計算機アーキテクチャ　115－11
設　計　自　動　化　78－11
（１９９５．　１２．　１４）

逐次改善的手法による組合せ回路の最大同時変化ゲート数の評価

張　凱　　篠木　剛　　林　照峯　　北　英彦

三重大学

〒514 三重県津市上浜町1515
三重大学 工学部 電気電子工学科
計算機工学研究室
☎ 0592-32-1211 内線 3901
zhang, shinogi, hayashi, kita@hayashi.elec.mie-u.ac.jp

CMOS回路では、各ゲートのスイッチングと消費電力とは密接な関係がある。そこで、本報告では、組合せ回路における同時にスイッチングするゲート数の最大値を求めるための手法を提案する。本手法は、最適化の一手法である反復改善法に基づいている。乱数で与えた初期入力ベクトルペアを、回路中の同時スイッチングゲート数が増える方向に、１ピンずつ順番に改善していく。これで得られる同時スイッチングゲート数の最大値は初期入力ベクトルペアに依存するため、複数の異なる初期入力ベクトルペアに対して改善を試み、最大スイッチングゲート数を得る。さらに、複数ピンを単位にして順番に改善する手法に拡張した。ISCAS'85のベンチマーク回路を用いた実験により、本手法の有効性が示された。

An Iterative Improvement Method for Evaluating the Maximum Number of Simultaneous Switching Gates for Combinational Circuits

Kai Zhang, Tsuyoshi Shinogi, Hidehiko Kita, Terumine Hayashi

Mie University

Computer Engineering Laboratory
Department of Electrical and Electronic Engineering
Faculty of Engineering, Mie University
1515 Kamihama-cho, Tsu-shi
Mie-ken, 514 JAPAN
Phone 0592-32-1211 ext. 3901
zhang, shinogi, hayashi, kita@hayashi.elec.mie-u.ac.jp

The switching gates in CMOS logic circuits are closely related to the power dissipation. This paper presents a new method of evaluating the maximum number of simultaneous switching gates for combinational circuits. In this method, an initial primary input vector pair is iteratively improved through changing its values pin by pin orderly so as to increase the number of switching gates in the circuit. To find a larger number of switching gates, the procedure is repeated with the different initial vector pairs generated randomly. Also, our method is extended to the method with multiple selected pins. Experimental results for ISCAS benchmark circuits show the effectiveness of our method.

1. Introduction

The severity of the problem of power dissipation increases in proportion to the level of integration of LSI. The advent of VLSI has led to much recent work on the estimation of power dissipation during the design phase ([1]-[4]), so that designs can be modified before manufacturing.

The switching gates in CMOS logic circuits are closely related to the power dissipation. We address the problem of evaluating the maximum number of switching gates for combinational circuits.

There have been published several methods of evaluating the maximum number of switching gates, such as the partial exhaustive enumeration method [5], the branch-and-bound method [6] and the method using genetic algorithm (GA) [7]. The partial exhaustive method is powerful though the procedure is simple, but it tends to spend too much computational power on the local optimization. The branch-and-bound method needs much more CPU time than the other methods. The GA method may be effective if the appropriate parameters can be provided. However, it is difficult to decide the values of parameter properly.

We propose a new method by which an initial primary input vector pair is iteratively improved through changing its values pin by pin orderly so as to increase the number of switching gates in the circuit. To find a larger number of switching gates, the procedure is repeated with the different initial vector pairs generated randomly. The rest of this paper is organized as follows. Section 2 indicates the origin of our research problem and explains the meaning of simultaneous switching gates. In Section 3, our iterative improvement method for evaluating the maximum number of simultaneous switching gates is presented. In Section 4, experimental results are shown.

2. Problem Formulation

Most of the power in CMOS circuits is dissipated by switching of output values of the gates in the circuits. The power dissipated by the circuit is given by:

$$P(V) = L * E^2 * \sum_i Ci * Ti(V) \qquad (1)$$

Where V ($V = ((u_1,u_2,\ldots,u_n),(v_1,v_2,\ldots,v_n))$ $u_j, v_j \in \{0,1\}$) is a primary input vector pair (hereafter, we call it vector pair), which means that the values given to the primary input pins are changed from $u_1,u_2,\ldots,u_n$ to $v_1,v_2,\ldots,v_n$. $P(V)$ denotes the power dissipation, Ci is the load capacitance of a gate i, E is the supply voltage and L is a constant. And $Ti(V)$ is 1 when the output of the gate i is changed by the vector pair V, otherwise, $Ti(V)$ is 0. Here, static currents are negligible compared to transient currents ([8],[9]). Although the load capacitance Ci varies with the number of fanouts and the other factors of each gate, it is assumed to be identical in order to make the problem simple in this paper. The program can be easily modified by giving the weight, corresponding to the load capacitance, to each gate. As E is a constant as well, the equation(1) can be changed to the following equation (2).

$$P(V) = K\sum_i Ti(V) \qquad (2)$$

Here, K is a constant and $\sum Ti\ (V)$ is the number of simultaneous switching gates. If we can find the maximum number of switching gates, we can evaluate the maximum power dissipation.

