Simultaneous Shield Insertion and Net Ordering for Capacitive and Inductive Coupling Minimization

Lei He
University of Wisconsin
1415 Engineering Drive
Madison, WI 53706
(608) 262-3736
lhe@ece.wisc.edu

Kevin M. Lepak
University of Wisconsin
1415 Engineering Drive
Madison, WI 53706
(608) 265-3810
lepak@ece.wisc.edu

ABSTRACT

In this paper, we first show that existing net ordering formulations to minimize noise are no longer valid with presence of inductive noise, and shield insertion is needed to minimize inductive noise. We then formulate two simultaneous shield insertion and net ordering (SINO) problems: the optimal SINO/NF problem to find a min-area SINO solution that is free of capacitive and inductive noise, and the optimal SINO/NB problem to find a min-area SINO solution that is free of capacitive noise and is under the given inductive noise bound. We reveal that both optimal SINO problems are NP-hard, and propose effective approximate algorithms for the two problems. Experiments show that our SINO/NB algorithm uses from 15% to 57% fewer shield wires when compared to separated net ordering and shield insertion procedure. Furthermore, under practical noise bounds, the SINO/NB solutions use from 44% to 67% fewer shield wires when compared to SINO/NF solutions, and use 10% to 40% fewer shield wires when compared to the theoretical lower bound for optimal SINO/NF solutions. Additionally, all our algorithms are extremely efficient to finish all examples in a few seconds. To the best of our knowledge, it is the first work that presents an in-depth study on the simultaneous shield insertion and net ordering problem to minimize both capacitive and inductive noise.

Keywords

VLSI design automation, noise minimization, shielding, net ordering.

1. INTRODUCTION

In deep submicron (DSM) designs, the wire thickness is often larger than the wire width, and the spacing between adjacent wires is often smaller than the distance between adjacent metal layers. This makes the coupling capacitance (herein referred to as Cx) between adjacent wires on the same layer larger than the ground capacitance (the sum of area and fringe capacitance), and in turn makes the coupling noise between adjacent wires a concern for DSM designs. Because the coupling capacitance between non-adjacent wires is negligible, we may permute the net ordering (or track assignment) so that sensitive nets[1] are not adjacent in order to reduce the coupling noise. The net ordering (track assignment) problem has been studied in [1-4] under the assumption that coupling is determined only by directly adjacent nets. However, this assumption is no longer true if we consider coupling inductance. This has been illustrated by the experiment in [6], where the coupling of a 18-bit bus is computed by SPICE simulations using the interconnect RLC model. It was assumed that all signal wires in the bus are switching simultaneously, except that the two central wires are quiet victims. As shown in Table 1, when there are no shielding wires (in short, shields), the noise in the quiet victims is 0.71V. As we insert one shield for every six signal wires (the row of two shields in Table 1), the noise is drastically reduced to 0.38V. Note that the two shields do not change the Cx coupling for the victims, therefore the noise reduction is due to reducing the inductive coupling (i.e., Lx coupling) that depends on both adjacent and non-adjacent nets by shield insertion. However, the shield insertion approach in [6] assumed that all nets are sensitive to one-another and hence did not exploit the design freedom of net ordering. Due to the aforementioned nature of Cx and Lx coupling, it is best to consider the sensitivity of nets, and carry out simultaneous shield insertion and net ordering (SINO).

# of Shields	Noise (% of Vdd)	K_i of victims
0	0.71V (55%)	10.08
2	0.38V (29%)	2.98
5	0.17V (13%)	1.17

Table 1. SPICE-computed noise for victims in [6]. Column 3 shows coupling coefficients to be defined in Section 2.1.2.

The rest of this paper is organized as follows: Section 2 introduces the inductive coupling model and formulates the SINO/NF and SINO/NB problems. Section 3 presents properties and algorithms for the two SINO problems. Section 4 details the experimental setting and compares results obtained by different problem formulations and algorithms. Section 5 concludes the paper.

