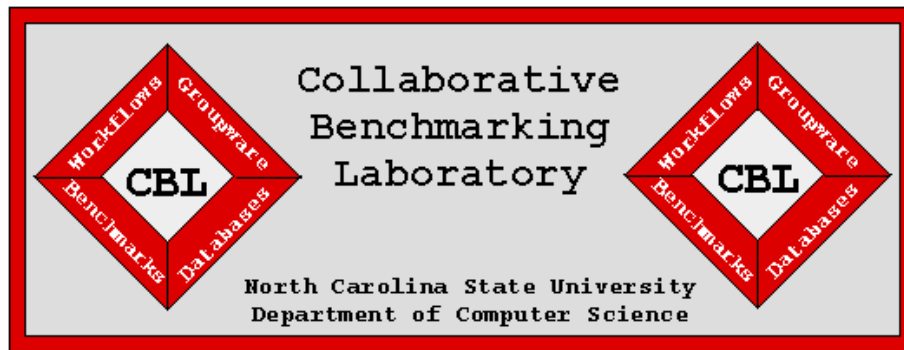


DRAFT



**Design of Experiments to Evaluate CAD Algorithms:  
Which Improvements Are Due to Improved Heuristic and  
Which Are Merely Due to Chance?**

Franc Brglez

April 1998

1998-TR@CBL-04-Brglez, Version 1.0

Reprinted, with permission, from  
<http://www.cbl.ncsu.edu/publications/>

Publications at this site are occasionally revised,  
please check for the latest version under the same title.

For more information about CBL, visit  
<http://www.cbl.ncsu.edu/>

or write to  
[info@cbl.ncsu.edu](mailto:info@cbl.ncsu.edu)

©1998 authors and CBL, All Rights Reserved

## ABOUT THIS DOCUMENT

**Acknowledgments.** The work presented in this document has been supported by 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), and a grant from Semiconductor Research Corporation.

**Abstract.** More than a thousand mathematical problems arising in engineering and science have been shown to be NP-hard. Problems of practical size that are NP-hard can only be solved by devising polynomial-time heuristics, with no guarantee whatsoever on the quality of the solution. Extensive experimentation and comparative analysis is required before we can decide on a 'better heuristic'. Past efforts to do either have been, for the most part, ad hoc. While there is no shortage of published claims of 'incremental improvements' with a particular heuristic, they are not supported by a test of hypothesis such as 'Is the improvement due to improved heuristic used by the algorithm or due merely to chance?'

This paper introduces motivation, context, and an approach to begin addressing and resolving such hypotheses using the design of experiment techniques rooted in the scientific method. We accept the methods of experimental design: *randomization*, *replication*, and *organization to reduce error*, first formalized by Fisher in the early 1920s, and adopted widely in many fields of science since. In the limited space available, we ask questions and propose solutions to the following, numbered in the order of sections in which we organized this paper:

(2) how to demonstrate the intrinsic variability in performance metrics that are optimized by heuristic algorithms solving NP-hard problems? Can we formulate and put to test the null hypothesis  $H_0$  for two algorithms: *There is no difference between the algorithms - i.e. any observed differences of 'optimized performance measures' are merely due to random fluctuations in sampling from the same population?*

(3) how to abstract the basic workflows in the design and execution of experiments?

(4) how to develop a number of well-defined equivalence classes of data as inputs for test of hypothesis with a variety of algorithms, ranging from physical design to logic synthesis and test?

(5) how to summarize, on a single page, results of a comprehensive 2-algorithm experiment, involving 8 data set sizes, 6 equivalence classes for each data set, and 100 instances of a problem in each equivalence class - a total of  $8*6*100 = 4800$  data files processed by each of the two algorithms under evaluation?

(6) how to relate this tutorial to the recent and on-going team effort on mutant classes, comprehensive experiments, and the infrastructure to support collaborative and peer-reviewed design, execution, and archival of CAD algorithm experiments on the World Wide Web?

**Keywords.** design of experiments, circuit mutants, equivalence class, benchmarking.

**Note.** This document has been published for viewing on the Web in Postscript and HTML formats. The latter is annotated with active hyperlinks to facilitate quick browsing of this and related documents.

**Citation.** If you choose to cite this report, please add the following entry to your bibliography database:

```
@techreport{
1998-TR@CBL-04-Brglez,
author = "F. Brglez",
title = "{Design of Experiments to Evaluate CAD Algorithms:
Which Improvements Are Due to
Improved Heuristic and Which Are Merely Due to Chance?}",
institution = "{CBL, CS Dept., NCSU, Box 7550,
Raleigh, NC 27695}",
number = "1998-TR@CBL-04-Brglez",
month = "April",
year = "1998",
note = "{Also available at
\tt http://www.cbl.ncsu.edu/publications/}"
}
```

CONTENTS

<b>I</b>	<b>Introduction</b>	<b>1</b>
<b>II</b>	<b>Motivation</b>	<b>1</b>
<b>III</b>	<b>Workflow Abstractions in DoE</b>	<b>3</b>
<b>IV</b>	<b>Classification of Circuit Mutants</b>	<b>4</b>
<b>V</b>	<b>Statistical Approach: An Example</b>	<b>6</b>
<b>VI</b>	<b>Collaborative Experiments</b>	<b>7</b>
<b>VII</b>	<b>Conclusions</b>	<b>7</b>

LIST OF FIGURES

1	Templates for tabulating old (a) and proposed (b-c) experiments with CAD algorithms. . . . .	2
2	Basic workflows in the design of experiments. . . . .	3
3	Wiring (a) and functional (b) perturbations inducing equivalence mutant classes and evaluation of algorithms. . . . .	5
4	Design of an experiment: comparing performance of two sorting algorithms. . . . .	6
5	Conceptual view of a testbed for the distributed design and distributed execution of experiments. . . . .	7
6	Web-based implementation of a specific placement evaluation workflow, reported by Place_Eval. . . . .	7

# Design of Experiments to Evaluate CAD Algorithms: Which Improvements Are Due to Improved Heuristic and Which Are Merely Due to Chance?

Franc Brglez

CBL (Collaborative Benchmarking Lab), Dept. of Comp. Science, Box 7550, NC State U., Raleigh, NC 27695  
<http://www.cbl.ncsu.edu/>

**Abstract** – *More than a thousand mathematical problems arising in engineering and science have been shown to be NP-hard. Problems of practical size that are NP-hard can only be solved by devising polynomial-time heuristics, with no guarantee whatsoever on the quality of the solution. Extensive experimentation and comparative analysis is required before we can decide on a ‘better heuristic’. Past efforts to do either have been, for the most part, ad hoc. While there is no shortage of published claims of ‘incremental improvements’ with a particular heuristic, they are not supported by a test of hypothesis such as ‘Is the improvement due to improved heuristic used by the algorithm or due merely to chance?’*

*This paper introduces motivation, context, and an approach to begin addressing and resolving such hypotheses using the design of experiment techniques rooted in the scientific method.*

**Keywords:** design of experiments, circuit mutants, equivalence class, benchmarking.

## I. INTRODUCTION

The design of experiments is regarded as an aspect of the scientific method: *the application of logic and objectivity to the understanding of the phenomena.* Its essential feature is the examination of what is already known and the formulation therefrom of the hypotheses which may be put to experimental test. Statistics provides a rationale for choosing criteria by which the possible differences in observations of two or more related experiments can be tested and interpreted.

