Electrical Engineering (2005) 87: 11–18
DOI 10.1007/s00202-003-0215-1

ORIGINAL PAPER

Sami H. Karaki · Ayman I. Kayssi · Houda S. Karaki

Capacitor placement for switching noise reduction using genetic algorithms and distributed computing

Received: 18 September 2003 / Accepted: 7 November 2003 / Published online: 15 January 2004

Abstract This work investigates capacitor placement on a printed circuit board (PCB) to reduce the effect of simultaneous switching noise as a genetic algorithm (GA) search problem. The solution process makes use of distributed computing resources available on a local area network in order to solve larger problems more efficiently. The objective is to determine the number of added capacitors with minimum cost, and their position on the PCB, while keeping the maximum voltage deviation within some specified noise margin. The presence of capacitors at the selected positions is represented by a stream of zeros and ones, which is interpreted as a genotype. At each generation, the GA assesses the fitness function of a population of genotypes using linear transient circuit analysis, which involves a single matrix inversion, by determining the maximum voltage dip given the capacitor locations. For large systems, the fitness calculations are divided among several processors according to a simple distributed computing algorithm.

Keywords Capacitor placement · Genetic algorithms · Simultaneous switching noise · Printed circuit board · Distributed computing · Power distribution grid

1 Introduction

Simultaneous switching noise (SSN) is a phenomenon that occurs when a large number of high-speed chip drivers switch simultaneously, causing a large current to be injected in the power distribution grid of a printed circuit board (PCB). Inductively induced voltage fluctuations in the power distribution grids of a PCB often manifest themselves in a variety of transient and permanent system malfunctions including the appearance of undesirable glitches on lines and the flipping of bits in registers and memories.

Many authors have modeled and analyzed the SSN phenomenon, and to mitigate its effects proposed placing decoupling capacitors on the PCB. Smith et al. [7] studied impedance versus frequency profiles of the power distribution system components of integrated circuit CMOS boards including the voltage regulator module, bulk decoupling capacitors, and high frequency ceramic capacitors to deduce simulation models. The models are analyzed in the time domain to find the response to load transients. Chen et al. [1] proposed a signal integrity analysis technique for simulating voltage fluctuations on power/ground planes in complex packaging structures using circuit and electromagnetic field solvers to determine the value, the number, and the location of the decoupling capacitor placed on packages or printed circuit boards. Drewniak et al. [3] used numerical modeling based on an integral equation formulation with circuit extraction to quantify the local decoupling phenomenon. They demonstrate that local decoupling can effectively reduce high-frequency power-bus noise for some PCB geometries. Yook et al. [10] presented a methodology combining macro- and micromodels, which allows for a system-level treatment of the problem without losing the necessary detailed descriptions of the power/ground planes, the signal traces, and the vertical interconnections through vias or plated holes. Capacitor placement is a traditional nonlinear optimization problem in power systems that was solved, using genetic algorithms, by Iba [4] and Sundhararajan and Pahwa [8] to determine the minimum reactive power compensation required for voltage support under heavy loading conditions.

Earlier work [5] investigated the placement of single-valued decoupling capacitors at selected positions of a PCB to reduce the effect of SSN using a genetic algorithm (GA) approach. This paper extends the GA formulation to determine the optimal placement of multi-valued capacitors on a PCB and presents a

S. H. Karaki · A. I. Kayssi (✉) · H. S. Karaki
Department of Electrical and Computer Engineering,
American University of Beirut, Riad El-Solh 1107 2020,
P.O. Box 11-0236, Beirut, Lebanon
E-mail: ayman@aub.edu.lb

distributed computing circuit evaluation approach using available local area network (LAN) resources, thus allowing larger systems to be studied. The objective is to reduce the cost of added capacitors while keeping the maximum voltage dip or ground bounce within some specified noise margin. The presence of capacitors of known values at the selected positions is represented by a stream of zeros and ones, which is interpreted as a genotype and manipulated using GA operators to systematically approach the optimal solution. At each step or generation of the GA the fitness evaluations of genotypes are assessed using linear transient circuit analysis [6] by determining maximum voltage dip, given the values and locations of capacitors specified in the genotype. The circuit analysis is made more efficient by formulating the problem in such a way that the transient analysis of a genotype involves the formation and inversion of the nodal admittance matrix only once at the onset of the circuit evaluation process.

The rest of the paper is organized as follows. Section 2 presents the formulation of the capacitor placement problem. Genetic algorithms are reviewed in Sect. 3. Section 4 presents the distributed computation technique used to speed up the GA search. The transient circuit analysis method is discussed in Sect. 5, and results are presented in Sect. 6. Conclusions are given in Sect. 7.

