# Resolving Signal Correlations for Estimating Maximum Currents in CMOS Combinational Circuits

*Harish Kriplani*†, *Farid Najm*†, *Ping Yang*†† *and Ibrahim Hajj*†

†University of Illinois at Urbana-Champaign ††Texas Instruments Inc., Dallas, TX.

**Abstract:** *Currents flowing in the power and ground (P&G) lines of CMOS digital circuits affect both circuit reliability and performance by causing excessive voltage drops. Maximum current estimates are therefore needed in the P&G lines to determine the severity of the voltage drop problems and to properly design the supply lines to eliminate these problems. These currents, however, depend on the specific input patterns that are applied to the circuit. Since it is prohibitively expensive to enumerate all possible inputs, this problem has, for a long time, remained largely unsolved. In [1], we proposed a pattern-independent, linear time algorithm (*`iMax`*) that estimates an upper bound envelope of all possible current waveforms that result from the application of different input patterns to the circuit. While the bound produced by* `iMax` *is fairly tight on many circuits, there can be a significant loss in accuracy due to correlations between signals internal to the circuit. In this paper, we present a new partial input enumeration (*`PIE`*) algorithm to resolve these correlations and significantly improve the upper bound (in one case, reducing the error by 64% on a circuit with about 1,700 gates). We also show good speed performance, analyzing circuits with more than 20,000 gates in about 2 hours on a SUN ELC. We demonstrate with extensive experimental results that the algorithm represents a good time-accuracy trade-off and is applicable to large VLSI circuits.*

## 1 Introduction

Currents flowing in the power and ground (P&G) lines of CMOS digital circuits affect both circuit reliability and performance by causing excessive voltage drops in the lines. Furthermore, the severity of these voltage drop problems intensify with the continuing push for denser chips and finer technologies. Indeed, as is known from the classical scaling theory [2], as the minimum feature size and the supply voltage are scaled down, while the total chip power remains constant, the required supply currents increase. With higher currents flowing in narrower lines, the voltage drop in the supply lines goes up and quickly becomes a limiting factor in the design of VLSI chips. Furthermore, a lower supply voltage means that the noise margin for the correct operation of a transistor decreases. In short, in order to avoid logic errors, power and ground lines need to be carefully designed to take care of the increased voltage drops and reduced noise margins. This highlights the need for efficient CAD tools to estimate the power supply and ground currents. Since worst case currents determine worst case voltage drops, our research is focused on the problem of estimating *maximum current* waveforms in the power or ground lines.

The current drawn by a CMOS circuit is a complex function of input excitations. For each input pattern applied to the circuit, a different transient current waveform is drawn from the supply lines. An input pattern for a circuit with $n$ inputs consists of a vector of $n$ excitations, where each excitation could be either a stable input state i.e., *low* or *high*, or a transition i.e., *high* to *low* or *low* to *high*. In the presence of such input dependent and transient current waveforms, one must carefully define the notion of *maximum* current waveform. In [1], we proposed the *Maximum Envelope Current* (`MEC`) waveform as an estimate of the maximum current. The `MEC` waveform at a contact point is the *upper bound* envelope of all the transient current waveforms that result from the application of different input patterns to the circuit.

*Accurate* estimation of the `MEC` waveform at every contact point is NP-complete, as this problem can be transformed to a Boolean satisfiability problem [3]. In [1], we proposed a *pattern-independent, linear time algorithm*, called `iMax`, that provides an upper bound for the `MEC` waveform (for completeness, this algorithm is summarized in the next section). However, in order to maintain reasonable execution times, the `iMax` algorithm neglects various signal correlations that may exist inside a circuit. This can result in a significant loss in accuracy (i.e. a *loose* upper bound), even with the simple improvement heuristics used in [1].

The main contribution of this paper is a new *partial input enumeration* (`PIE`) algorithm that efficiently resolves these correlations and leads to significant improvements in the upper bound (in one case, reducing the error by 64% on a circuit with about 1,700 gates). This technique is based on (1) intelligently selecting a few *critical* input nodes and (2) enumerating a limited number of cases at these nodes to produce an overall improvement in the upper bound. It turns out that the choice of these critical nodes is the key, and we will present two heuristics for doing this that have shown good results in practice. While this technique may be

This research is supported by the Semiconductor Research Corporation (92-DP-109) and Texas Instruments Inc.



