$  and  $-6.78$ , indicating that VIS 1.1 achieved significantly better orders than VIS 1.3.

**5. Performance of Dynamic Reordering.** Figure 5 presents results for seven classes of circuits derived from the well-known C432 benchmark circuit [8]. The reference circuit has 7 outputs; we have constructed classes which represent subfunctions of C432 having 1, 2,  $\dots$  6 outputs. For each such subfunction, an isomorphism class was constructed and evaluated. Visual inspection indicates that the differences between VIS 1.1 and 1.3 are insignificant in all cases other than treatment 2 and 4. Under VIS 1.3, the

small distribution of BDD sizes under treatment 2 and 4 shows that the static ordering algorithm is now achieving good orders consistently, as demonstrated by the very tight distributions of BDD sizes.

### 3 CONCLUSIONS

The experiments described above lead to several observations:

1. VIS 1.3 static ordering is an improvement over VIS 1.1 static ordering. The latter was shown to be equivalent to random ordering in [6].
2. VIS 1.3 static ordering still relies on hashing the order of data read from the input file. Hence for the same netlist, the same variable order will be produced, regardless of the order of the input. The only way to invoke the complete behavior of the static ordering in VIS 1.3 is to randomly re-label all nodes, as was previously demonstrated for VIS 1.1 in [6].
3. VIS 1.3 dynamic reordering appears not to result in orders as good as VIS 1.1. This is particularly surprising, since static ordering is no longer random.
4. Neither release is able to consistently achieve optimal orders for small, regular circuits with obvious best ordering.

The conclusions of these experiments are not meant to disparage VIS, or any system, but to demonstrate how systematic experimental design can be used to reveal whether perceived differences between releases are significant.

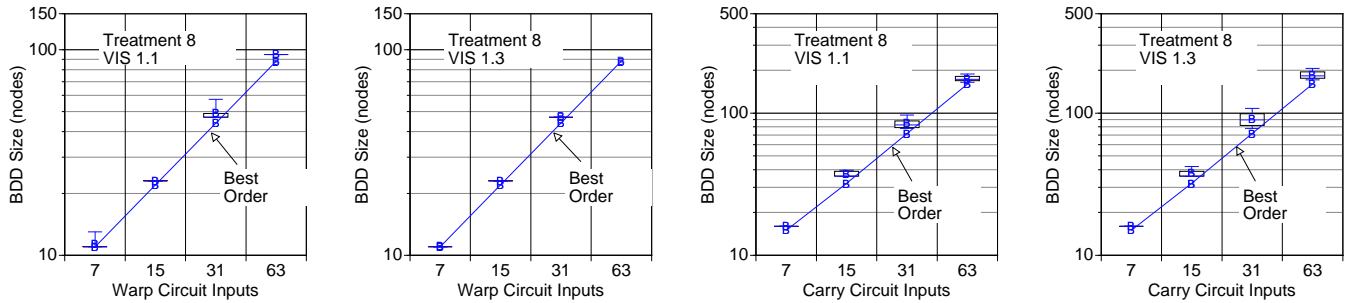


Fig. 4. Asymptotic behavior of two families of scalable circuit classes, each isomorphic and logically equivalent, with 128 circuits per class.