We accept the methods and principles of experimental design: *randomization, replication, and organization to reduce error*, first formalized by R.A. Fisher in [1, 2], and adopted widely in many fields of science since. In the limited space available, we ask questions and propose solutions to the following, numbered in the order of sections in which we organized this paper:

(2) how to demonstrate the intrinsic variability in performance metrics that are optimized by heuristic algorithms solving NP-hard problems? Can we formulate and put to test the null hypothesis  $H_0$  [3, 4] for two algorithms: *There is no difference between the algorithms – i.e. any observed differences of ‘optimized performance measures’ are merely due to random fluctuations in sampling from the same population?*

(3) how to abstract the basic workflows in the design and execution of experiments?

This research has been supported by contracts from the Semiconductor Research Corporation (94-DJ-553), SEMATECH (94-DJ-800), DARPA/ARO (P-3316-EL/DAAH04-94-G-2080), and (DAAG55-97-1-0345), and a grant from Semiconductor Research Corporation.

“Permission to make digital/hard copy of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copying is by permission of CBL. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.”

© 1998 CBL

(4) how to develop a number of well-defined equivalence classes of data as inputs for test of hypothesis with a variety of algorithms, ranging from physical design to logic synthesis and test?

(5) how to summarize, on a single page, results of a comprehensive 2-algorithm experiment, involving 8 data set sizes, 6 equivalence classes for each data set, and 100 instances of a problem in each equivalence class – a total of  $8*6*100 = 4800$  data files processed by each of the two algorithms under evaluation?

(6) how to relate this tutorial to the on-going team effort in [5, 6, 7, 8, 9, 10, 11, 12, 13], on mutant classes, comprehensive experiments, and the infrastructure to support collaborative and peer-reviewed design, execution, and archival of CAD algorithm experiments on the World Wide Web?

## II. MOTIVATION

We analyze examples of experiments that are tabulated in “old-style” templates and contrast them with templates that conform to accepted norms in the design of experiments. The purpose of the symbolic “old-style” template in Table I is to draw attention to the fact that a report of an experiment on *single instances* of unrelated subjects would not be acceptable in publications related to bio-medicine<sup>1</sup>.

TABLE I  
A TEMPLATE NOT ACCEPTABLE IN BIO-MEDICINE.

Subject	Effect of Treatment1	Effect of Treatment2	“Improvement” %
one cat	a1	a2	$100(a2/a1 - 1)$
one cow	b1	b2	$100(b2/b1 - 1)$
one rat	c1	c2	$100(c2/c1 - 1)$
...	...	...	...
...	...	...	...
			...

One could raise at least two obvious objections to such comparison of two treatments: (1) a single subject is not representative of the population of its class, (2) subjects in this experiment are too diverse, genetically and in size, for meaningful comparisons. And yet, this template is precisely the one we use most often for the “comparative performance analysis” of two algorithms: just by replacing ‘subject’ with ‘circuits’ and ‘Treatment’ with ‘Algorithm’ we get the table as shown in Figure 1(a). This data is based on an actual publication, scaled to avoid identification. The 21 circuits are a subset from [14]. Most readers of CAD-related publications will readily recognize this template. The question arises: can we really ‘jump to conclusions’ that Algorithm2 is ‘better’

<sup>1</sup>JAMA, 1997;277:927-934. Under <http://www.ama-assn.org/public/journals/jama/sc6336.htm> the guidelines stipulate: *Describe statistical methods with enough detail to enable a knowledgeable reader with access to the original data to verify the reported results. When possible, quantify findings and present them with appropriate indicators of measurement error or uncertainty (such as confidence intervals). ... Discuss the eligibility of experimental subjects. Give details about randomization.*

(a) “Old-style” performance comparisons

Unrelated Circuits	Min. Cost via Alg1	Min. Cost via Alg2	Claimed Reduction
Circuit1	649.00	563.00	13.25%
Circuit2	120.00	86.00	28.33%
Circuit3	379.00	348.00	8.18%
Circuit4	213.00	207.00	2.82%
Circuit5	907.00	843.00	7.06%
Circuit6	277.00	276.00	0.36%
Circuit7	1775.00	1625.00	8.45%
Circuit8	276.00	256.00	7.25%
Circuit9	277.00	275.00	0.72%
Circuit10	1478.00	1359.00	8.05%
Circuit11	6150.00	5217.00	15.17%
Circuit12	745.00	675.00	9.40%
Circuit13	972.00	862.00	11.32%
Circuit14	223.00	162.00	27.35%
Circuit15	400.00	312.00	22.00%
Circuit16	118.00	72.00	38.98%
Circuit17	218.00	151.00	30.73%
Circuit18	112.00	90.00	19.64%
Circuit19	24.00	18.00	25.00%
Circuit20	67.00	52.00	22.39%
Circuit21	467.00	380.00	18.63%
		Average	15.48%

## Summary of observations:

- Data listed as column *Claimed Reduction* in template (a) has no statistical significance. Sample to sample variations of same magnitude are routinely observed in such experiments and also illustrated in (b-c) below (which also use 21 samples only).
- Mean/variance corresponding to minimized cost columns in (a) are 754.62/1354.7 and 658.52/1158.73 for Algorithm1 and Algorithm2, respectively. With data as shown, t-test, F-test, and U-test all confirm the null hypothesis that population means are equal, i.e. any improvement we may attribute to Algorithm2 is not significant and due to chance only.
- Data in the minimized cost columns in (b) raise the question whether the 21 samples are sufficient to reject the null hypothesis that that population means are equal. We find  $t = 2.36 > t_{crit=0.975} = 2.02$ , hence at the 5% level, the differences in the mean are significant and we reject the null hypothesis. With 100 samples, the difference in mean is significant at level much less than 1%.
- Data in the minimized cost columns in (c) raise the question whether the distributions reported by the same algorithm for (isomorphic) instances of the same circuit can indeed meet the null hypothesis. We find, for 21 samples, (circuit instances of 10 to 30 and 80 to 100 respectively)  $t = 0.576 > t_{crit=0.975} = 2.02$ , i.e. the difference in population means is not significant at the level of 5%. Same conclusions apply to larger sample sets.