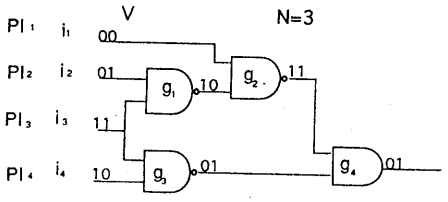


Fig.1 Example 1 for CMOS Circuit

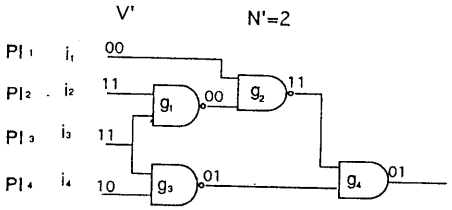


Fig.2 Example 2 for CMOS Circuit

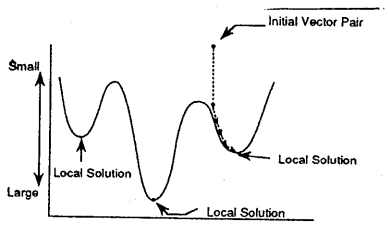


Fig.3(a) a Single Initial Vector Pair

Next, we explain the meaning of simultaneous switching gates. A gate of which the output value changes with the input vector pair is called the switching gate. In Fig.1, suppose that the vector

pair$V$ = ((0, 0, 1, 1), (0, 1, 1, 0)) is given for the primary inputs, i_1, i_2, i_3, i_4, then the output values of g_1, g_3, g_4, are changed from 1, 0, 0 to 0, 1, 1, respectively. The output value of g_2 remains unchanged. The gates g_1, g_3, g_4, are switching gates. There are three simultaneous switching gates for the vector pair V. If the vector pair V is changed to V' (Fig.2), there are two simultaneous switching gates for the vector pair V'. It is understood that the number of simultaneous switching gates varies with vector pairs.

3. Iterative Improvement Method

3.1 Overview of the Method

Our approach to evaluate the maximum number of simultaneous switching gates is to modify the values of an initial vector pair pin by pin orderly toward the increase of the number of simultaneous switching gates, as follows. This approach is called the iterative improvement method.

First, generate an initial vector pair randomly and compute the number of switching gates for it by logic simulation.

Then, select a primary input pin; modify the corresponding values of the selected pin in the vector pair; and compute the numbers of switching gates for the new vector pairs. Additionally, keep the "current best vector pair" that brings the current largest number of switching gates obtained so far.

This step is repeated by carrying the "current best vector pair" to the next repetition. In the next repetition, another primary input pin is selected to modify the "current best vector pair". The repetition continues until the current largest number of switching gates does not increase even though the values of every primary input pin are modified.

As the finally obtained largest number of switching gates depends on the initial vector pair, the above procedure is applied to multiple initial vector pairs.

Before describing the procedure of the iterative improvement method in detail, we illustrate it through the example in Fig. 4.

First, an initial vector pair V is generated randomly, as V = ((1, 0, 0, 0), (0, 0, 1,0)) in the left part of Fig. 4, and the number N of switching gates for V is computed, as N=0 unfortunately in this case.

Secondly, select a primary input pin for improvement at random. We call it the "selected pin". Now, suppose that PI2 is the selected pin. Compute the numbers N', N'', N''' of switching gates for V', V'', V''' modified from V. Although V, V', V'', V''' are different from each other, they have the same values except the selected PI2. In this case, the value on PI2 for V is 00, then the values on PI2 for V', V'', V''' are 01, 10, 11, respectively, and the computed values of N', N'', N''' are 1, 2, 3, respectively. The maximum number of switching gates (N''') and its vector pair (V''') are reserved. As the maximum number of switching gates (N''') in this step has become larger than the number of switching gates (N) for the vector pair (V), we call such a step an improved process and the new vector pair (V''' in this step) is carried to the next step. If every value of N', N'' and N''' does not exceed N, we call it an unimproved process. In this case, the vector pair which was carried from the previous step survives and passes to the next step. In this way, the "current best vector pair" that brings the current largest number of switching gates obtained so far is carried to the next step.