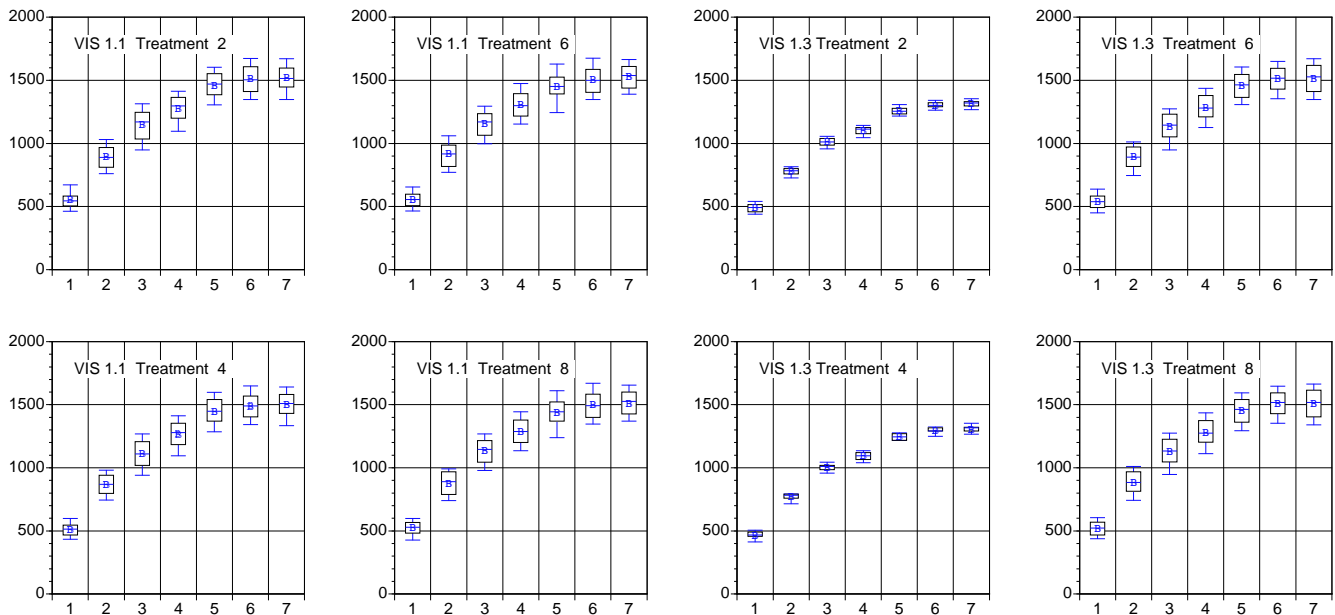


Fig. 5. BDD sizes for 7 classes of C432 with 1 to 7 outputs. Each class consists of 100 isomorphic and logically equivalent circuits.

**ACKNOWLEDGMENTS.** A number of the isomorphic and logically equivalent circuit classes used here were generated by Debabrata Ghosh. Nevin Kapur contributed a statistical summary generator for experimental results. Tom Shiple, Rajeev Ranjan, and Fabio Somenzi provided valuable assistance in understanding the workings of VIS.

#### REFERENCES

- [1] P. Ashar and M. Cheong. Efficient breadth-first manipulation of binary decision diagrams. In *Proceedings of the International Conference on Computer Aided Design*, pages 622–627, November 1994.
- [2] F. Somenzi et al. Colorado University Decision Diagram package (CUDD), release 2.3.0, 1997. Software available from <ftp://vlsi.colorado.edu/pub/cudd-2.3.0.tar.gz>.
- [3] D. E. Long. CMU BDD package, 1993. Software available from <http://emc.cs.cmu.edu/pub/bddlib.tar.Z>.
- [4] The VIS Group. VIS: A system for verification and synthesis. In R. Alur and T. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification*, number 1102 in Lecture Notes in Computer Science, pages 428–432, New Brunswick, NJ, July 1996. Springer. Version 1.3 is available from UC Berkeley Design Technology Warehouse at <http://www-cad.eecs.berkeley.edu>.
- [5] F. Brglez and R. Drechsler. Design of Experiments in CAD: Context and New Data Sets for ISCAS'99. In *IEEE 1999 International Symposium on Circuits and Systems - ISCAS'99*, May 1999. Also available from <http://www.cbl.ncsu.edu/publications/#1999-ISCAS-Brglez>.

To access all papers in this session, see <http://www.cbl.ncsu.edu/experiments/DoEArchives/I999-ISCAS-session>.

- [6] J. E. Harlow and F. Brglez. Design of Experiments for Evaluation of BDD Packages Using Controlled Circuit Mutations. In *Proceedings of the International Conference on Formal Methods in Computer-Aided Design (FMCAD'98)*. Springer Verlag, Lecture Notes in Computer Science, November 1998. Also available from <http://www.cbl.ncsu.edu/publications/#1998-FMCAD-Harlow>.
- [7] J. E. Harlow and F. Brglez. Design of Experiments in BDD Variable Ordering: Lessons Learned. In *Proceedings of the International Conference on Computer Aided Design*. ACM, November 1998. Also available from <http://www.cbl.ncsu.edu/publications/#1998-ICCAD-Harlow>.
- [8] F. Brglez and H. Fujiwara. Special Session on ATPG (Also introducing 'A Neutral Netlist of 10 Combinational Benchmark Circuits'). In *Int. Symp. On Circuits and Systems*, 1985. Now a benchmark directory IS-CAS85 at <http://www.cbl.ncsu.edu/benchmarks/Benchmarks-upto-1996.html>.