(b) Data for proposed performance comparisons

Class of rd53	Min. Cost via SIS-alg	Min. Cost via SIS-bool
WD-0001	42	31
WD-0002	40	45
WD-0003	40	30
WD-0004	42	29
WD-0005	42	41
WD-0006	40	41
WD-0007	42	31
WD-0008	42	41
WD-0009	38	45
WD-0010	42	44
WD-0011	40	29
WD-0012	40	40
WD-0013	40	30
WD-0014	38	47
WD-0015	42	29
WD-0016	42	29
WD-0017	42	30
WD-0018	42	40
WD-0019	40	37
WD-0020	40	43
WD-0021	38	45
Mean	40.67	37.00
St. Dev.	1.47	6.81

(c) Data for proposed evaluation of equivalence classes

Class of V65E3-64	Min. Cost via DOT	Class of V65E3-64	Min. Cost via DOT
WD-0010	2	WD-0080	0
WD-0011	21	WD-0081	6
WD-0012	46	WD-0082	11
WD-0013	24	WD-0083	25
WD-0014	57	WD-0084	0
WD-0015	34	WD-0085	36
WD-0016	24	WD-0086	57
WD-0017	17	WD-0087	10
WD-0018	44	WD-0088	32
WD-0019	28	WD-0089	20
WD-0020	26	WD-0090	19
WD-0021	0	WD-0091	57
WD-0022	52	WD-0092	0
WD-0023	10	WD-0093	23
WD-0024	17	WD-0094	36
WD-0025	34	WD-0095	13
WD-0026	57	WD-0096	34
WD-0027	50	WD-0097	22
WD-0028	24	WD-0098	32
WD-0029	5	WD-0099	49
WD-0030	0	WD-0100	24
Mean	27.24		24.10
St. Dev.	18.00		16.69

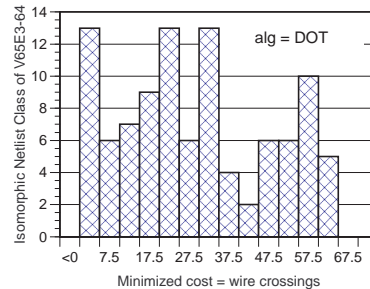
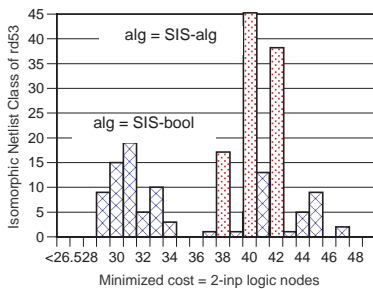


Fig. 1. Templates for tabulating old (a) and proposed (b-c) experiments with CAD algorithms.

just because the ‘claimed reduction’ column seems to suggest this?

Data presented in Figure 1(b) is part of a larger set of experiments in *logic synthesis* available from the Web [8]. Specifically, we list optimized number of *2-input nodes* as reported by SIS-alg and SIS-bool [?] for a set of 21 *isomorphic* instances of a small circuit rd53 – generated as Class.WD of circuit mutants with 0 perturbations, by merely randomly reordering

the nodes in the netlist (as a special case of perturbation-based classes of mutants, also posted on the Web). Clearly, both algorithms are quite *sensitive to the order of nodes in the netlist*, shown in more detail in the companion histogram for 100 circuits in this class. The question arises: are the population means of these distributions the same?

Data presented in Figure 1(c) is also a part of a larger set of experiments in *physical design* available from the Web

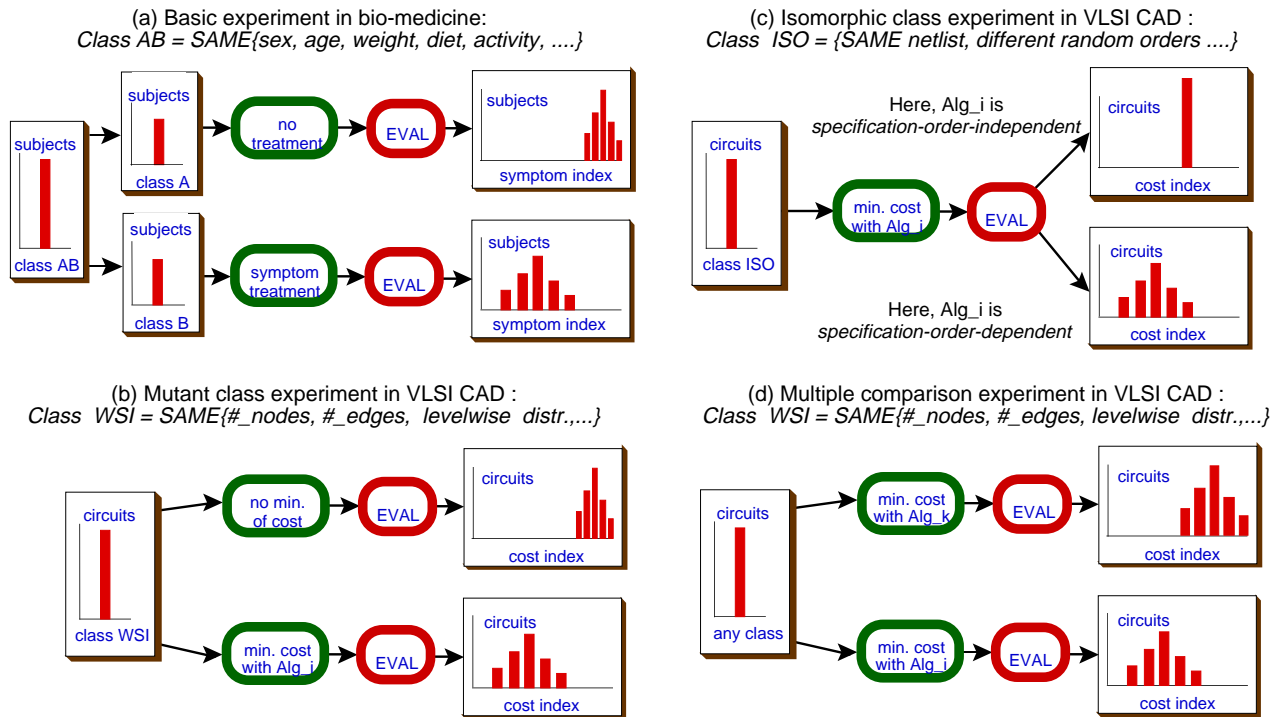


Fig. 2. Basic workflows in the design of experiments.