**To appear in 30th ACM/IEEE Design Automation Conference 1993.**

slower than the simple iMax technique, we still demonstrate good speed performance, solving circuits with more than 20,000 gates in about 2 hours on a SUN ELC. Our algorithm has the attractive property that it does an *iterative improvement*, so that one can stop it at any time, and still obtain a better upper bound than the simple iMax result. We will demonstrate with extensive experimental results that the PIE algorithm represents a good time-accuracy trade-off and is applicable to large VLSI circuits.

This paper is organized as follows. Following the next background section, we discuss the signal correlation problem in section 3. This is followed by a discussion of possible methods that can be used to resolve the signal correlations in section 4. In section 5, we present the partial input enumeration method. In section 6, we present experimental results on several benchmark circuits. Finally, the salient features of this paper are summarized in section 7.

## 2 Main Ideas of the iMax Algorithm

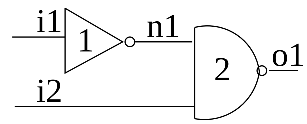
We will briefly review the iMax algorithm presented in [1], in which the following simplifying assumptions are made. Firstly, the combinational circuit under consideration is assumed to be part of a synchronous sequential circuit and, therefore, all of its inputs switch only at time zero. Secondly, the delay of each gate in the circuit is assumed to be fixed and is a user-specified number.

We define *excitation* at a node[1] at time $t$ as the stimulus (or signal value) present at the node at that time. At any time, a node in the circuit could be either stable at *low* or *high*, or could transition from *high* to *low* or from *low* to *high*. Thus, the excitation could be any single value from the set $\mathbf{X} = \{l,\ h,\ hl,\ lh\}$, where $l =$ *low*, $h =$ *high*, $hl =$ *high* to *low* transition and $lh =$ *low* to *high* transition. The set of all possible excitations that a node $\mathbf{n}$ can assume at any time $t$ is called the *uncertainty set* for the node at time $t$ and is denoted by $X_{\mathbf{n}}(t)$. Clearly, $X_{\mathbf{n}}(t) \subseteq \mathbf{X}$.

The iMax algorithm uses a gate level description of the circuit and, unless specified otherwise, assumes that the uncertainty set for each input at time zero is $\mathbf{X}$. The basic idea of the algorithm is to propagate the "uncertainty" present at the inputs inside the circuit so as to determine the entire range of possible excitations and their associated timing at the output of every logic gate. From this information, the worst case current waveforms are computed. The details of the algorithm can be found in [1].

An example illustrating the algorithm is shown in Fig. 1. In this example, we assume that the uncertainty set for each input at time zero is $\mathbf{X}$. Therefore, each input can transition from *low* to *high* or from *high* to *low* at time zero, or stay at *low* or *high* for all time. Transitions at various nodes of the circuit are represented by intervals. Thus, a transition at a specific time point $T$ can be represented by a closed interval which begins and ends at $T$. Given the above description at the input of the inverter ($i1$), the output ($n1$) can transition from *low* to *high* or from *high* to *low* at time 1 (assuming the delay of the inverter as 1 unit) or stay at *low* or *high* for all time. Similarly, assuming the delay of the NAND gate as 2 units, the output of the gate ($o1$) can transition $lh$ or $hl$ at time 2 due to the input $i2$, or transition $lh$ or $hl$ at time 3 due to the output of the inverter ($n1$); or stay at *low* or *high* for all time. In this fashion, the algorithm computes the set of all possible transitions at the output of every logic gate. The current waveform of each gate is calculated from this set of all possible transitions, and then the current waveforms from different gates are combined at the contact point(s). As each gate current is computed from the set of all possible transitions, the current waveform at the contact point is a point-wise upper bound on the MEC waveform (also see [1]).

[1] A *node* in a circuit is either a primary input or the output of a gate.