2 Problem formulation

The formulation of capacitor placement on a PCB as a GA search problem is best explained through an example. Consider the system shown in Fig. 1, which represents a PCB layout containing 16 integrated circuits (IC) and the corresponding 16 possible places to place capacitors, with values of 0.5 or 1 μF. Each inch of wire is modeled as an RLC section, and the IC is modeled as a current source with a triangular wave shape, as shown in Fig. 2. The behavior of the circuit in terms of the voltage deviation resulting from the simultaneous switching of the ICs can be predicted using the linear transient analysis method [6]. The location and value of capacitors will have a significant influence on the observed maximum voltage deviation, and the problem is to determine the minimum cost of capacitors that will yield a maximum voltage dip smaller than some specified noise level. The presence of capacitors at their proposed values and locations can be indicated by two simple arrays as shown in Fig. 3, which indicate that nodes 9, 19, and 43 have capacitors of 0.5 μF, and that nodes 31 and 39 have 1-μF capacitors, whereas other nodes do not have any.

The use of GA will permit a systematic search of the possible combinations in order to determine a "best" solution without necessarily being exhaustive in the search. The quality of a particular design is evaluated by a cost or fitness function, which decreases with an increasing cost of capacitors and voltage deviation. The fitness function is given by:

$$F(C_G, \Delta V_{\max}) = \alpha C_G + F_{\max} + \gamma(\Delta V_{\text{spec}} - \Delta V_{\max})$$

where C_G is the cost of capacitors used in a particular genotype G, $F_{\max}$ and $F_{\min}$ are the maximum and minimum fitness values, and $\alpha = (F_{\min} - F_{\max})/C_{\max}$, where $C_{\max}$ is the total cost of capacitors. The constant γ should be selected so that the largest permissible voltage deviation (ΔV_{spec}) does not bring the fitness of a solution with C_G capacitors below that of a solution with C_{G+1}

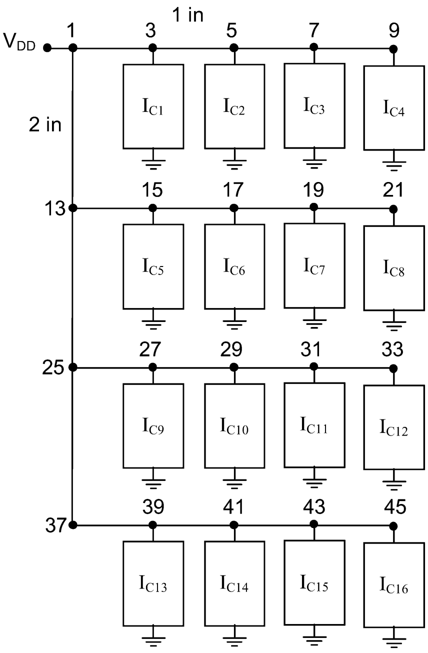


Fig. 1 Example PCB layout

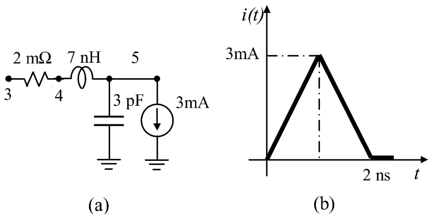
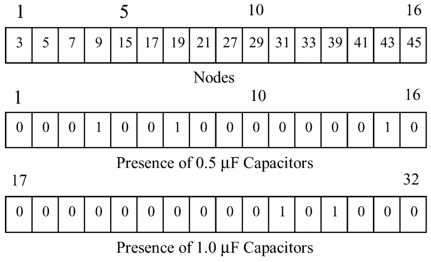


Fig. 2 Equivalent RLC 1-in. section with current source (**a**) and corresponding wave shape (**b**)


1
5
10
16

3	5	7	9	15	17	19	21	27	29	31	33	39	41	43	45

Nodes

1
10
16

0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0

Presence of 0.5 μF Capacitors

17
32

0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Presence of 1.0 μF Capacitors

Fig. 3 Arrays indicating nodal location and presence of capacitors

capacitors, which means that γ should obey the following inequality:

$$\gamma \leqslant -\frac{\alpha}{\Delta V_{\mathrm{spec}}}.$$

3 Genetic algorithms overview

Many real world problems are complex and have high-dimensional spaces with limited domain knowledge that is difficult to search using brute force, even when using powerful computers. Genetic algorithms (GA) constitute a systematic approach to sample the solution space and reach an ‘‘optimum’’ solution in a process that mimics natural biological evolution by applying the genetic operators of selection, crossover, and mutation on a population of individuals to produce a new, fitter one. GA can be customized to suit any given application and has no restrictive assumptions on the objective function but requires some experimentation to find good settings of its strategy parameters, namely the population size, the crossover and mutation probabilities, and the number of generations, which are problem-dependent. Even though GA are heuristic in nature, with no guarantee of reaching a global optimum, there is usually a high confidence associated with the individuals produced at the end of the search. GA may have high CPU requirements but, even in the simple algorithm used in this paper, they inherently permit the use of parallel or distributed computing for evaluating the fitness function of individuals within a population.