1 See Section 2.1.2 for the definition of net sensitivity

2. PROBLEM FORMULATIONS

2.1 Preliminaries

2.1.1 Coplanar Interconnect Structures

Throughout this work, we consider only parallel coplanar interconnect structures with all wires having the same length. These are characterized as a number of signal traces and power/ground traces which run parallel in the same layer. We give an example of this structure in Figure 1. In the Figure, P and G represent the power and ground grids (*P/G grids*), s represents signal wires (denoted as *s-wires*), and g is a shield wire that often has similar width as an s-wire and is connected to P/G grids. Both P/G grids and shield wires provide dedicated current return paths for signals, and are denoted as *g-wires* in this paper. We use the terms "wire" and "net" interchangeably.

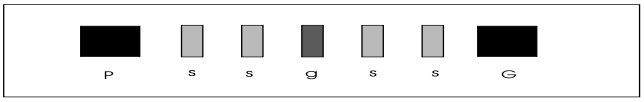

Figure 1. A cross-section view of a coplanar interconnect structure with a shield inserted.

An interconnect structure can be represented by a string, where each symbol stands for an s-wire or a g-wire. For example, the interconnect structure in Figure 1 can be represented by *gssgssg* if we do not distinguish these s-wires. If we label the s-wires from left to right as s_1, s_2, s_3 and s_4, then the string $gs_1s_2gs_3s_4g$ is a unique representation of a net ordering and shield insertion solution (referred to as a SINO solution or a SINO string). In this paper, a SINO string implicitly includes a shield trace (the power and ground grids) as its first and last element. These P/G grids are shield resources, but are not considered in solution size computations as they are present generally, with or without noise considerations. The "size" of a SINO solution can be determined directly from the length of the SINO string. As an example, consider the following: $<g>s_1s_3gs_2s_4s_5gs_7s_6s_0<g>$. This string represents an eight (signal) bit interconnect structure with two g-wires (plus two implicit g-wires for the P/G grids denoted as <g>).

Note that we can apply our formulations and algorithms to be presented to any group of wires which may contain pre-routed g-wires more than just a pair of P/G grids. We call the group of wires sandwiched between adjacent g-wires a *block*, and the number of s-wires in a block as the block size. A block can be represented as a substring of a SINO string (i.e. block 0 of the above string would be written as s_1s_3). As in the original SINO string, the g-wires on each end are implicit with the substring.

2.1.2 Sensitivity and Inductive Coupling Model

We define two nets s_1 and s_2 to be sensitive to each other if a switching signal on s_1 will cause s_2 to malfunction (due to extraordinary crosstalk or delay variation) or vice-versa. The sensitivity for all s-wires in a given problem can be represented compactly with a sensitivity matrix S of size *nxn* (where n is the number of s-wires). An entry of {1,0} in location (i,j) indicates that s_i and s_j are sensitive or not sensitive, respectively, to one another. By definition, the matrix must be symmetric (i.e. $S_{ij} = S_{ji}$). For all formulations, we assume that an appropriate sensitivity matrix is given *a prior*i.

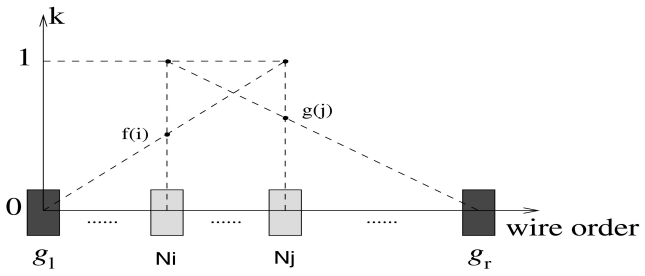

Figure 2. Illustration of K_{eff} computation.