[8]. Specifically, we list optimized number of *wire crossings* as reported by a state-of-the-art program DOT [15] for *two sets* of 21 *isomorphic* instances of a small *planar* two-layer graph V65E3-64. Again – these graphs are in Class\_WD of circuit mutants with 0 perturbations. Given that this is a planar graph, one may be surprised that the algorithm reports only 11 instances out of 100 where wire crossing has been reduced to 0! The companion histogram shows that for some initial orderings of the netlist, wire crossing cannot be reduced below 65 with this algorithm. The question arises: will the hypothesis test confirm that the distributions generated by the *same algorithm* for (isomorphic) instances of the *same circuit* belong to the *same population*?

Correct answers to these questions are not obvious without applying some statistical tests such as *t*-test, *F*-test, *U*-test, and others, now accessible in textbooks, e.g. [?], and in a number of statistical software packages. Some of the answers are summarized in the box of Figure 1. Overall, the limited set of results discussed in this section brings out the following:

- The “old-style” of reporting results such as shown in Figure 1(a) does not provide enough support to claims of improvements at a recognized and acceptable level of statistical significance.
- There can be a nontrivial level of pairwise differences between two algorithms on any given sample. More significant tests, such as discussed earlier, and the confidence intervals for the mean and differences of means, must be considered before we can draw conclusions about the merits of each algorithm.
- There can be nontrivial levels of pairwise differences between two instances of a circuit in the same equivalence class, even when evaluated by the same algorithm. To establish class equivalence, more significant tests, such as discussed earlier, and the confidence intervals for the mean and differences of means, must be considered before we can draw conclusions about the unique properties of each class.

### III. WORKFLOW ABSTRACTIONS IN DoE

We discuss four abstractions in the design of experiments (DoE): a simplified flow from bio-medicine, and three flows related to experiments in VLSI CAD. At the level of abstraction considered in this paper, the flows shown in Figure 2 are quite similar. The input data to each flow is an *equivalence class*, depicted as a *single bar distribution*: e.g. a class of closely related subjects for experiments in bio-medicine, or closely related circuits for experiments in CAD algorithms. In the bio-medical experiment, the treatment may consist simply of prescribing a drug to lower blood pressure. In the CAD experiment, we consider an algorithm that minimizes some cost functional as analogous to a treatment in bio-medicine. Henceforth, we will use the term *treatment* interchangeably in both contexts since the same or similar statistical methods will apply to evaluation of distributions that arise *after treatment*, regardless of the class and target application.

(a) **A basic experiment in bio-medicine.** The distinguishing characteristics of this experiment are (1) the effort to produce a sufficiently ‘similar’ and sufficiently ‘large’ subject class *AB* before splitting them into equal size classes *A* and *B*, and (2) the effort to maintain a consistent environment for subjects in both classes during the entire duration of the experiment. Ideally, at the beginning of the experiment, subjects in either class will maintain properties of class *AB*, such as

$$\text{Class } AB = \text{SAME}\{\text{sex, age, weight, diet, activity, blood pressure, ...}\} \quad (1)$$

Treatment may consist of administering, at regular intervals and for a significant period of time, a placebo to group *A*, and a blood pressure reducing drug to group *B*. Upon completion of treatment, subjects are measured for blood pressure levels and any side-effects. Statistical methods are applied to analyze new data gathered from both groups and validate tests of hypotheses, e.g. is the blood pressure reducing drug effective

at a level that is statistically significant as well as safe.

**(b) Circuit mutant class experiments.** These experiments mirror the concepts introduced for the basic experiment in bio-medicine as discussed above. The only difference is that subject are now circuits and the treatment implies application of an algorithm to the circuits. While we formalize the notion of circuit mutant classes in the next section, we draw attention to analogies in the definition of subject classes such as introduced in (1) and the circuit mutant classes introduced in (2), (3). The class *WSI*, wiring signature-invariant class, maintains invariant most parameters that relate to *circuit structure*. Any parameter variations are within well-defined bounds.

$$\text{Class } WSI = \text{SAME}\{|\text{nodes}|, |\text{edges}|, \text{levelwise} \\ \text{and typed net distributions, ...}\} \quad (2)$$

The class *FPI*, functional perturbation-invariant class, maintains invariant most parameters that relate to *circuit function*. Any parameter variations are within well-defined bounds.

$$\text{Class } FPI = \text{SAME}\{|\text{nodes}|, |\text{edges}|, \text{functional} \\ \text{perturbation distributions, ...}\} \quad (3)$$

In the context of Figure 2(b), ‘no minimization of cost’ is equivalent to the ‘placebo’ or ‘no treatment’ described earlier. For example, we may compare quality indices of placements of a netlist in ‘natural order’ versus placements optimized by an algorithm. In both cases, we will get a *distribution of placement indices*. Again, statistical methods and tests of hypotheses can be applied to quantify the effectiveness of the particular algorithm. However, before proceeding to discuss comparisons of *two* algorithms, we need to introduce and analyze experiments with the isomorphic class.

**(c) Isomorphic class experiments.** There is no exact analogue for isomorphic classes in bio-medicine. The closest one can get is to select class *AB* in (1) under additional constraints, e.g. identical twins if subjects are humans; genetically engineered clones if subjects are animals or plants. The class *ISO*, isomorphic class, is simply a class of identical netlists, just rendered in different random order.

$$\text{Class } ISO = \{\text{SAME netlists, in different} \\ \text{random orders, ...}\} \quad (4)$$

For a given algorithm, experiments with isomorphic classes of circuits have one of the two characteristic outcomes: (a) there is *no dispersion* in the minimized cost index for any order of the netlist; and (b) there is *dispersion* in the minimized cost index for at least one order of the netlist. We say that the algorithm is *specification-order-dependent* upon observing the outcome (a), and *specification-order-independent* upon observing the outcome (b).

The example in Figure 1(b) illustrates instances of two state-of-the-art algorithms in logic minimization that are *specification-order-dependent*, even for isomorphic netlists with less than 100 nodes. Figure 1(c) illustrates instances of a state-of-the-art algorithm in wire crossing minimization that is *specification-order-dependent* for similarly-sized netlists. Maximum variations observed in Figure 1(b) are  $(45/29 - 1) = 55.2\%$ , and  $(57/2 - 1) = 2950\%$  in Figure 1(c). As shown in the companion histograms, these variations increase further as we increase the size of the isomorphic class from 21 to 100. On the other hand, the maximum variation claimed as ‘reduction’ of Algorithm2 vs Algorithm1 in the ‘old-style’ experiment in Figure 1(a) is 38.98%, again raising the question whether any reported ‘improvements’ are due to

chance alone. Additional results, with larger isomorphic class circuits, for algorithms ranging from logic synthesis to placed and routed layouts, are reported and archived for ready access on the Web [8].

The first known recorded instance of using relatively small isomorphic netlist classes to test performance of placement algorithms was reported more than a decade ago [16]. Apparently, merits of such experiments were not put to another test until generalized to perturbation-based mutant classes in [5, 17, 10].