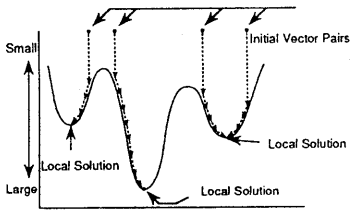


Fig.3(b) Multiple Initial Vector Pairs

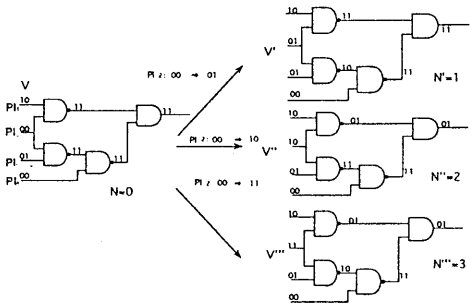


Fig.4 Example for Pattern Generation (1/3)

In the next step (Fig. 5), the next primary input pin, PI3, is selected as the selected pin. And the same procedure is repeated. As the maximum number N''' in this step is equal to the current largest number N, this step is judged as an unimproved process, the vector pair$V$ that was carried from the previous step passes to the next step. Next, PI4 becomes the selected pin. The same procedure is repeated as in Fig. 6.

This repetition continues until the current largest number of switching gates does not increase even though the values of every primary input pin are modified. This condition is determined with the times of successive unimproved processes that have reached the number of primary input pins.

As the finally obtained largest number of switching gates depends on the initial vector pair as explained generally in the next paragraph, the above overall procedure is repeated again and again, starting with another initial vector pair generated randomly.

We generally explain the necessity of the repetition. Fig. 3(a) represents the behavior without any repetition. In this case, only one of the local maximum solutions will be obtained. Fig. 3(b) implies the effect by multiple initial vector pairs generated randomly. In this case, a better solution can be obtained.

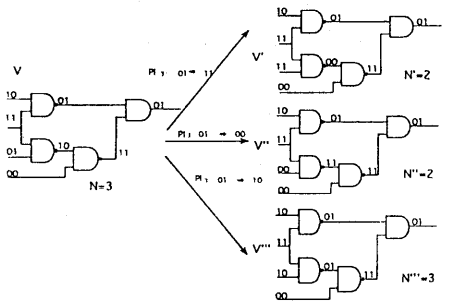


Fig.5 Example for Pattern Generation (2/3)

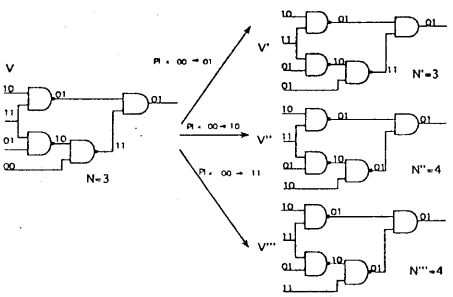


Fig.6 Example for Pattern Generation(3/3)

3.2 Procedure for the Iterative Improvement Method

The main flow of our method is as shown in Fig.7, and the procedure is described as follows:

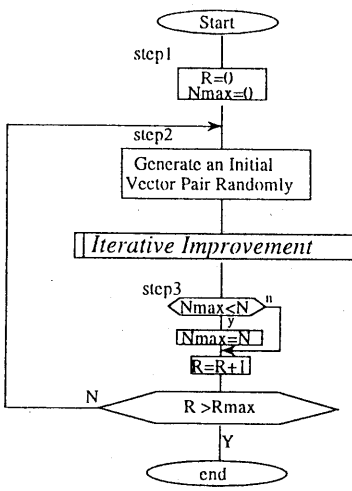
The Procedure of our method:

(*Step1*) Set R and $Nmax=0$. Here, R denotes the counter of the main repetition, and $Nmax$ is the maximum number of switching gates.

(*Step2*) Generate an initial vector pair V at random. Then, enter the iterative improvement. This part is shown at Fig.8 and will be explained later.

(*Step3*) Suppose that N is the result of the iterative improvement. Set $Nmax = N$, if N is larger than $Nmax$. And increase the repeated counter R by 1. If R has reached the predetermined limit value $Rmax$, the procedure concludes. Otherwise, go back to *Step2*.