In order to accurately model inductive coupling between two s-wires, we use the "effective coupling model", i.e., K_{eff} model developed in [5], with the notation illustrated in Figure 2. In this model, when nets *i* and *j* are in different blocks, the coupling coefficient is $K_{ij} = 0$. When the two nets are in the same block, the coupling coefficient is $K_{i,j} = (f(i)+g(j))/2$, where f(i) and g(j) are two functions defined as follows: let g_l and g_r be the track ordering for the g-wires at ends of the block, and N_i and N_j be track ordering for nets i and j. We assume that the coupling is 0 at g_l, and is 1 at N_j, then f(i) is the linear interpolation, i.e., $f(i) = (N_i-g_l)/(N_j-g_l)$. Similarily, we my compute $g(j) = (g_r-N_j)/(g_r-N_i)$. It is shown that this model is accurate within 15% of numerical extraction and is conservative in most cases [5]. As indicated in Table 1, it is further observed that the higher the Keff, the larger the noise. Note that we only consider insertion of minimum width shields because it is shown in [6] that, in general, using additional shields is more effective than increasing the shield wire width. Based on the K_{eff} model, if s-wire s_i is a net sensitive to s-wire s_j in the same block, we may compute the total amount of Lx coupling for s_i as: $K_i = \sum_j K_{ij}$

2.2 Optimal SINO Problems

We say that an s-wire s_i is capacitive noise free if it is not directly adjacent to any other s-wire s_j that is sensitive to it. Similarly, we say that s_i is inductive noise free if it does not share a block with any other sensitive wire s_j. We say a placement P (or equivalently, a SINO solution, or a SINO string) is noise free if, and only if, all nets s_i within P are free of both capacitive noise and inductive noise. With respect to these concepts, we define the following SINO problem to eliminate Cx and Lx noise and call it the noise free SINO problem (SINO/NF).

Formulation 1 (Optimal SINO/NF problem): For a given placement P, find a new placement P' by simultaneous shield insertion and net re-ordering such that P' is noise free and the total area of P' is minimal.

In general, the SINO/NF problem is over-constrained and may lead to over-designed solutions as shown in Section 4. To address more realistic design constraints, we define the following SINO problem to meet a given noise bound and call it the noise bounded SINO problem (SINO/NB).

Formulation 2 (Optimal SINO/NB problem): For a given placement P, find a placement P' with the minimum area by simultaneous shield insertion and net re-ordering such that any s_i in P' is free of capacitive noise and its inductive coupling to all sensitive wires s_j is less than a given value.

In this paper, we use the K_{eff} model as the figure of merit for inductive coupling, i.e., we solve the following optimal SINO/NB-K_{eff} problem: For a given placement P, find a new

placement P' by simultaneous shield insertion and net re-ordering such that P' is free of capacitive coupling and the inductive coupling K_i satisfies $K_i <= k_i$ for all K_i where k_i is given *a priori* and is a measure of inductive noise that can be tolerated in an s-wire to maintain correct operation. Throughout the rest of this paper, we use a uniform value of k_i denoted as K_{thresh} (a noise threshold for K). However, our formulation, algorithms, and implementation are all able to handle non-uniform k_i.

We attempt to solve both the SINO/NF problem and SINO/NB-K_{eff} problem in Section 3. For simplicity of presentation, we use SINO/NB as shorthand for SINO/NB-K_{eff} throughout the remainder of the paper. Note that our formulation and algorithms to be presented are applicable to inductive noise models more accurate than the K_{eff} model.

3. PROPERTIES AND ALGORITHMS

3.1 Properties for SINO/NF and SINO/NB Problems

Theorem 1: The Optimal SINO/NF problem is NP-hard.

Proof: Please see [12] for details of the proof.

Theorem 2: The optimal SINO/NB problem is NP-hard.

Proof: Please see [12] for details of the proof.

Given that the SINO/NF and SINO/NB problems are NP-hard, we will focus on developing heuristic/approximate algorithms to solve the problems with satisfactory results. Before we present these algorithms, we first introduce the concept of *sensitivity graph*. We can build a sensitivity graph such that a node corresponds to an s-wire and an edge exists between two nodes if and only if the correspondent s-wires are sensitive to one another. We then propose the following lower bound for the optimal SINO/NF solution:

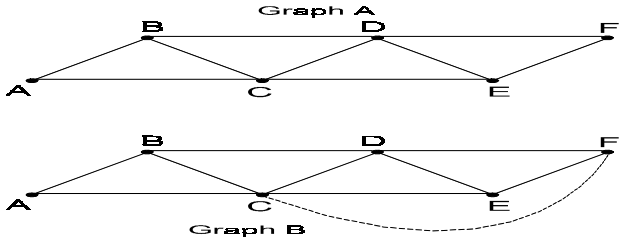

Figure 3. Sensitivity graphs illustrate max. clique size as a lower bound.