**(d) Multiple comparison experiments.** As shown in Figure 2(d), these experiments execute any of the three circuit classes, *WSI*, *FPI*, *ISO*, with two or more algorithms. Often, *WSI* and *ISO* class may be used to test the performance of algorithms in physical design as well as logic synthesis. Typically, classes *FPI* and *ISO* may be used to test the performance of algorithms in logic synthesis and verification. Rigorous multi-comparison statistical methods can be applied to the analysis of distributions resulting from such experiments [4].

#### IV. CLASSIFICATION OF CIRCUIT MUTANTS

The instances of isomorphic class circuits *ISO* in (4) and Figure 1(b-c), bring out important properties of typical CAD algorithms: most may induce a class distribution with significant variance rather than a variance of 0 (expected by an ideal algorithm). The question arises: can we generate equivalence classes in some well-defined ‘neighborhoods’ of the isomorphic class, such that we can study the *sensitivity* of each algorithm to *differences* between the ‘neighborhood class’ and the isomorphic class.

We have generated the first generation of classes in the ‘neighborhoods’ of the isomorphic class and called them *circuit mutants* [17, 9, 10]. The mechanism of choice to generate a mutant class is a perturbation technique. First, we select any circuit from the isomorphic class and call it a *reference circuit*. Second, we characterize its major invariant parameters to create a *class signature*. Third, we subject each copy of a reference circuit to a set of random but controlled-size perturbations, giving rise to a mutant class of circuits for each perturbed copy of the reference circuit.

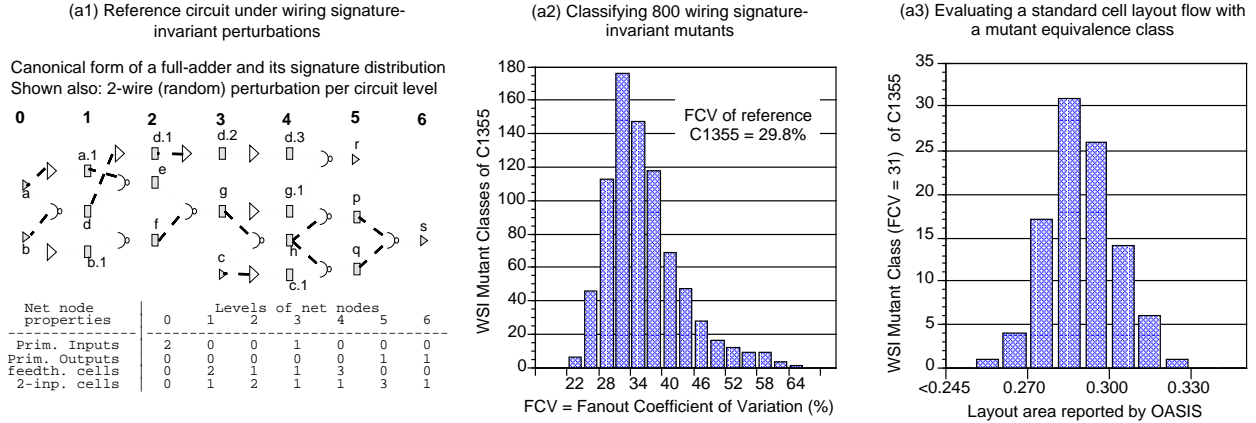
Specifically, we introduce *wiring perturbations* and *functional perturbations*, each giving rise to distinct classes of circuit mutants. The class *WSI*, wiring signature-invariant class introduced in (2), maintains invariant most parameters that relate to *circuit structure*. The class *FPI*, functional perturbation-invariant class introduced in (3), maintains invariant most parameters that relate to *circuit function*. We briefly illustrate the fundamental principles for both classes.

##### Wiring Perturbation Classification and Analysis.

Consider the simple example of the reference circuit in its canonical form in Figure 3(a1) [17]. The canonical form gives rise to a *conditional incidence relation*, a *basic signature*, and an *extended signature*. The *basic signature* for this example can be verified by inspection. There are 3 primary input nets, 2 at level 0, 1 at level 3. There are 7 nets driven by feedthroughs: 2 at level 1, 1 at level 2, 1 at level 3, etc. There are 9 nets driven by 2-input cells: 1 at level 1, 2 at level 2, 1 at level 3, etc. The *extended signature* is in the form of fanout bounds that must be *always* satisfied during the mutant synthesis process. In the example shown, the circuit is subject to 2-wire perturbations at each level. During the perturbation phase, two wires are selected randomly and removed at each level. During the mutant synthesis phase, two wires are re-introduced at each level and connected randomly, *subject to wiring signature invariance constraints*.

The example in Figure 3(a2) is a histogram of 800 *WSI*

(a) Wiring perturbations inducing equivalence mutant classes and evaluation of algorithms



(b) Functional perturbations inducing equivalence mutant classes and evaluation of algorithms

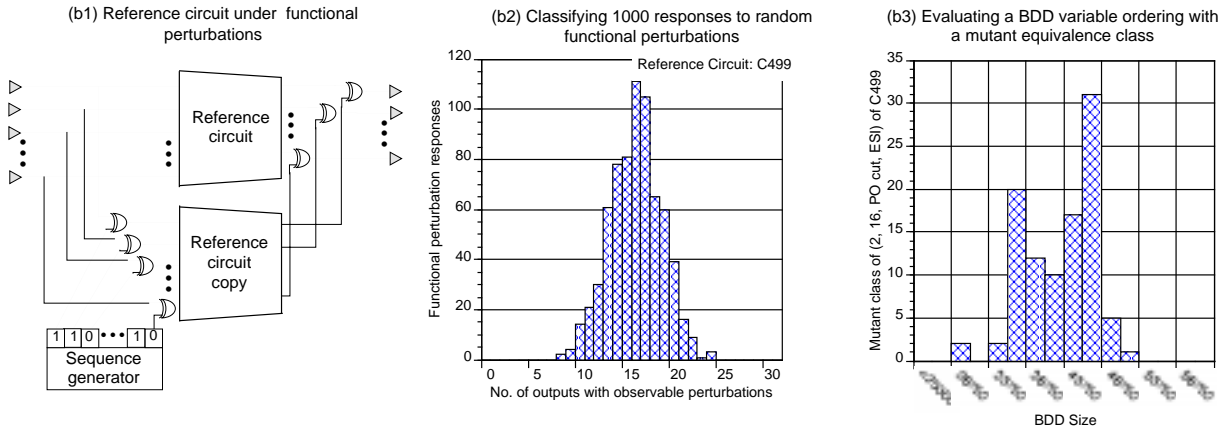


Fig. 3. Wiring (a) and functional (b) perturbations inducing equivalence mutant classes and evaluation of algorithms.

mutants of circuit C1355 [14], classified in terms of the *fanout coefficient of variation FCV*, an average across all levels of standard coefficient of fanout variation [9]. Note that the distribution peaks for a value of  $FCV = 31.0\%$ , while for the reference circuit (and isomorphic class of C1355) we have  $FCV = 29.8\%$ . We propose to use the classification in terms of  $FCV$  to extend the range of isomorphic classes. Applying 100 mutants from the C1355 WSI subclass with  $FCV = 31\%$  to a layout tool [18] gives rise to a near-normal distribution in Figure 3(a3) – similar to the distribution obtained for the isomorphic class, but with a larger reported mean and a larger variance.