The GA process (Fig. 4) starts from an initial population of N individuals, produced at random. The fitness of each of the N individuals is then evaluated, and M of these individuals (parents) are selected to produce new individuals (offspring) using a crossover process. A process called mutation is usually applied to avoid getting stuck in local minima manifested by premature convergence. To appreciate further the GA search mechanism we shall now briefly describe the three genetic operators of selection, crossover, and mutation.

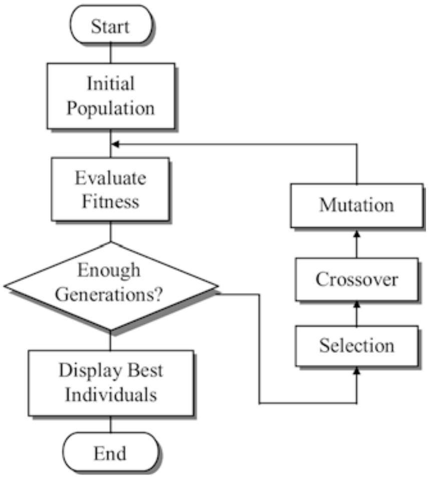


Fig. 4 Genetic algorithm structure

3.1 Selection

The two methods most commonly used for selection are the roulette wheel and the tournament selection methods. In the roulette wheel method the individuals are aligned as contiguous segments of a line with each segment length proportional to its fitness, as shown for 10 individuals in Fig. 5. A random number, of uniform distribution, is then generated and the individual whose segment spans the random number is selected. The process is repeated over several trials until the desired number M of individuals is obtained.

In tournament selection a number of K individuals are chosen at random from the N offspring and the fittest of this group is selected. This process is repeated M times to complete the mating population. Here the choice of K is critical, as a small value makes the chance of selecting weaker individuals higher.

3.2 Crossover

This is the process by which the genotypes of two parents from the selected individuals are mixed to produce two offspring. As illustrated in Fig. 6, a cut is taken at random and the genes of the parents to the left of the cut are interchanged to produce the two offspring. This is a

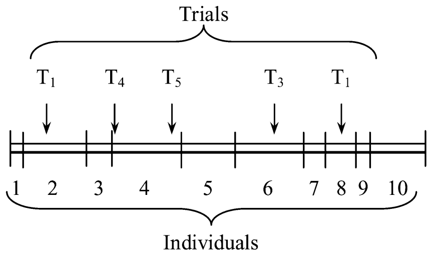


Fig. 5 Roulette wheel selection

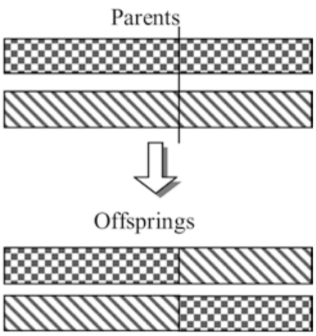


Fig. 6 Single point crossover

single point crossover process. In a multi-point crossover two or more cuts are taken to produce more offspring [2].

3.3 Mutation

This is the process whereby a gene in an offspring genotype is randomly changed from 0 to 1 or vice versa with a probability that is usually kept low to allow some properties to appear in individuals but not create disorder in the search process as a whole. A proper selection of the mutation probability will avoid convergence to a local minimum and will thus improve the quality of solutions produced.

4 Distributed processing

In this work, the distributed computing process is applied on the fitness evaluation of the population of each generation. All processes shown in Fig. 4 except the ‘‘Evaluate Fitness’’ process would be carried out on the main processor, termed a master. In the fitness evaluation phase, as illustrated in Fig. 7, the members of one population are divided equally into groups and distributed among the master and several other computing processors (the slaves). At the onset of the distributed computing process the master reads from a file the network address of each potential slave that would contribute to the problem solution and identifies its readiness through the presence of a flag. The circuit data file is then read by the master and dispatched to the available slave processors which would then have all information on the circuit problem being solved, namely the network topology, parameters, and location and size of the decoupling capacitors to be added.