Theorem 3: The maximum clique size in the sensitivity graph is a lower bound of the number of blocks required in all optimal SINO/NF solutions.

The theorem follows directly from the formulation of the sensitivity graph and the definition of the maximum clique in a graph [7]. To illustrate this conclusion and further show that the maximum clique size is not an upper bound for the total number of blocks in optimal SINO/NF solutions, we present two examples in Figure 3. For sensitivity graph A, it is easy to verify that the graph has a maximum clique size of three. An optimal SINO/NF solution is ADgEBgFC. We can see that there are three blocks in the solution. Sensitivity graph B is nearly the same as A. The reader may easily verify that indeed this graph still only has a maximum clique size of three, but it is not possible to find a SINO/NF solution for this graph with three blocks. One valid solution is ADgEBgCgF.

The lower bound presented can be used to judge the quality of approximate algorithms to be presented in the next subsection. Comparisons between the lower bound and the number of shield wires used by these algorithms will be presented in Section 4.

3.2 Algorithms for Solving SINO/NB Problems

We develop four approximate algorithms for solving the SINO/NB problem: greedy-based shield insertion (*SI)* algorithm, net ordering for minimizing Cx noise followed by SI algorithm (*NO+SI* algorithm), graph-coloring based SINO algorithm (*SINO/GC* algorithm), and simulated-annealing based SINO algorithm (SINO/SA algorithm). We solve the SINO/NF problem by using the SINO algorithms and setting the noise bound to zero. We will compare different problem formulations and algorithms in Section 4. For simplicity of presentation, we explain NO+SI, SINO/GC and SINO/SA algorithms in the context of Lx coupling only--we will show at the end of this section that in general, solving the SINO problems formulated in this way makes Cx solutions very simple to find. We will also illustrate this by experimental results in Section 4. In order to more succinctly describe the algorithms and not clutter them with trivial details, we also define the following operations and quantities:

Insert_Shield(a): Given a placement P of size m, move all wires (s-wires and g-wires) at locations *a, a+1,..., m-1* to locations *a+1, a+2,..., m* in the placement, creating a new placement P'. Then, insert a g-wire at location *a* in P'. Finally set P=P'.

Compute_Coupling(s_i): Given a placement P of size *m*, for each s_k != s_i in the current block in P, if s_i is sensitive to s_k compute the K_{eff} between s_i and s_k. Sum the K_{eff} over all s_k.

Compute_Block_Coupling(s_i): For each s_i in the current block, find the maximal K_{eff} computed by Compute_Coupling(s_i) and return it (the maximal K_i for any s-wire within the block).

Max_Clique(S): Compute the max. clique in sensitivity graph S.

Compute_Placement_Cost: Compute the cost for a placement. Details of this are given in Section 3.2.3.1.

With these definitions in place, we can succinctly describe our SINO/NB algorithms. The *SI* and *SINO/GC* algorithms can be described easily and intuitively. *SINO/SA* is slightly more complicated, hence we assume that the reader is familiar with SA (for a discussion of SA in other VLSI design contexts, see [8,9]).

3.2.1 Greedy Based Shield Insertion Algorithm

The complete text of the greedy-based shield insertion (*SI)* algorithm is given in [12] due to its relative simplicity. The essence of the algorithm is the following: Run through the given placement P. If we encounter two adjacent sensitive nets, insert a shield wire between them. Also, at each location in the placement, calculate the maximum value of K_i that would exist in the current block if we allowed net s_i to become a member. If K_i is greater than K_{thresh}, then create a new block.