### Functional Perturbation Classification and Analysis.

Consider the generic example of the reference circuit connected to its own copy as shown in Figure 3(b1) [10]. *Each input* of the reference circuit copy is perturbed with outputs from the sequence generator which has the property that all possible sequences can be generated on its outputs except the sequence  $\{0000 \dots 00\}$ . This condition ensures that at least one input of the reference circuit copy is always perturbed relative to inputs that are applied to the reference circuit. *All* outputs of both circuits are XOR-ed and are of interest in the proposed simulation experiment.

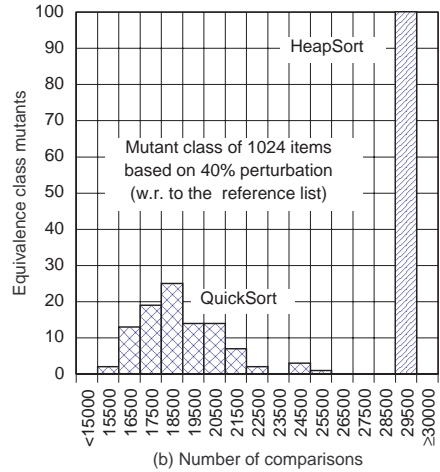
The experiment involves (1) application of random patterns to circuit primary inputs, (2) application of random sequences to perturb the inputs of the reference circuit copy, (3) classifying the XOR-ed outputs of both circuits by the number of outputs with observable perturbations. Results of such an

experiment are shown for circuit C499 [14] in Figure 3(b2); notably there are over 160 circuits where perturbations are observable on exactly 17 outputs (from a total of 41 outputs). The near-normal distribution that peaks at 17 outputs becomes apparent already after the simulation of 100 patterns and its shape changes little from the one shown for 1000 patterns.

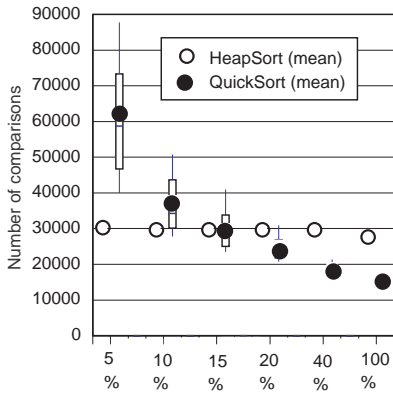
Data and classification based on the simulation of the reference circuit give rise to synthesis of the *FPI* (functional perturbation-invariant) mutant classes. For example, we can readily identify input patterns that perturb exactly 17 outputs in the context of Figure 3(b1). Subsequently, we can form classes with 1-minterm, 2-minterm, 3-minterm,  $\dots$ , perturbations that will perturb exactly 17 outputs. We must merely remove the sequence generator and replace the reference circuit copy with minterms that *always* fan out to different but exactly 17 XORs marked during the simulation. Further, we can refine such classes to be also *entropy-invariant*; and by monitoring interior cuts in the circuit, we can also introduce *function-invariant* circuit mutants [10]. The histogram in Figure 3(b3), is based on 100 mutants, from the C499 subclass of 2-minterm perturbation onto 17 outputs while maintaining entropy-invariance. BDD variable ordering algorithm [19] has found mostly different orders (and hence BDD sizes) for each mutant. The spread in BDD sizes is considerably out-of-proportion to the applied functional perturbations. A known good order, applied to the same mutant class, induces a distribution whose variance is negligible compared to the variance shown in Figure 3(b3) [10].

Number of comparisons required to sort N = 1024 items (in several mutant equivalence classes)									
Mutant	5% class		10% class		15% class		... 100% class		
	HS	QS	HS	QS	HS	QS	HS	QS	
1	30266	32998	30194	31130	30053	43708	...	28343	16591
2	30308	58905	30074	33000	30158	36918	...	28262	16839
3	30329	134017	30167	39974	30122	25672	...	28328	16021
4	30329	38812	29921	56403	29891	22132	...	28364	16154
5	30278	88484	30182	29019	30110	23334	...	28244	15228
6	30236	76125	29930	36908	30170	24913	...	28172	16548
7	30299	48076	30194	46443	30158	40072	...	28424	15487
8	30326	66827	30272	31091	30089	29174	...	28244	15674
9	30257	85059	30221	43510	30188	26300	...	28274	17247
10	30287	44791	30152	51236	30158	27408	...	28361	17097
...	...	...	...	...	...	...	...	...	...
100	30272	48445	30134	33554	30146	35943	...	28424	16736

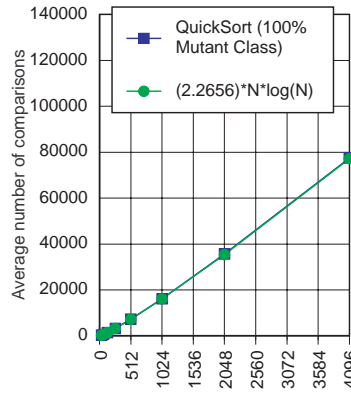
(a) In each class, a total of 100 files, N=1024 items in each file are sorted



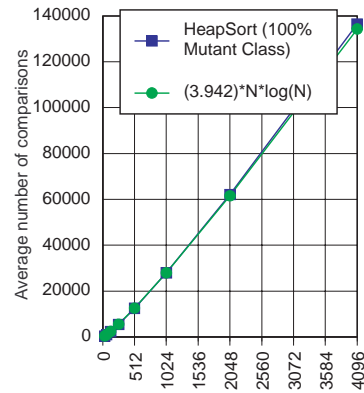
(b) Number of comparisons



(c) Mutant equivalence classes (N = 1024)



(d) Mutant class size (N)



(e) Mutant class size (N)

Fig. 4. Design of an experiment: comparing performance of two sorting algorithms.

## V. STATISTICAL APPROACH: AN EXAMPLE

We select two algorithms whose average-case behavior can, to a large extent, be analyzed explicitly: HeapSort (HS) and QuickSort (QS). Both have known average-case behavior: to sort a record of  $N$  items, the average number of comparisons is  $c_{HS} * N * \log(N)$  and  $c_{QS} * N * \log(N)$ , respectively. The constants  $c_{HS}$  and  $c_{QS}$  depend on the ordering profile of data: the worst-case behavior for QuickSort is  $c_{QS} * N^2$  in the special case where data is already sorted. We have designed the experiment in three phases.