Each slave receives a set of genes, which is interpreted and processed using the function ‘‘Evaluate Circuit’’. The slave produces results corresponding to that particular genotype, which are the maximum voltage deviation observed in the circuit evaluation ($\Delta V_{\max}$) and the node at which it occurs ($N_{\max}$). The master also follows the same process on its assigned group. Once all the slaves and the master are done evaluating the whole population, the master uses the collected data to carry out the genetic operators on the present population or to stop when enough generations have been processed, as illustrated in Fig. 4.

To keep track of which members each slave received, the master has some bookkeeping to do; it assigns to each slave a start index, a genotype size, and number of genotypes sent to a particular slave. When slave X, for example, receives a group of genes, it will divide them into several genotypes, given the genotype size, and evaluates for each the corresponding $\Delta V_{\max}$ and $N_{\max}$ in the same order in which the group of genes was received. When slave X returns results prior to slave Y even though the latter was sent the data earlier, this simple bookkeeping process insures that the returned results will fit in the corresponding slot.

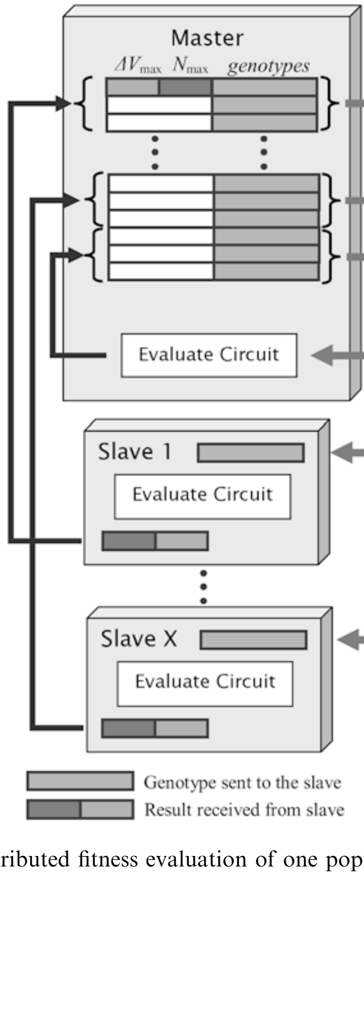

Master
$\Delta V_{\max}$ $N_{\max}$ genotypes
Evaluate Circuit
Slave 1
Evaluate Circuit
Slave X
Evaluate Circuit
Genotype sent to the slave
Result received from slave

Fig. 7 Distributed fitness evaluation of one population

5 Circuit analysis

The methodology of circuit analysis will be illustrated using the circuit shown in Fig. 8. Prior to the turn-on of the chip, the initial conditions on the circuit [i.e., $v_C(0)$, $i_{L1}(0)$, $i_{L2}(0)$] can be determined by simple DC analysis. Linear transient analysis [6] will be used to determine the voltage and current conditions at time $t + \Delta t$ given those at time t. The voltage-current relationship for the capacitor during an interval Δt can be discretized using the trapezoidal rule to give:

$$v_C(t + \Delta t) = v_C(t) + \frac{\Delta t}{2C}[i_C(t) + i_C(t + \Delta t)]. \quad (1)$$

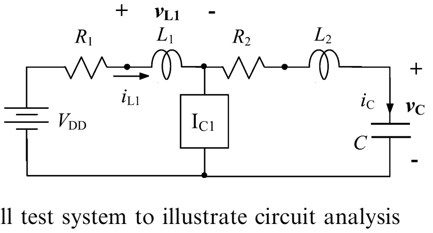


Fig. 8 Small test system to illustrate circuit analysis

The quantities $i_C(t)$ and $v_C(t)$ are known from either a previous solution point or initial conditions. So if we assign $G_{ec} = 2C/\Delta t$ and $I_{ec} = -i_C(t) - (2C/\Delta t)v_C(t)$, then Eq. 1 can be represented by a Norton equivalent circuit (Fig. 9). The model of inductor L_1 (and similarly for inductor L_2) can be obtained from Eq. 1 by the principle of duality:

$$i_{L1}(t+\Delta t) = i_{L1}(t) + \frac{\Delta t}{2L_1}[v_{L1}(t) + v_{L1}(t+\Delta t)]. \tag{2}$$

Here again, the quantities $i_{L1}(t)$ and $v_{L1}(t)$ are known from either a previous solution point or initial conditions. So by assigning $G_{eq} = \Delta t/2L_1$ and $I_{eq} = i_{L1}(t) + (\Delta t/2L_1)v_{L1}(t)$, Eq. 2 can also be represented by a Norton equivalent circuit. By replacing the capacitor and inductors of the circuit of Fig. 8 with their equivalent circuits we obtain a linear transient circuit whose parameters depend on the voltage and current conditions at time t, as shown in Fig. 9.