Because one shield wire is needed for every pair of adjacent sensitive wires, the solution given by the SI algorithm depends on the initial placement. Obviously, the number of shield wires can be reduced by first running existing net ordering algorithms to re-

order nets so that no sensitive nets are adjacent to each other, then invoking the SI algorithm. This leads to the *NO+SI* algorithm.

3.2.2 Graph-Coloring Based SINO Algorithm

```
Graph Coloring (GC) Algorithm:
Given an initial sensitivity graph S of signal nets:
MC = Max_Clique(S)
P = new placement with MC blocks
for each s-wire s_i
    for each block b in P (sorted from largest to smallest)
        if Compute_Block_Coupling(s_i) == 0
            Insert s_i into b
            next s_i
    endfor
    for each block b in P
        if Compute_Block_Coupling(s_i) < K_thresh
            Insert s_i into b
            next s_i
    endfor
    Insert_Shield(end_of_placement)
    Insert s_i into the new block
endfor
```

Figure 4. Graph-Coloring SINO Algorithm (SINO/GC)

In Figure 4, we present the graph-coloring based SINO (*SINO/GC)* algorithm. This algorithm attempts to approximate a solution to the weighted graph coloring problem by using the maximum clique as a starting point. It works in the following way: First, determine the maximum clique in the sensitivity graph S. We saw in Section 2 that this is a lower bound for the number of shields required in the SINO/NF problem. Let *m* be the maximum clique size. We first create a placement with *m* blocks. We then take each net s_i and try to place it into a block with no other sensitive wires. If we cannot do this, we insert s_i into a block where inserting s_i will cause the least amount of noise to be added but without causing a noise violation. If we cannot place s_i without creating a K_{thresh} violation, simply create a new block and place s_i into it.

Note that SINO/NF problem can be mapped into the graph-coloring problem as outlined in the proof of Theorem 3, and then be solved using the graph coloring algorithms given in [10,11].

3.2.3 Simulated Annealing Based SINO Algorithm

In Figure 5, we present the simulated-annealing based SINO (*SINO/SA)* algorithm. We give details of our SINO/SA algorithm in the following subsections:

3.2.3.1 Cost Function

Compute_Cost(P) computes the cost for a placement P. The cost is the weighted sum of the following components: (i) *Cap_violation*: total number of nets that are adjacent to their sensitive nets in P; (ii) *Area*: total number of g-wires present in P; (iii) *Ind_Noise*: total number of $K_i > K_{thresh}$ violations in P; and (iv) *Inductance Violation Figure*. It is computed for a placement as shown in Figure 6. The purpose of the *Inductance Violation Figure* is to penalize a placement for the magnitude of K_{thresh} violations. Its usage (as opposed to simply forbidding placements P' that have K_{thresh} violations) allows the algorithm to potentially trade inductive noise violations for smaller overall placement size depending on the result desired, and can be useful in different SINO formulations.

The weighting factor for each cost component can be tuned for different design objectives. In this paper our stated goal is to minimize placement size without violating K_{thresh} noise constraints, hence weighting factors were chosen to help us achieve those goals with maximal efficiency.

```
Simulated Annealing Algorithm: Given a placement P:
Repeat
    Temp = Initial_Temperature;
    Cost = Compute Placement Cost(P);
    Repeat
        Random_Move(P, P');
        Candidate_Cost = Compute_Cost(P');
        ds = Candidate_Cost - Compute_Cost(P);
        if (ds < 0)
            P = P';
        else
            r = RANDOM(0,1);
            if (r < exp(-ds/Temp))
                P=P';
    Until equilibrium at Temp is reached;
    Temp=Temp*Temperature_Adjustment;
    /*(0 < T_A < 1)*/
Until Temp == Freezing_Point;
```

Figure 5. Simulated Annealing SINO Algorithm (SINO/SA)

```
Total_Violation_Figure = 0;
for each s_i in placement P
    if Compute_Coupling(s_i) > K_thresh
        Total_Violation_Figure += (1+K_eff-K_thresh)^3
end
```

Figure 6. Computation of the Inductance_Violation_Figure

3.2.3.2 Random Moves

Random_Move(P, P') performs one of the following changes to placement P to generate a new placement P': (i) Combine two random blocks in P, (ii) Swap two random s-wires in P, (iii) Move a single random s-wire to a new and random location, (iv) Insert a g-wire at a random location in P.