It follows that the flow of iterative improvement is as shown in Fig.8, and the procedure is described as follows:



R : the number of initial two-pattern tests
Nmax: the maximum number of switching gates
N : number of switching gates computed for V

Fig.7 Main Flow of Our Method

The Procedure of Iterative Improvement:

(*Step1*) Compute the number N of switching gates for the vector pair V by logic simulation.

(*Step2*) Select a primary input pin $PIi\ (1 \leq i \leq n)$ at random for initial improvement, where n denotes the number of the primary inputs. Set $K=0$. Here, K is used for counting the times of successive unimproved processes.

(*Step3*) Compute the numbers N', N'' and N''' of switching gates for V', V'' and V''' modified from V. Although V, V', V'' and V''' are different from each other, they have the same values except the value on the primary input pin PI_i. For example, if the value on PI_i for V is 01, then the values on P_i for V', V'' and V''' are 00, 10 and 11, respectively. Let Nc be the maximum number of N', N'' and N''', and let Vc be the vector pair that brings the number Nc.

(*Step 4*) If $N < N_C$, then set $V=V_C$, $N=N_C$, $K=0$ and $i=i(mod\ n)+1$, and go back to *Step 3*. Otherwise, proceed to the next *Step 5*.

(*Step 5*) Set $K=K+1$. If $K \geq n$, then return to the main flow. Otherwise, set $i=i(mod\ n)+1$ and go back to *Step 3*.

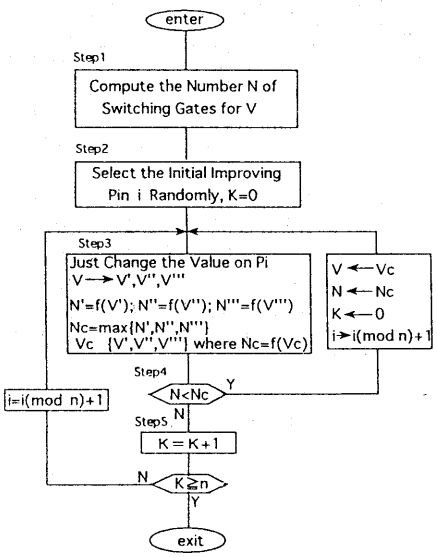


Fig.8 Flow of Iterative Improvement

3.3 Iterative Improvement by Multiple Selected Pins

In the above iterative improvement procedure, only a single pin is selected for improvement each time. We can select multiple pins instead of a single pin. The number of switching gates is computed for every combination of values on the selected pins. We call the method of iterative improvement with multiple pins IIP(k). Here, k is the number of the selected pins and in section 3.2, k is equal to 1. The larger the number of selected pins is, the greater is the possibility of finding the optimal solution. When the number of selected pins is equivalent to the number of primary input pins, it results in exhaustive enumeration and the optimal solution can be found. As the exhaustive enumeration needs enormous computational time, it is impracticable. The more the number of selected pins is, the more computational time is needed, and therefore we select only 2 pins in view of the limits permitted by time.

We consider a representation of the procedure for IIP(2) similar to that for IIP(k) (k >= 3) as follows.

First, an initial vector pair V is generated randomly, and the number N of switching gates for V is computed.

Secondly, select two successive primary input pins for improvement at random. For example, the pin PI_j and PI_{j+1} are the selected pins. Compute the numbers $N1, N2...,N15$ of switching gates for $V1, V2,...,V15$ modified exhaustively from V. Although $V, V1,.....,V15$ are different from each other, they have the same values except the values on the primary input pins PI_j and PI_{j+1}. The maximum number among $N, N1,..., N15$ is reserved and its vector pair is carried to the next step. In the next step, the input pins for improvement are moved to PI_{j+1} and PI_{j+2} and the above procedure is repeated. The iterative improvement continues until the times of successive unimproved processes has reached the number of primary input pins.

Finally, the above-mentioned procedure is repeated with the different initial vector pair generated randomly.

4. Experimental Results

Our method has been implemented on a personal computer 466/m DELL Computer Corporation. We used the ISCAS'85 benchmark circuits[10] for our experiments. There have been published several methods of evaluating the maximum number of switching gates, such as the partial exhaustive enumeration method[5], the genetic algorithm method[6] and the branch-and-bound method[7]. Table 1 shows our experimental results together with the results of the previously published methods.