The resulting circuit can be solved using straightforward DC analysis. Its nodal equations in matrix form can be written as follows:

$$\mathbf{Gv} = \mathbf{I} \tag{3}$$

where:

$$\mathbf{G} = \begin{pmatrix} G_1 + G_{c1} & -G_{c1} & & \\ -G_{e1} & G_2 + G_{c1} & -G_2 & \\ & -G_2 & G_2 + G_{c2} & -G_{c2} \\ & & -G_{c2} & G_{c2} + G_{ec} \end{pmatrix},$$

$$\mathbf{v} = \begin{pmatrix} v_1(t+\Delta t) \\ v_2(t+\Delta t) \\ v_3(t+\Delta t) \\ v_4(t+\Delta t) \end{pmatrix},$$

and

$$\mathbf{I} = \begin{pmatrix} -I_{e1} + G_1 V_{DD} \\ -I_1(t+\Delta t) + I_{e1} \\ -I_{c2} \\ I_{e2} - I_{cc} \end{pmatrix}.$$

Time is divided into an appropriate number of increments all of equal duration Δt. Fixing Δt causes the equivalent conductance representing an inductance or capacitance to be constant for a given network for the duration of the transient analysis. This makes it possible for the conductance matrix $\boldsymbol{G}$ in Eq. 3 to be formed and triangulated using optimal pivot ordering *only once* at the onset of the analysis. At each time-step of the linear transient analysis only the right-hand side of Eq. 3 needs updating, making the transient analysis much faster, which is a critical requirement for an efficient overall solution process. This transient analysis process is presented in the flowchart shown in Fig. 10.

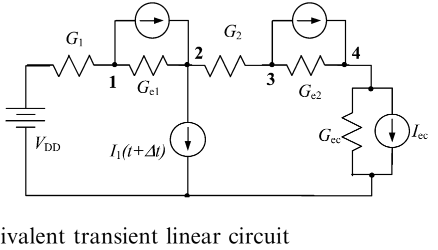


Fig. 9 Equivalent transient linear circuit

6 Results

Four systems were used to test the developed GA-based capacitor placement tool. Systems S_1 and S_2 are subcircuits of the larger system (S_3) shown in Fig. 1, which consists of 46 nodes including ground. System S_2 contains 22 nodes and 8 IC chips with an equal number of capacitors, and system S_1 contains 10 nodes and 4 IC chips. System S_4, shown in Fig. 11, is a large PCB consisting of 64 IC chips.

The validity of the circuit analysis method was established by comparing the solution it produces with those of PSPICE [9]. The circuit simulation for system S_3 was carried out for 10 ns in increments of 0.01 ns. Figure 12 shows the voltage waveforms for some of the nodes of S_3 without any capacitors, which is an identical match with the corresponding ones obtained from PSPICE. Figure 13 shows the plots for the same nodes when capacitors are added at nodes 9, 19, 31, 39, and 43, which again gives a perfect match with results obtained from PSPICE.

The best solution obtained using the GA tool for S_1 is the addition of one capacitor at node 7, at IC3, with an average number of 22 generations. The maximum voltage deviation observed was 0.0486 V at node 6. The best

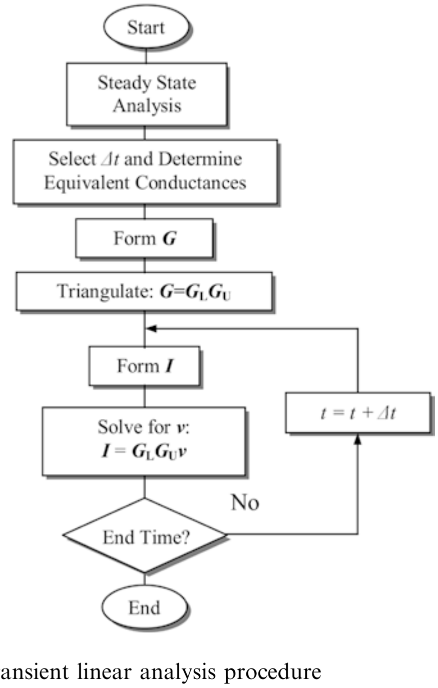

Start
Steady State Analysis
Select Δt and Determine Equivalent Conductances
Form $\boldsymbol{G}$
Triangulate: $\boldsymbol{G}=\boldsymbol{G}_L\boldsymbol{G}_U$
Form $\boldsymbol{I}$
Solve for $\boldsymbol{v}$: $\boldsymbol{I} = \boldsymbol{G}_L\boldsymbol{G}_U\boldsymbol{v}$
$t = t + \Delta t$
End Time?
No
End