**Input Description** : i1, i2 $\in \{l, h, hl, lh\}$ at time 0.
**Uncertainty Intervals** :
i1, i2: $lh[0,\ 0]$, $hl[0,\ 0]$, $l[0,\ \infty)$, $h[0,\ \infty)$
n1: $lh[1,\ 1]$, $hl[1,\ 1]$, $l[0,\ \infty)$, $h[0,\ \infty)$
o1: $lh[2,\ 2][3,\ 3]$, $hl[2,\ 2][3,\ 3]$, $l[0,\ \infty)$, $h[0,\ \infty)$
**Key** : Excitation[Interval Begin, Interval End]

**Figure 1.** An example illustrating the iMax algorithm.

In order to assess the quality of the upper bound obtained from iMax, we need to determine the exact MEC waveform. However, as mentioned earlier, doing so is practically impossible for most circuits with more than about 10 inputs. We therefore, use an iterative optimization scheme, namely the *simulated annealing* (SA) algorithm [4], to calculate a current waveform that is close to the MEC waveform. In SA, different input patterns are selectively applied to the circuit and then a logic simulator is used to calculate the outputs of various gates. From these gate outputs, the supply currents are easily calculated. We use the peak value of the overall current waveform as the objective function to be maximized in the annealing algorithm. Since we cannot afford to examine all input patterns, the results of SA will be only a *lower bound* for the MEC waveform. By comparing the upper bound obtained from iMax to this lower bound, we obtain a measure of the maximum deviation of the iMax upper bound from the true MEC [1].

## 3 The Signal Correlation Problem

In general, signal values at internal nodes of a circuit are *correlated*. This limits the number of transitions that can possibly occur at the outputs of the gates, an effect that is ignored by the iMax algorithm [1]. An example of how signal correlation limits the number of transitions is illustrated in Fig. 2.

In this figure, the signal lines $x1$ and $x2$ are correlated i.e., they carry the same signal. Depending upon the specific excitation present at $x$, only one of the two gates can switch at a time. However, since iMax ignores the signal correlation present between $x1$ and $x2$, it erroneously concludes that both gates may switch at the same time. It is this kind of approximation that contributes to a loose iMax upper bound. As is clear from this example, the source of the signal correlation problem, in general, is a gate (or input) whose output

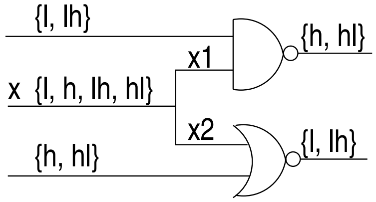



**Figure 2.** The signal correlation problem.

*fans out* to several other gates. Such gates are called *multiple fan-out* (MFO) gates.

The basic `iMax` algorithm ignores all signal correlations and, therefore, overestimates the supply currents. The advantage of ignoring correlations in the algorithm is its, very desirable, linear time performance.

## 4 Resolving Signal Correlations

The upper bound produced by the `iMax` algorithm can be made exact by doing a brute-force enumeration at the inputs of the circuit and storing the envelope of the current waveforms produced. In enumeration, since unambiguous input patterns are applied to the circuit, there is no uncertainty present at the inputs and therefore, signal correlations do not become an issue. In a similar fashion, one can improve the results of the `iMax` algorithm by doing a *partial enumeration* at a few selected nodes in the circuit.

An example of how the partial enumeration helps improve the upper bound can be seen from Fig. 2. In this circuit with no enumeration, `iMax` would assume that the signal lines $x1$ and $x2$ are mutually independent and therefore, infer that both the NAND and the NOR gates can switch at the same time. However, if we do a partial enumeration at signal line $x$, then we would generate four cases corresponding to when $x = l$, $x = h$, $x = hl$ and $x = lh$. When $x = l$ or $hl$, only the NOR gate switches. Similarly, when $x = h$ or $lh$, only the NAND gate switches. Thus, by splitting the problem into four sub-problems, we have improved our result, i.e., found that only one of the two gates may switch at any given time.