### (1) Benchmark data synthesis:

- create a *reference data set* of 8 data files, each containing 32-, 64-, 128-, 256-, 512-, 1024-, 2048-, 4096-unique items in a *sorted order*.
- create *eight mutant data directories*, each consisting of *mutant class subdirectories* of 5%-, 10%-, 15%-, 20%-, 40%-, 100%-mutant classes, with 100 data files in each class. Each file in the  $q$ -mutant class is derived from the parent reference data file by *randomly permuting* positions of  $q$ -% items in the file.

### (2) Algorithm evaluation:

- invoke HeapSort to sort items in each of the  $8 * 6 * 100 = 4800$  benchmark data files and tabulate the number of comparisons taken to complete each of 4800 sorts.
- invoke QuickSort to sort items in each of the  $8 * 6 * 100 = 4800$  benchmark data files and tabulate the number of comparisons taken to complete each of 4800 sorts.

### (3) Statistical evaluation:

- organize tables of results from both algorithms to support various standardized forms of sample testing.
- make a determination, within specified limits of confidence, about the comparative performance of each algorithm.

### Summary of this experiment in Figure 4:

- *Figure 4(a)* shows actual samples of data tabulated for some of the perturbation classes. Note the relative ‘stability’ of comparisons for the HeapSort (HS).
- *Figure 4(b)* shows histograms for QuickSort (QS) and HeapSort (HS) for the mutant class of  $N = 1024$  items, based on the 40%-perturbation subclass.
- *Figure 4(c)* shows the ‘box graphs’ for QuickSort (QS) and HeapSort (HS) of all 6 perturbation subclasses for the mutant class of  $N = 1024$ . Note that at the 15%-perturbation subclass, there is a ‘crossover’ point for the two algorithms. It is also clear that for 40%-100% perturbation subclasses, QuickSort (QS) performs measurably better.
- *Figure 4(d-e)* summarizes the performance of both algorithms for the 100%-perturbation subclass and a range of values for  $N$ . Note that one of the major objectives of this comparison has been achieved, since we have determined the ‘constants’ for both algorithms:  $c_{HS} = 3.942$  and  $c_{QS} = 2.2656$ .

Since both algorithms are relatively stable, the precision of statistical estimates in this example is much higher than nominally required – i.e. we could do with less than 100 samples and fewer classes to arrive at comparable conclusions. Complete data sets related to this experiment are posted on the Web [8].

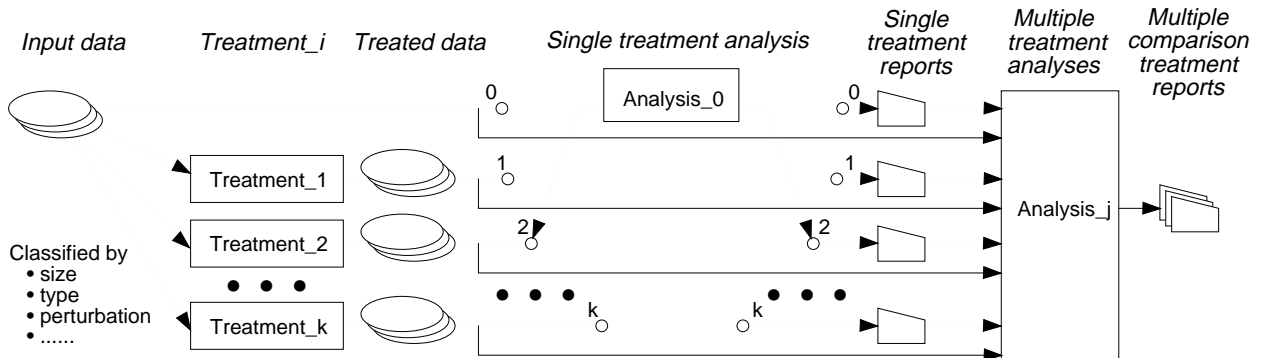


Fig. 5. Conceptual view of a testbed for the distributed design and distributed execution of experiments.

## VI. COLLABORATIVE EXPERIMENTS

We are working towards a web-based testbed environment to facilitate design of experiments for multi-disciplinary teams of distributed participants. The testbed will support design, execution, and performance evaluation of algorithms that address NP-hard problems in the context of specific applications in CAD of VLSI systems. In addition, the testbed will support clearance, classification, synthesis, and archival of test data serving as inputs to various algorithms, and standardized report generation based on archived results of experiments.

### Near-term Objectives.

1. Generate a number of equivalence class mutant circuits, conduct a series of comprehensive *illustrative experiments*, archive tables of results and summaries, and post everything on the Web for easy access and independent comparative analysis. The first pass of the initial experiments, and data which can be observed on the Web pages is detailed in [8].
2. Post guidelines and calls for participation in collaborative, peer-reviewed experiments and engage a core of leaders to contribute to the initial experiments.
3. Plan a series of meetings with participants and peers during forthcoming meetings, such as ICCAD'98.
4. Convene a workshop to consider formalization of methods in design of experiments and data classification, issuing a set of guidelines on reporting results of experiments, similar to those adopted by over 500 journals in biomedicine, in existence since 1978.

**Project Status.** Figure 5 depicts a high-level view of the proposed web-based and web-executable testbed environment. As explained earlier, having drawn analogies to traditional fields in the design of experiments, the terminology such as *Treatment<sub>k</sub>* is interchangeable with *Algorithm<sub>k</sub>*. Significant components of the testbed environments are:

- problem-specific but universally accessible archives of **Input data** (serving also as a 'placebo' for the experiments in the same treatment class);
- problem-specific and user-configurable encapsulation of a program that implements a specific algorithm in place of *Treatment<sub>k</sub>*;
- problem-specific but universally accessible archives of **Treated data**;
- problem-specific parameterized but *shared analysis program* *Analysis<sub>0</sub>*, generating impartial **Single treatment reports**, including tests of significance and hypothesis;
- problem-specific parameterized but *shared analysis programs* *Analysis<sub>j</sub>*, generating impartial **Multiple comparison treatment reports**, including tests of significance and hypothesis.