Fig. 10 Transient linear analysis procedure

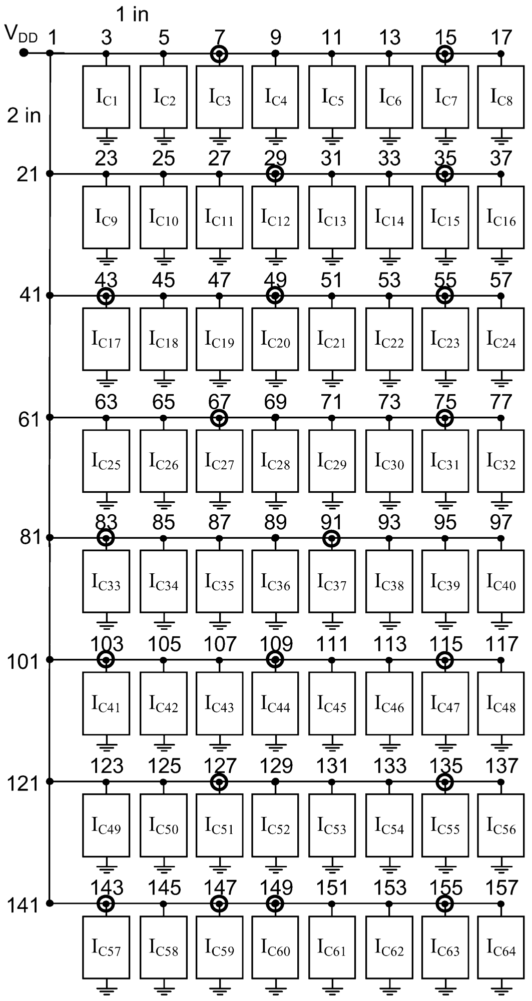

1 in
VDD 1 3 5 7 9 11 13 15 17
2 in
IC1 IC2 IC3 IC4 IC5 IC6 IC7 IC8
21 23 25 27 29 31 33 35 37
IC9 IC10 IC11 IC12 IC13 IC14 IC15 IC16
41 43 45 47 49 51 53 55 57
IC17 IC18 IC19 IC20 IC21 IC22 IC23 IC24
61 63 65 67 69 71 73 75 77
IC25 IC26 IC27 IC28 IC29 IC30 IC31 IC32
81 83 85 87 89 91 93 95 97
IC33 IC34 IC35 IC36 IC37 IC38 IC39 IC40
101 103 105 107 109 111 113 115 117
IC41 IC42 IC43 IC44 IC45 IC46 IC47 IC48
121 123 125 127 129 131 133 135 137
IC49 IC50 IC51 IC52 IC53 IC54 IC55 IC56
141 143 145 147 149 151 153 155 157
IC57 IC58 IC59 IC60 IC61 IC62 IC63 IC64

Fig. 11 A PCB with 64 chips and capacitor locations shown as encircled nodes

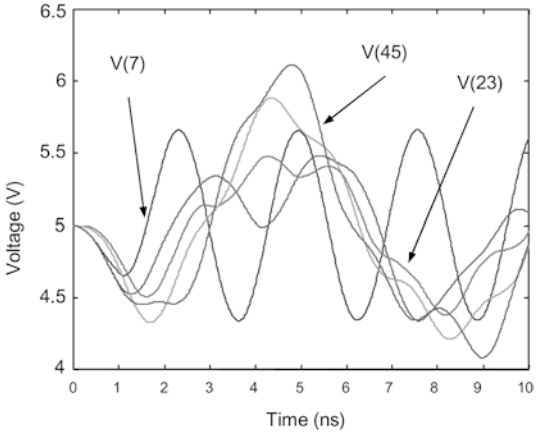


Fig. 12 Voltages at some nodes of S_3 without capacitors

solution for S_2 is the addition of two capacitors at nodes 9 and 21, i.e., at IC4 and IC7, with an average of 43 generations. The maximum voltage deviation observed was 0.123 V at node 16. The best solution for system S_3 had five capacitors at nodes 9, 19, 31, 39, and 43, with a maximum voltage deviation of 0.159 V.