It is worthwhile to note that combining two random blocks in a placement P is also equivalent to removing a g-wire if the two blocks are adjacent. Moves which create two-adjacent g-wires in a placement are categorically rejected and a new move is tried.

3.2.3.3 Temperature Adjustment and Stopping Criterion

The method of temperature adjustment is shown in Figure 5. We use a simple multiplicative constant of the current temperature. At each temperature step, the variance of the current placement cost from its previous value is taken and averaged over several random moves to determine the stability of the system at each temperature. When the variance is less than a set threshold, we move to the next temperature step. The starting temperature, freezing point, temperature adjustment, and variance threshold factors were all determined experimentally.

3.2.4 Cx Considerations

As stated previously, all of the algorithms given above assumed only Lx coupling (with the exception of greedy-based shield insertion (*SI) algorithm*). However, we've been asserting throughout this paper that we are simultaneously minimizing both

Cx and Lx noise. How is this so? The answer is somewhat intuitive.

Given that Lx is a "long-range" effect and Cx is a "short-range" effect, we expect a solution to Lx noise to be a naturally good starting solution for minimizing Cx noise, because SINO solutions for inductance already tend to distance sensitive nets from one another. For a specific example, consider the SINO/NF problem. Any solution that considers only Lx noise must put sensitive nets in different blocks, therefore these nets are not adjacent to each other and there is no Cx coupling. If we consider SINO/NB problems, we will always try to move Cx coupled nets far away from one another (to satisfy the $K_i < K_{thresh}$ constraint), hence making any Cx violations easy to solve in general by a trivial reordering of the nets within a block (this step is implicit as a "clean up" step in the coding of the SINO algorithms and is omitted above for clarity. In our SINO/SA, a weight is also specifically given to minimize capacitive effects). Even though our algorithms are able to carry out net ordering to eliminate Cx noise during every step of shield insertion and s-wire assignment (to blocks), we observed in our experiments that the following two-phase approach works very well: we first perform the shield insertion and s-wire assignment without considering Cx and net ordering, then reorder nets within each block with consideration of both Lx and Cx noise. The experiment results presented in the next section are produced using this two-phase approach. There is no solution with Cx noise.

4. EXPERIMENTAL RESULTS

	SINO/NF	SINO/NB			
K_{thresh}	GC	SI	NO+SI	GC	SA
Net Sensitivity Rate: 10%					
0.5	3.2 (2.0)	6.8	4.6	2.6	2.0
1.0		5.0	2.8	2.0	1.8
1.5		4.4	2.0	2.0	1.0
2.0		4.2	1.2	2.0	1.0
Net Sensitivity Rate: 30%					
0.5	6.0 (3.8)	13.2	6.0	4.4	3.8
1.0		13.2	4.4	4.2	3.0
1.5		13.2	3.2	3.8	2.6
2.0		13.2	2.8	3.8	2.0
Net Sensitivity Rate: 60%					
0.5	13.6 (8.2)	22.6	7.0	8.2	7.0
1.0		22.4	5.4	8.2	5.0
1.5		22.4	4.2	8.2	4.0
2.0		22.4	4.0	8.2	3.4

Table 2. Number of shields inserted by SINO algorithms. The numbers in parenthesis are lower bounds on the number of shields required in SINO/NF solutions.

We have implemented all algorithms in the C programming language, and have tested our implementations using a large number of examples. In this section, we first compare results obtained by different approximate algorithms to the SINO/NB formulation, and then compare results given by two formulations (SINO/NF versus SINO/NB). We also report the running time for different formulations and algorithms.