While enumerating a node, we need to process only those gates that are affected by a change in excitation at the node. We define *COne of INfluence* (COIN) of a node as the set of all the gates that can possibly be affected by a change in excitation at that node. Thus, a gate is in the COIN of a node if it is either directly driven by it or is connected to the output of a gate that is in COIN. While enumerating a node, we only need to consider those gates that are in its COIN.

One technique to partially enumerate the internal nodes of a circuit, called *Multi-Cone Analysis* (`MCA`), was reported in [1]. The motivation behind such an approach was to be able to enumerate at the outputs of the MFO gates, which are the sources of the signal correlation problem. However, the `MCA` approach offers only modest improvement in results. In the next section, we present a partial input enumeration approach that significantly improves the `iMax` results and represents a good speed-accuracy trade-off.

## 5 Partial Input Enumeration (`PIE`)

There are usually many more MFO nodes than primary inputs in a circuit. Secondly, as stated in section 2, all the inputs to the circuit switch at most once at time 0. Therefore, there is only one time point at which a primary input needs to be enumerated. This is in contrast to an internal node which usually needs to be enumerated at several time points. These observations, combined with the fact that `iMax` is an extremely fast algorithm led us to explore the following *partial input enumeration* (`PIE`) method to improve the upper bound.

Let $x_1$, $x_2$, ..., $x_N$ be the $N$ primary inputs of a circuit under consideration. Let $X_i$ represent the uncertainty set for input $x_i$ at time 0. The *input search space* for the circuit consists of all valid input patterns that can be applied to it. Mathematically, the input search space is $\{(e_1, e_2, \ldots, e_N) \mid e_1 \in X_1,\ e_2 \in X_2, \ldots,\ e_N \in X_N\}$. For brevity, we denote this by $(X_1, X_2, \ldots, X_N)$. Suppose, for the purposes of this illustration, for a particular input $x_i$, $X_i = \mathbf{X}$. Then the input search space $(X_1, X_2, \ldots, X_N)$ for the circuit can be divided into four disjoint parts, namely $(X_1, X_2, .., \{l\}, .., X_N)$, $(X_1, X_2, .., \{h\}, .., X_N)$, $(X_1, X_2, .., \{hl\}, .., X_N)$ and $(X_1, X_2, .., \{lh\}, .., X_N)$. We can compute the maximum current waveforms for each of these four parts by running the `iMax` algorithm. Since the four parts combined together constitute the complete search space, by taking an upper bound envelope of the four current waveforms, we can still guarantee an upper bound on the `MEC` waveform. Since, in each of the four runs of `iMax`, specific excitation values are present at input $x_i$, signal correlations due to $x_i$ disappear and the resulting current waveform should be an improvement on the original upper bound. In a similar fashion, the upper bounds for the individual subcases can be improved.

The set of inputs selected for enumeration has a definite influence on the quality of the solution obtained. If all the inputs are selected then the upper bound obtained would be exact. However, doing this is practically impossible for most circuits. From Fig. 2, we observe that some inputs contribute more to signal correlation than others e.g., enumerating input $x$ is more beneficial than enumerating any of the other two. Hence, by selecting and enumerating inputs in an intelligent fashion, we can significantly improve the `iMax` upper bound, without spending too much cpu time.

We have implemented the partial input enumeration approach in the form of a *best first search* (BFS) algorithm [5]. Various *search nodes* (call it `s_nodes`) generated during the search correspond to partial input specifications, as explained above. During the search, we always expand `s_nodes` which correspond to the highest peak value (objective of the search) of the upper bound waveform. Because of this best first strategy, there is a gradual reduction in the peak value of the upper bound. This is a very important feature of the algorithm for large circuits where an exhaustive exploration of the input space is practically impossible. The BFS algorithm can be stopped at any intermediate stage and the current best upper bound can still be reported.

The BFS algorithm starts with the initial uncertain state i.e., $(X_1, X_2, \ldots, X_N)$. During the search, a node with the highest objective value is repeatedly selected and its descendent nodes are generated by enumerating an input, as explained in the following outline. Here, *List* is an ordered list of `s_nodes`, arranged in their decreasing objective values.

1. *List* $\leftarrow$ Initial uncertainty state. `UB` $\leftarrow$ its `iMax` value. `LB` $\leftarrow$ `obj` value for an input pattern.