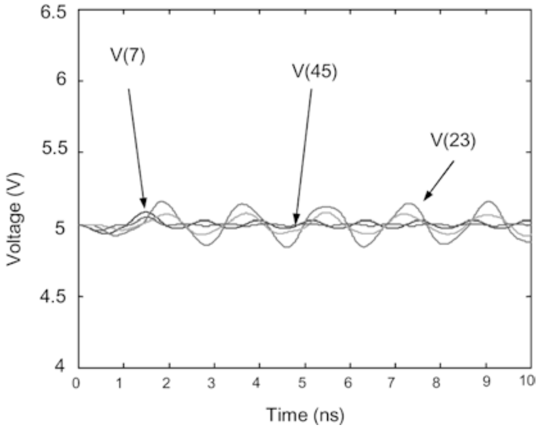


Fig. 13 Voltages at some nodes of S_3 with capacitors at nodes 9, 19, 31, 39, and 43

To show the effectiveness of the GA search, a manual search was carried out to find the first "best" locations for S_3 using a sequential approach. System S_3 was first simulated using PSPICE without any capacitor and the node with the largest voltage deviation is selected to place a capacitor. This process was then repeated with the capacitor added to identify a new location and place another capacitor there, until the observed voltage deviation was within the specified limit of 0.2 V. Using this sequential approach, it was found that six capacitors were needed to reduce the voltage to within 0.2 V. The GA tool, on the other hand, was able to find a solution with five capacitors, instead of six.

The effect of the various GA parameters was investigated: the population size, the probability of mutation, and crossover probability. Table 1 shows the fitness variation in S_3 as the population size changes. Each column reports some summary statistics on the fitness of the best solutions observed after 10 runs. One can clearly observe an improvement in the quality of solutions obtained as the population size increases, which is indicated by an increase in the average of the best solutions and a decrease in the corresponding standard deviation.

An increase in the mutation probability from 0.01 to 0.05 increased the likelihood of obtaining the best solution at each run of the simulations but caused the population to have a lower average fitness. In other words the best solution is more likely to emerge from a population with a high variety of individuals. The results

Table 1 Variation of fitness with population size for S_3

Fitness	Population size			
	5	10	15	20
Best	0.7514	0.7603	0.7603	0.7603
Worst	0.5378	0.6709	0.7013	0.7013
Average	0.6513	0.7266	0.7279	0.7371
σ	0.0613	0.0296	0.0230	0.0235

Table 2 Results of 10 runs on system S_3

No.	Best fitness	ΔV (V)	Best chromosome	Average fitness
1	0.7574	0.1703	0001001010100010	0.4518
2	0.7603	0.1589	0010001000101010	0.5483
3	0.7603	0.1589	0001001000101010	0.5238
4	0.7603	0.1589	0010001000101010	0.4843
5	0.7603	0.1589	0010001000101010	0.4858
6	0.7259	0.0963	0010001010011001	0.4879
7	0.7603	0.1589	0010001000101010	0.3915
8	0.7514	0.1944	0010001000100101	0.5462
9	0.7603	0.1589	0001001000101010	0.4736
10	0.7603	0.1589	0001001000101010	0.5779

Population size: 30
Number of generations: 50
Mutation probability: 0.05

from 10 simulation runs on system S_3 are shown in Table 2, obtained for the conditions noted below the table. The results show that in 90% of the cases a superior solution with five capacitors is obtained. (The number of capacitors is the number of 1s in the chromosome.)

The distributed computing was carried out on a Sun Ultra 30 server (the master) and Ultra 10 workstations (the slaves) running the Solaris 2.6 operating system. The CPU speed and memory of each machine were 300 MHz and 256 MB, respectively. Table 3 shows several runs carried out on system S_4 (shown in Fig. 11) with a varying number of slave processors.

Figure 14 shows the speed-up ratio as the number of slave processors increases from 0 to 7 for systems S_3 and S_4. Clearly there are important benefits of using the simple distributed computing process. However, the saving in computation time is initially linear and then tends to saturate as the number of slave processors increases. The saturation is essentially due to some form of diminished returns as the number of processors increases, which is enhanced by the fact that the serial part of the calculation becomes more significant as the load on each processor is reduced. The saturation is more evident in the case of system S_3 since the six slave processors and the master would be sharing the evaluation of a relatively small population size of 30. In this case, the load of each processor is obtained from the result of the integer division of 30/7 with the remainder being allocated among the slave processors until it is exhausted. So the load of the master and slaves would be 4, 5, 5, 4, 4, 4, and 4. In the case of 7 slave processors, the load of the master and slaves would be 3, 4, 4, 4, 4, 4, 4, and 3. The time required for the master processor to solve system S_3 under the conditions given in Table 2 is 1,315 s.