In the case of our method IIP(1), we have obtained larger *Nmax*'s for c880, c1908, c5315, c6288 and c7552 than those by the previous methods. Moreover, the method IIP(2) has not only enhanced the results obtained by the IIP(1) for c880, c1908, c5315 and c7552, but also found the largest *Nmax*'s for c2670 and c3540 among all methods in the table, though the CPU time has increased. Notably, the *Nmax*'s for c5315 and c7552 by our method are much larger than those by the previously published methods. This shows the effectiveness of our method for large circuits.

5. Conclusions

We presented a new method by which an initial primary input vector pair is iteratively improved through changing its values pin by pin orderly so as to increase the number of switching gates in the circuit. To find a larger number of switching gates, the procedure is repeated with the different initial vector pairs generated randomly. Also, our method is extended to the method with multiple selected pins. For each circuit of ISCAS'85, we compared the number of the switching gates obtained by our method to those obtained by the previous methods. These results show the effectivess of our method. Especilly, our method is effective for the large-scale circuits.

Table 1. the Comparison between our Method and the Previous Methods

circuit name	our method				partial exhaustive enumeration [5]				method using genetic algorithm [6]				branch-and-bound method [7]	
	1pin (Rmax:150)		2pins(Rmax:150)		RND2(N=6)*5		back operation *5		M1*6		M2*6			
	Nmax	CPU *1	Nmax	CPU*1	Nmax	CPU*2	Nmax	CPU*2	Nmax	CPU*3	Nmax	CPU*3	Nmax	CPU*3
c432	145	16	145	53	---	---	---	---	---	---	---	---	---	---
c499	119	12	119	63	---	---	---	---	---	---	---	---	---	---
c880	318	81	319	399	300	180	315	91	318	468	315	373	313	2521
c1355	282	55	295	342	296	269	290	378	288	203	296	309	305	3337
c1908	595	86	598	467	591	241	592	395	588	438	587	307	590	3676
c2670	793	869	807	4640	758	826	776	623	791	2439	755	1368	806	6024
c3540	917	290	923	1442	915	454	904	726	919	833	901	495	869	8381
c5315	1461	1465	1478	7845	1429	1406	1412	1511	1402	4528	1449	3005	1434	36000
c6288	1562	326	1556	1591	1556	1484	1449	823	1538	1226	1539	1100	1516	6511
c7552	2150	2648	2172	13716	2094	2974	2125	2114	2100	6434	2099	4462	2133	10878

*1 cpu time (in sec.), on DELL466/M

*2 cpu time (in sec.), on Fujitsu S-4/LC

*3 cpu time (in sec.), on Sun SS / Classic

*5 the difference for initial pattern.

*6 the difference for probability of mutation.

References

[1] A. Ghosh, S. Devadas, K. Keutzer and J. White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," *Proc. 29th Design Automation Conference*, pp.253-259, 1992.

[2] A. P. Chandrakansan, S. Sheng and R. W. Brodersen, "Low-Power CMOS Digital Design," *IEEE Journal of Solid-State Circuits*, vol.27, no.4, pp.473-483, April 1992.

[3] S. Chowdhury and J. S. Barkathullah, "Estimation of Maximum Currents in MOS IC Logic Circuits," *IEEE Trans. on Computer-Aided Design*, vol.9, no.6, pp.642-654, June 1990.

[4] S. Iman and M. Pedram, "Multi-Level Network Optimization for Low Power," *Proc. ICCAD-94*, pp372-377, 1994.

[5] H. Ueda and K. Kinoshita, "Evaluation of the Maximum Number of Switching Gates for CMOS Circuits," *Technical Report of IEICE, Japan*, vol.93, no.303, pp.49-56, 1993.

[6] H. Ueda, A. Kiyama and K. Kinoshita, "Evaluation of the Maximum Number of Switching Gates for CMOS Logic Circuits Using Genetic Algorithm," *In the 33rd FTC Meeting, Japan*, 1995.

[7] H. Ueda and K. Kinoshita, "Evaluation of the Maximum Number of Switching Gates for CMOS Circuits," *IEICE Trans. Inf.&Syst.*, vol.J78-D-I, no.3, pp.367-375, Mar. 1995.

[8] P. Nigh and W. Maly, "Test Generation for Current Testing," *IEEE Design and Test*, vol.7, no.2, pp.26-38, Feb. 1990.

[9] C. H. Chen and J. A. Abrahan, "High Quality Tests for Switch-Level Circuits Using Current and Logic Test Generation Algorithms," *Proc. Int. Test Conf.*, pp.615-622, Oct. 1991.

[10] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translation in Fortran," *ISCAS'85 : Special Session on ATPG and Fault Simulation*, 1985.