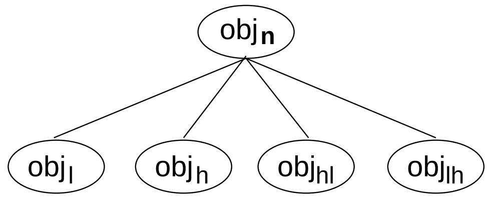



**Figure 3.** The $H_1$ `Splitting Criterion`.

**2.** While `Stopping Criterion` is not satisfied, do

- **2.1** Remove top `s_node` from *List*.
- **2.2** Calculate next input to enumerate from `SC`.
- **2.3** Generate all ($\leq 4$) children `s_nodes` by enumerating the above input and calculate their `obj` values.
- **2.4** If these children are leaf `s_nodes`, then update the `LB`, else, insert them in *List*, after pruning if any.
- **2.5** `UB` $\leftarrow$ `obj` value of top `s_node` in *List*.

**3.** Report the best `UB`, `LB` found. STOP.

The following functions are used in the algorithm.
**Objective Function:** It is the peak value of the upper bound waveform obtained from `iMax`.
**Stopping Criterion:** We stop the search when the number of `s_nodes` expanded during the search exceeds a certain user specified limit (`MaxNoNodes`).
**Pruning Criterion:** During the search, a `s_node` for which the upper bound exceeds the lower bound can be deleted from the search.
**Splitting Criterion (SC):** The splitting Criterion is a very important component of the BFS algorithm. This criterion specifies the input that should be enumerated next from any `s_node` during the search. Let us suppose that during the search, we are at a particular `s_node` $\mathbf{n}$ and we select an input $x_i$ for enumeration. If we assume that the uncertainty set for $x_i$ at time 0 is $\mathbf{X}$, then by enumerating $x_i$, we would generate four children `s_nodes`, as shown in Fig. 3. We assume that the objective value of `s_node` $\mathbf{n}$ is denoted by $obj_{\mathbf{n}}$ and the objective values of the children `s_nodes` are denoted by $obj_l$, $obj_h$, $obj_{hl}$ and $obj_{lh}$. If

$$\Delta obj_i = obj_{\mathbf{n}} - \max\{obj_l,\ obj_h,\ obj_{hl},\ obj_{lh}\},$$

then by enumerating $x_i$, we can improve the objective value of `s_node` $\mathbf{n}$ by an amount $\Delta obj_i$.

Based on this observation, we have come up with the following (more general) heuristic function called $H_1$:

$$\begin{aligned} H_1 \;=\; & A \times (obj_{\mathbf{n}} - obj_1) \;+\; B \times (obj_{\mathbf{n}} - obj_2) \;+ \\ & C \times (obj_{\mathbf{n}} - obj_3) \;+\; (obj_{\mathbf{n}} - obj_4) \end{aligned}$$

where $obj_1$, $obj_2$, $obj_3$ and $obj_4$ are the objective values of the children `s_nodes` arranged in decreasing order and $A$, $B$ and $C$ are three constants such that $A \gg B \gg C \gg 1$. At every `s_node` during the search, we calculate the heuristic value for every input and select the input with the maximum associated heuristic value. However, for large circuits, it is very expensive

**Table 1. Results of** `PIE` **for 10 ISCAS-85 circuits.**

| Circuit | iMax | Static $H_1$ SC | | | Static $H_2$ SC | | |
|---|---|---|---|---|---|---|---|
| | | BFS (100) | BFS (1k) | Time (100) | BFS (100) | BFS (1k) | Time (100) |
| c432 | 1.12 | 1.08 | 1.05 | 5m 14s | 1.12 | 1.12 | 1m 34s |
| c499 | 1.33 | 1.33 | 1.33 | 4m 40s | 1.33 | 1.33 | 1m 23s |
| c880 | 1.31 | 1.25 | 1.22 | 17m 16s | 1.28 | 1.26 | 4m 5s |
| c1355 | 1.52 | 1.52 | 1.52 | 21m 28s | 1.52 | 1.52 | 6m 13s |
| c1908 | 1.64 | 1.49 | 1.46 | 33m 17s | 1.58 | 1.54 | 11m 51s |
| c2670 | 1.35 | 1.29 | 1.28 | 1h 57m | 1.35 | 1.35 | 11m 56s |
| c3540 | 2.01 | 1.45 | 1.36 | 51m 12s | 1.59 | 1.37 | 17m 3s |
| c5315 | 1.48 | 1.42 | 1.40 | 3h 2m | 1.48 | 1.47 | 26m 2s |
| c6288 | 1.28 | 1.28 | 1.27 | 2h 5m | 1.28 | 1.28 | 57m 28s |
| c7552 | 1.57 | 1.52 | 1.50 | 6h 21m | 1.53 | 1.53 | 45m 4s |