We use a coplanar interconnect structure containing 32 s-wires as an example to determine the performance of the algorithms for different combinations of K_{thresh} and *sensitivity rate*. We consider the following values for K_{thresh}: 0.5, 1.0, 1.5 and 2.0. When K_{thresh} =0.5, the total coupling coefficients for a net is less than 0.5 in the target SINO solution. We iterate the following sensitivity rates: 10%, 30% and 60%. When the sensitivity rate is 10%, each net is sensitive to 10% of all nets, and these sensitive nets are picked randomly for the given s-wire. The number in parenthesis is the SINO/NF lower bound on the number of shields from Section 3.1.

For each combination of K_{thresh} and sensitivity rate (see Table 2), we report the resulting number of shields for different formulations and algorithms. To make the comparisons "fair" among the different algorithms, we run each algorithm on the same initial placement for five different random placements/sensitivities. The average of these five runs is shown in Table 2. Note that there is no entry in the tables for Cx noise because there was not any in all cases. Finally, it is worthwhile to point out that we did not tune our SINO/SA algorithm for different examples.

4.1 Comparison Between Different SINO/NB Algorithms

We compare the following approximate SINO/NB solutions given by the following algorithms: greedy- based shield insertion (*SI*), net ordering followed by SI (*NO+SI*), graph-coloring based SINO (*SINO/GC*), and simulated-annealing based SINO (*SINO/SA*). One may easily see from Table 2: First, SI is always significantly worse than all of the other solutions in terms of the number of shields inserted. In the worst case, SI yields a result about 550% worse than SA in terms of the number of shields inserted (see sensitivity rate of 60% and K_{thresh} of 2.0). Furthermore, performing net ordering before SI, i.e., NO+SI significantly outperforms SI only. On the other hand, at lower sensitivity rates (10% and 30%) and low K_{thresh} values, SINO/GC performs significantly better than NO+SI in terms of shields inserted (ranging from 77% better to 5% better). As we move to higher sensitivity rate (60%) and K_{thresh}, we see that NO+SI begins to overtake SINO/GC, and can approach the performance of SINO/SA. For example, at sensitivity rate of 60% and K_{thresh} = 1.0, NO+SI performs 35% better than SINO/GC and 8% worse than SINO/SA.

As we expect, SINO/SA always performs the best for any given setting. In terms of the number of shield wires, compared to NO+SI, its improvement increases as sensitivity rates and K_{thresh} decrease. It is 15% better at sensitivity rate of 60% and K_{thresh} = 1.0, and is 57% better at sensitivity rate of 10% and K_{thresh} = 0.5. Therefore, it is important to consider simultaneous net ordering and shield insertion, rather than separated net ordering and shield insertion. The improvement of SINO/SA over SINO/GC does not depend much on the sensitivity rate, but increases when K_{thresh} goes up. At sensitivity rate of 60%, it is 15% better for K_{thresh} = 0.5, and is 58% better for K_{thresh} = 2.0.

4.2 Comparison Between SINO/NB and SINO/NF Formulations

We first compare the SINO/NF and SINO/NB solutions. We base our comparison on the results produced by the best SINO/NB algorithm (SINO/SA) and the GC algorithm for the SINO/NF formulation. For the smallest K_{thresh} value of 0.5 (in order to make SINO/NB most closely approximate SINO/NF), compared to the SINO/NF formulation, the SINO/NB formulation uses about 35% fewer shield wires for sensitivity rates of 10% and 30%, and uses 48% fewer shield wires for the sensitivity rate of 60%. As shown in Table 1, when K_{thresh} is 1.27 and with Cx noise, the total noise computed by SPICE is 13% of the supply voltage (see details in [6]). Our SINO solution under K_{thresh} = 1.0 and without Cx noise will surely meet the similar SPICE-computed noise bound. In this case, the saving in terms of shield wires introduced by using SINO/NB formulation rather than SINO/NF formulation will be from 44% to 63%. Because our GC algorithm provides an approximate to the SINO/NF formulation, we computed the average lower bound of shield wires via average maximum clique size (based on Theorem 3), and present these lower bounds between parentheses in the column for GC algorithm, SINO/NF formulation. Compared to these lower bounds, the SINO/NB solutions still use from 10% to 40% fewer shield wires.