An example of the Internet-based executable workflow environment that implements many features represented in Figure

5 is shown in Figure 6. Notably, we use a *single evaluation*

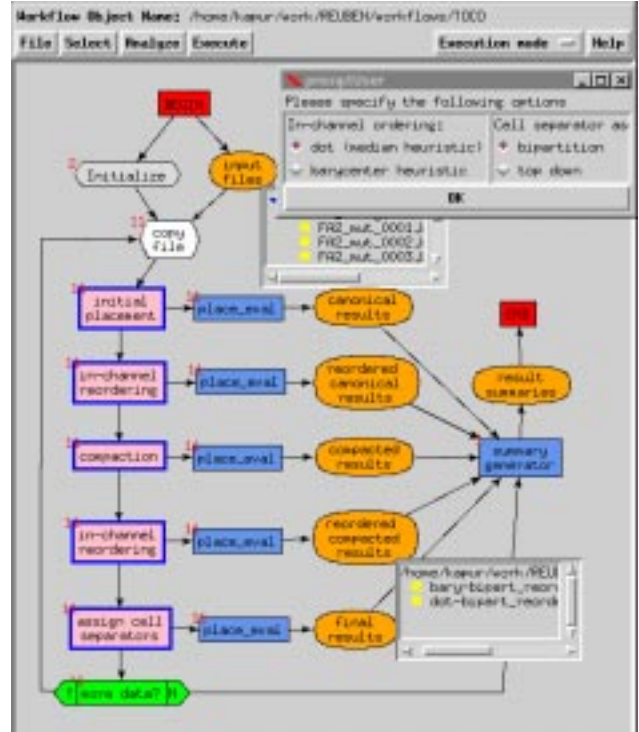


Fig. 6. Web-based implementation of a specific placement evaluation workflow, reported by Place\_Eval.

*program* Place\_Eval to evaluate results reported during intermediate and final phases of a specific placement algorithm: these include total and critical wire crossing, total and critical wire length, average wire density, layout width, height, area, etc. This is just one of the workflows we plan to demonstrate on-line in the University University Booth during the 1998 Design Automation Conference, June 15–17 in San Francisco.

## VII. CONCLUSIONS

The brief analyses of data generated by various algorithms support the notion that extensive experimentation and rigorous statistical analysis are required to test a hypothesis such as

*‘Is the improvement due to the choice of the algorithm or due to chance?’*

This is particularly true for problems of practical size that are NP-hard. We conclude that in order to measure performance

of graph-based algorithms, the research community in CAD for VLSI and computer science should fundamentally change the way it *designs and reports* on results of such experiments.

ACKNOWLEDGMENTS. Many ideas described in this paper are an integral part of a larger team project and are described in more detail in [5, 6, 7, 8, 9, 10, 11, 12, 13]. This paper is a tutorial introduction to the overall scope of this effort.

## REFERENCES

- [1] R. A. Fisher. *Statistical Methods, Experimental Design, and Scientific Inference*. Oxford University Press, 1993. Reprinted, with corrections, from earlier versions, 1925-1973.
- [2] R. A. Fisher. *Statistical Methods and Scientific Inference*. Hafner Press, 1973. Reprinted, with corrections, from earlier versions, 1956-1959.
- [3] K. A. Brownlee. *Statistical Theory and Methodology In Science and Engineering*. Krieger Publishing, 1984. Reprinted, with revisions, from second edition, 1965.
- [4] J. C. Hsu. *Multiple Comparisons: Theory and Methods*. Chapman & Hall, 1996.
- [5] N. Kapur, D. Ghosh, and F. Brglez. Towards A New Benchmarking Paradigm in EDA: Analysis of Equivalence Class Mutant Circuit Distributions. In *ACM International Symposium on Physical Design*, April 1997.
- [6] N. Kapur and F. Brglez. Benchmarking Cell Placements Using A Bipartite Graph Model. Technical Report 1997-TR@CBL-08-Kapur, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, Oct 1997. Also available at <http://www.cbl.ncsu.edu/publications>.
- [7] D. Ghosh, N. Kapur, J. Harlow, and F. Brglez. Synthesis of Wiring Signature-Invariant Equivalence Class Circuit Mutants and Applications to Benchmarking. In *Design Automation & Test - Europe (DATE'98)*, February 1998. Also available at <http://www.cbl.ncsu.edu/publications/>.
- [8] F. Brglez (Editor). A Brief Tour From The Home Page on WWW Statistical Experiment Archives: Benchmark Descriptions and Posted Solutions to NP-hard Problems. Technical Report 1998-TR@CBL-01-Brglez, Version 1.0, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, February 1998. Also available at <http://www.cbl.ncsu.edu/publications>.
- [9] D. Ghosh. Synthesis of Wiring Signature-Invariant Equivalence Class Sequential Circuit Mutants and Applications to Benchmarking. Technical Report 1998-TR@CBL-05-Ghosh, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, April 1998. Also available at <http://www.cbl.ncsu.edu/publications>.
- [10] J. E. Harlow III and F. Brglez. Characterization of Functional Perturbation-Invariant Mutant Equivalence Classes and Application to Design of Experiments in Verification and Logic Synthesis. Technical Report 1998-TR@CBL-06-Harlow, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, April 1998. Also available at <http://www.cbl.ncsu.edu/publications/>.
- [11] H. Lavana, A. Khetawat, F. Brglez, and K. Kozminski. Executable Workflows: A Paradigm for Collaborative Design on the Internet. In *Proceedings of the 34th Design Automation Conference*, pages 553-558, June 1997. Also available at <http://www.cbl.ncsu.edu/publications/>.
- [12] H. Lavana and F. Brglez. WebWiseTclTk: A Safe-Tcl/Tk-based Toolkit Enhanced for the World Wide Web. Technical Report 1998-TR@CBL-02-Lavana, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, Apr 1998.
- [13] H. Lavana and F. Brglez. WebWiseTclTk, OmniDesk and OmniFlows: A User-Configurable Distributed Design Environment inside a Web-Browser. Technical Report 1998-TR@CBL-03-Lavana, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, Apr 1998. Also available at <http://www.cbl.ncsu.edu/publications/>.
- [14] S. Yang. Logic Synthesis and Optimization Benchmarks User Guide. Technical Report 1991-IWLS-UG-Saeyang, MCNC, Research Triangle Park, NC, January 1991.
- [15] E.R. Gasner, E. Koutsifios, S.C. North and K.P. Vo. A Technique for Drawing Directed Graphs. *IEEE Trans. Software Engg.*, 19:214-230, 1993. Software available from <http://www.research.att.com/sw/tools/graphviz/>.
- [16] M. R. Hartoog. Analysis of Placement Procedures for VLSI Standard Cell Layout. In *23rd Design Automation Conference, ACM/IEEE*, pages 314-319, July 1986.
- [17] D. Ghosh, N. Kapur, J. Harlow, and F. Brglez. Synthesis of Wiring Signature-Invariant Equivalence Class Circuit Mutants and Applications to Benchmarking. In *Design Automation & Test - Europe (DATE'98)*, February 1998. Also available at <http://www.cbl.ncsu.edu/publications/>.
- [18] K. Kozminski, (Ed.). OASIS2.0 User's Guide. MCNC, Research Triangle Park, N.C. 27709, 1992. (Over 600 pages, distributed to over 60 teaching and research universities worldwide).
- [19] 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.