to repeat this process at every `s_node`. Therefore, instead of calculating the heuristic value for every input at every `s_node`, the heuristic values for every input are calculated at the beginning of the search. All the inputs are arranged in decreasing order of these heuristic values and during the search, inputs are enumerated in this fixed order. This criterion is called *static* ($H_1$) *splitting criterion*.

The number of gates that are affected by a change in excitation at an input is another good heuristic measure of how much influence the input has on the upper bound waveform. Inputs which affect more number of gates (i.e., which have larger COINs) should be enumerated before others. This leads us to another (static) splitting criterion $H_2$, whose value is equal to the size of the COIN associated with the input. As with $H_1$, all the inputs are arranged in decreasing order of $H_2$ values and during the search, inputs are enumerated in this fixed order. We will show in the next section that, while both static $H_1$ and $H_2$ splitting criteria give good results in practice, $H_2$ is much better in terms of speed and has accuracy comparable to $H_1$.

## 6 Experimental Results

The results of partial input enumeration using the BFS algorithm and both $H_1$ and $H_2$ static splitting criteria for the ISCAS-85 benchmark circuits [6] are documented in Table 1. In the table, under various `iMax` and BFS columns, we show the ratio of the respective upper bound to the lower bound obtained from simulated annealing. The numbers in parentheses under the BFS columns indicate the number of `s_nodes` that were generated before stopping the search (i.e., the `MaxNoNodes` parameter; 1k stands for 1000). Total cpu times needed by the algorithm on a sun SPARC station ELC (with `MaxNoNodes` = 100) are also shown in the table.

From Table 1, we note that for all the circuits, the ratio of the upper bound to the lower bound is at most 1.52, as opposed to a worst case of 2.02 for the simple `iMax` algorithm. This ratio can be further improved by running the algorithm for longer durations. We emphasize that, since we can only compare the upper bound to a *lower bound* obtained from SA, the numbers in

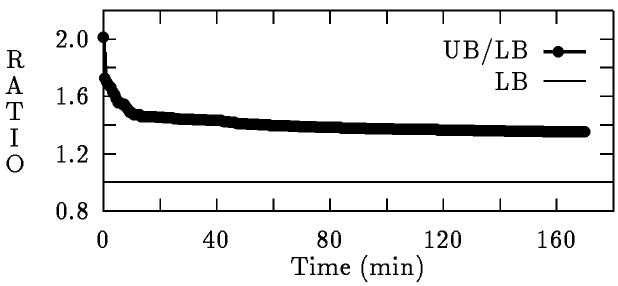



**Figure 4.** 'UB / LB vs Time' plot for c3540.

the table are only upper bounds on the error. It is prohibitively expensive to measure the true error.

While the improvement over the original iMax algorithm is not large in all cases, in those cases where the iMax bound *was* very loose, such as c3540, the new PIE algorithm gives *significant* improvement : the ratio of 2.02 (maximum over-estimation by 1.02) is now 1.37 (maximum over-estimation by 0.37) with $H_2$, a reduction in the maximum over-estimation by about 64%.

We also emphasize the following attractive property of the algorithm : a significant amount of improvement in the upper bound occurs in the first few s_node expansion (about 50-200) in the algorithm. This is illustrated in Fig. 4, where the ratio of the upper bound to the lower bound is plotted as a function of cpu time for c3540. The figure clearly indicates that our heuristics are working well to select the most critical s_nodes first. It also points to the fact that we can stop the search at any intermediate step and still be able to obtain some improvement in results. Similar behavior is observed for most other circuits.