Table 3 Eight runs with a varying number of slave processors for system S_4

Number of slaves	Average fitness	Best fitness	ΔV (V)	Num caps	Time (s)
0	0.4555	0.6802	0.1163	26	101,934
1	0.4572	0.7277	0.1575	22	64,559
2	0.5030	0.7302	0.1163	22	44,020
3	0.5203	0.7275	0.1596	22	34,481
4	0.4734	0.7302	0.1163	22	27,653
5	0.4907	0.7288	0.1397	22	24,739
6	0.5275	0.7380	0.1928	21	20,598
7	0.5126	0.7302	0.1163	22	20,014

Population size: 120
Number of generations: 200
Mutation probability: 0.05

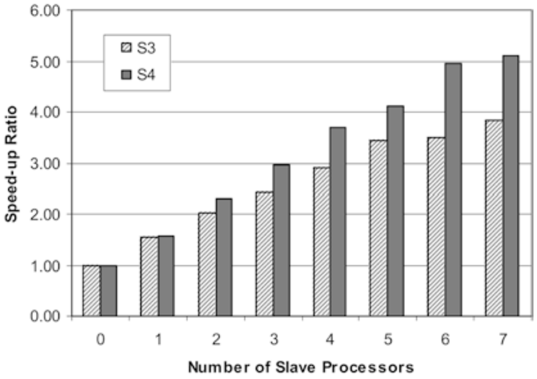


Fig. 14 Speed-up ratio versus number of slave processors for systems S_3 and S_4

7 Conclusion

In this work we have investigated capacitor placement on a printed circuit board to reduce the effect of SSN as a GA search problem. The solution process makes use of distributed computing resources available on a LAN in order to solve large problems efficiently. The objective used in the formulation was to reduce the cost of added capacitors, while keeping the maximum voltage deviation within some specified noise margin. The presence of capacitors at the selected positions was represented by a stream of zeros and ones, which is interpreted as a genotype and manipulated using GA operators to systematically approach an optimal solution. The fitness of a genotype was defined as a decreasing function of the cost of capacitors needed and the voltage deviation obtained using linear transient circuit analysis. In addition to the main problem definition and formulation this work tackled the issue of establishing some guidelines to set the various parameters of the GA. Through several simulation runs on various systems, it was established that the GA requires a population size equal to about double the size of the genotype and a number of generations increasing in some polynomial form. For large systems, it was possible to achieve a speed-up ratio of about five with seven slave processors by dividing the fitness calculations among several processors available on a LAN according to a simple distributed computing algorithm.

References

1. Chen Y, Chen Z, Fang J (1996) Optimum placement of decoupling capacitors on packages and printed circuit boards under the guidance of electromagnetic field simulation. In: Proceedings of the 46th electronic components and technology conference, Orlando, FL, pp 756–760
2. Chipperfield A (1997) Introduction to genetic algorithms. In: Zalzala A, Fleming PJ (eds) Genetic algorithms in engineering systems. IEE, Bath, pp 6–18
3. Drewniak JF, Knighten JL, Smith JL, Orlandi NW, Van Doren A, Hubing TP, DuBroff TH (2001) Quantifying smt decoupling capacitor placement in dc power-bus design for multilayer pcbs. IEEE Trans Electromagn Compat 4:588–599
4. Iba K (1994) Reactive power optimization by genetic algorithms. IEEE Trans Power Syst 2:685–691
5. Karaki SH, Kayssi AI, Abdouni B, Raad M-C (2002) Capacitor placement using genetic algorithms to reduce switching noise. In: Proceedings of the 14th international conference on microelectronics, Beirut, Lebanon, December 2002, pp 242–246
6. Pillage LT, Rohrer RA, Viswesvariah C (1995) Electronic circuit and system simulation methods. McGraw-Hill, New York
7. Smith LD, Anderson RE, Forehand DW, Pelc TJ, Roy T (1999) Power distribution system design methodology and capacitor selection for modern cmos technology. IEEE Trans Adv Packag (3):284–291
8. Sundhararajan S, Pahwa A (1994) Optimal selection of capacitors for radial distribution systems using a genetic algorithm. IEEE Trans Power Syst 3:1499–1505
9. Tuinenga P (1995) SPICE: a guide to circuit simulation and analysis using PSPICE. Prentice-Hall
10. Yook JG, Chandramouli V, Katehi LPB, Sakallah KA, Arabi TR, Schreyer TA (1997) Computation of switching noise in printed circuit boards. IEEE Trans Components Packag Manuf Technol A 1:64–75