	SINO/NF	SINO/NB			
	GC	SI	NO+SI	GC	SA
Runtime	0.1sec	0.1sec	0.1sec	0.3sec	1.5sec

Table 3. Approximate run times for SINO algorithms with sensitivity rate of 30%.

Finally, we report runtime in Table 3, where the times are for a single run for a single interconnect structure. Note that the running time for NO+SI algorithm does not include the time for net ordering—we simply applied the SI algorithm to initial placements that are free of Cx noise. As verified in our experiments, whether the initial placement is free of Cx noise does not affect the quality of solution given by SINO/GC and SINO/SA algorithms. The machine used to collect running times has a 450MHz Intel Pentium II processor. All algorithms finished the test examples in a few seconds. Therefore, large interconnect structures can be solved easily by formulations and algorithms we have proposed.

5. CONCLUSIONS AND DISCUSSIONS

We have shown that existing net ordering formulations to minimize noise are no longer valid with the consideration of inductive noise, and shield insertion is needed to minimize inductive noise. We have formulated two simultaneous shield insertion and net ordering (SINO) problems: the optimal SINO/NF problem to find a min-area SINO solution that is free of capacitive and inductive noise, and the optimal SINO/NB problem to find a min-area SINO solution that is free of capacitive noise and is under the given inductive noise bound. We have revealed that both optimal SINO problems are NP-hard, and have proposed effective approximate algorithms for the two problems. Experiments show that our best SINO/NB algorithm uses from 15% to 57% fewer shield wires when compared to separated net ordering and shield insertion. Furthermore, under practical noise bounds, the SINO/NB solutions use from 44% to 63% fewer shield wires compared to SINO/NF solutions and use from 10% to 40% fewer shield wires compared to the theoretical lower bound for optimal SINO/NF solutions. Additionally, all of our algorithms are efficient and finish all examples in a few seconds. To the best of our knowledge, this is the first work that presents an in-depth study on the simultaneous shield insertion and net ordering problem to minimize both capacitive and inductive noise.

6. REFERENCES

[1] T. Gao and C. L. Liu, "Minimum Crosstalk Channel Routing", *Proc. ICCAD*, pp.692-696, 1993.

[2] T. Gao and C. L. Liu, "Minimum Crosstalk Switchbox Routing", *Proc. ICCAD*, pp. 610-615, 1994.

[3] T. Xue and E. S. Kuh, "Post global routing corsstalk synthesis", *IEEE Trans. CAD-16*, no.12, pp. 1418-1430, Dec. 1997

[4] J. S. Yim and C. M. Kyung. "Reducing Cross-Coupling among Interconnect Wires in Deep-Submicron Datapath Design." In *Proceedings of DAC 99*, June 1999.

[5] L. He and M. Xu, "Characteristics and Modeling for On-chip Inductive Coupling", U. of Wisconsin at Madison, *Technical Report ECE-00-1*.

[6] L. He, N. Chang, S. Lin, and O. S. Nakagawa, "An Efficient Inductance Modeling for On-Chip Interconnects", Proc. IEEE Custom Integrated Circuits Conference, pp. 457-460, May 1999.

[7] M. R. Garey and D. S. Johnson, "Computers and Intractability: A guide to the Theory of NP-Completeness", W.H. Freeman and Co. 1991.

[8] C. Sechen, "An improved simulated annealing algorithm for row-based placement", *Proc. ICCAD, pp. 478-481, 1997.*

[9] N. Sherwani, "Algorithms for VLSI Physical Design Automation", Kluwer Academic Publishers, 1999.

[10] O. Coudert, "Exact Coloring of Real-Life Graphs is Easy", *Proc. 34th DAC,* June 1997.

[11] D. Kirovski and M. Potkonjak, "Efficient Coloring of a Large Spectrum of Graphs", *Proc.35th DAC,* June 1998

[12] L. He and K. M. Lepak, "Simultaneous Shield Insertion and Net Ordering for Capacitive and Inductive Coupling Minimization", U. of Wisconsin at Madison, *Technical Report ECE-00-2*.