The cpu time needed for generating the input list by the $H_2$ splitting criterion is negligible compared to the time needed by the $H_1$ criterion. For VLSI circuits with several hundred inputs, where the time needed by the $H_1$ criterion may be large, $H_2$ criterion may be used instead. As can be seen from Table 1, the results produced by using either splitting criteria are quite comparable, specially for those circuits where iMax did not produce a good upper bound.

In order to demonstrate the applicability of the partial input enumeration algorithm for VLSI circuits with several thousand gates, we have also experimented with the ISCAS-89 benchmark circuits [7]. For these synchronous sequential circuits, we have extracted the combinational blocks by deleting the flip-flops. These combinational blocks have gate counts ranging up to 22,000 and number of inputs ranging up to 1750. The results of the BFS algorithm on some of the ISCAS-89 circuits using the $H_2$ splitting criteria are summarized in Table 2. It is clear from the table that even for circuits of this size, our algorithms show good speed and accuracy performance.

**Table 2. Results of PIE for ISCAS-89 circuits.**

| Circuit | No. Gates | iMax | Static H2 SC BFS (100) | BFS (1k) | Time (100) |
|---|---|---|---|---|---|
| s1488 | 653 | 2.21 | 1.41 | 1.06 | 2m 49s |
| s1494 | 647 | 2.18 | 1.39 | 1.05 | 2m 51s |
| s5378 | 2779 | 1.38 | 1.30 | 1.23 | 13m 21s |
| s9234 | 5597 | 1.76 | 1.56 | 1.56 | 37m 18s |
| s13207 | 7951 | 1.37 | 1.30 | 1.26 | 36m 53s |
| s15850 | 9772 | 1.81 | 1.64 | 1.57 | 1h 11m |
| s35932 | 16065 | 1.66 | 1.56 | 1.56 | 2h 6m |
| s38417 | 22179 | 1.73 | 1.72 | 1.68 | 2h 46m |
| s38584 | 19253 | 1.45 | 1.39 | 1.37 | 2h 15m |

## 7 Conclusions

In this paper, we have described the signal correlation problem which arises while estimating maximum currents in CMOS combinational circuits by the pattern independent approach iMax [1]. We have presented a new *partial input enumeration* (PIE) algorithm to resolve the signal correlations and significantly improve the upper bound obtained from the iMax algorithm (in one case, reducing the error by 64% on a circuit with about 1,700 gates). We also show good speed performance, solving circuits with more than 20,000 gates in about 2.25 hours on a SUN ELC. The algorithm is based on the *best first search* (BFS) technique and represents a good time-accuracy trade-off. The PIE algorithm involves a search procedure, but this search need not be carried too deep to obtain good results. The algorithm is quite applicable to large VLSI circuits, as is demonstrated by the experimental results. In our future research, we plan to extend the study to include better delay models and to estimate worst case voltage drops in supply lines, using RC models, from the maximum current estimates.

## References


[1] H. Kriplani, F. Najm, and I. Hajj, "Maximum current estimation in CMOS circuits," in *Proceedings of 29th ACM/IEEE Design Automation Conference*, pp. 2–7, Anaheim, CA, June 8-12, 1992.

[2] C. Mead and L. Convey, *Introduction to VLSI Systems.* Reading, MA: Addison-Wesley, 1980.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractibility, A Guide to the Theory of NP-Completeness.* New York, NY: W. H. Freeman and Co., 1979.

[4] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 13 May 1983.

[5] J. Pearl, *Heuristics – Intelligent Search Strategies for Computer Problem Solving.* Reading, MA: Addison-Wesley, 1984.

[6] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a target translator in fortran," in *Proceedings of International Symposium on Circuits and Systems*, pp. 695–698, June 1985.

[7] F. Brglez, D. Bryan, and K. Koźmiński, "Combinational profiles of sequential benchmark circuits," in *Proceedings of International Symposium on Circuits and Systems*, pp. 1929–1934, May 1989.