NOSTRADAMUS: A FLOORPLANNER OF UNCERTAIN DESIGNS

K. Bazargan *S. Kim* *M. Sarrafzadeh*
Department of Electrical and Computer Engineering
Northwestern University
Evanston, IL 60208
{kiarash, samjung, majid}@ece.nwu.edu

ABSTRACT

Floorplanning is an early phase in chip planning. It provides information on approximate area, delay, power, and other performance measures. Careful floorplanning is thus of extreme importance. In many applications while a good floorplan is needed, not all modules' information are available, or even worse, part of the provided information is inaccurate. Floorplanning with uncertainty is the problem of obtaining a good floorplan under uncertainty. In this paper, the floorplanning problem with uncertainty is formulated. It is established that traditional floorplanners are incapable of handling uncertainty. An effective method for dealing with uncertain data is proposed. Experiments show that, for example, with up to 30% input uncertainty an area estimate with less than 7% error can be obtained.

1. INTRODUCTION

Floorplanning is a fundamental problem in the design process of complex chips. It is the highest level of the physical design process. Floorplanning, thus has a profound impact on the area, delay, power, and many other design parameters. To ensure an effective and reliable design, careful and accurate floorplanning is necessary.

Effective approaches to the floorplanning problem have been proposed in the past two decades. Among them are simulated annealing methods [17, 10], bottom-up and top-down hierarchical methods [2, 6, 15], linear and quadratics programming based techniques [12], force directed paradigms [13], clustering method [3], and techniques based on geometric dualization of the netlist [4, 1, 19, 20]. For other research related to floorplanning, see, e.g. [14, 18, 7]. For a detailed list of existing floorplanning methods see textbooks on physical design, e.g., [5, 11, 9].

Traditionally it has been assumed that the entire design (the netlist and the database library) was completely specified and available at the time of floorplanning. This is not a valid assumption anymore and will be less valid as designs are getting more complex. Floorplanning with uncertainty is the process of obtaining an accurate floorplan with missing data. More specifically, the netlist is known, however, dimensions of the modules are specified using probability distributions. There are many sources of uncertainty, among them:

1. *Missing cells*: After the architecture of a chip is defined it is necessary to obtain area and performance measures to assess whether the architecture is acceptable or if it has to be completely or partially redesigned. At this stage, not all cells/blocks have been designed. Estimates from the previous designs can be used to fill the missing data. For example, if a filter is being used in the design and it is known that a similar unit in the previous design had area A, and the new one is somewhat more complex than the previous one, the area of the new unit can be estimated as: 1.1A with probability .6 and 1.2A with probability .4. These estimates can be adjusted based on many other factors. For example, if the designer in charge is not experienced, area estimates can be adjusted to 1.2A with probability .4 and 1.3A with probability .6.
2. *Evolving library*: At the floorplanning stage, the existing library might be incomplete, for more instances of cells are being, or will be, designed (for better performance). Area of these cells can be estimated based on existing implementations of the cells and based on requirements of the new implementation. For example, if the new cell is to be faster than existing ones, its area can be estimated to be larger than existing ones.

A straightforward approach to floorplanning with uncertainty would be to use the expected area of a block and use that information in the floorplanning process. Experiments show that this is very naive and inaccurate (see Section 3.). A more conservative approach, where maximum possible area of a cell is used, is also highly inaccurate.

We propose a method that aims to minimize a linear combination of the expected value and the standard deviation of the final area. The proposed technique carries the modules' probability distribution functions (pdf) throughout the floorplanning process. At each stage, all dominant floorplans (i.e., those that may contribute to a minimum area floorplan) are obtained and redundant floorplans are discarded. Including standard deviation as part of the cost function will "hide" modules with uncertain shapes inside fixed modules. For example, if the expected width of a module is between 10 and 15, placing it below another module with width 16 would "hide" the uncertainty. The proposed floorplanning scheme achieves exactly that.

We also study the relationship between input and output uncertainty. It is shown that in some cases the two are highly correlated and in other cases they are not as correlated. Numerous results, some counter-intuitive, are demonstrated.

The rest of the paper is organized as follows: In Section 2., we formulate the problem of floorplanning with uncertainty, and explain how it differs from the traditional floorplanning problem. In Section 3., we show how to enable the traditional floorplanning algorithms to deal with uncertainty. We will also show why these methods cannot generate an accurate floorplan. Then we will present our method, called Nostradamus, which tries to minimize the expected value and standard deviation of the floorplan. In Section 4., we will formulate the correlation of standard deviation in input data to that of the final floorplan. Detailed experiments are shown in Section 5.. Different floorplanning parameters and cost functions are compared. A simulation environment is developed to test different paradigms. Section 6. contains conclusion and future work.

2. PROBLEM FORMULATION

The traditional floorplanning problem takes as input a set of modules (blocks), their widths and heights, and interconnections between them. It tries to find a floorplan such that the total area, delay, and power is minimized. Our work focuses only on area minimization, However, extending it to deal with other optimization goals is straightforward.

In this paper, we consider floorplanning with uncertain data. Such data will consist of width distribution lists $\mathcal{W}_i$ ($1 \leq i \leq n$) and height distribution lists $\mathcal{H}_i$ ($1 \leq i \leq n$), where n is number of modules. Each distribution list contains pairs of numbers: width (or height) of a module and its probability.

$$\mathcal{W}_i = \{(w_{i1}, p(w_{i1})), (w_{i2}, p(w_{i2})), \dots, (w_{im_{w_i}}, p(w_{im_{w_i}}))\} \qquad \sum_{j=1}^{m_{w_i}} p(w_{ij}) = 1$$

$$\mathcal{H}_i = \{(h_{i1}, p(h_{i1})), (h_{i2}, p(h_{i2})), \dots, (h_{im_{h_i}}, p(h_{im_{h_i}}))\} \qquad \sum_{j=1}^{m_{h_i}} p(h_{ij}) = 1$$

Equivalently, we can also consider the case where each width/height pair has a probability. Such a distribution would look like:

$$\mathcal{L}_i = \{(w_{i1}, h_{i1}, p_{i1}), (w_{i2}, h_{i2}, p_{i2}), \dots, (w_{im_i}, h_{im_i}, p_{im_i})\}$$

We will use distributions $\mathcal{W}_i$ and $\mathcal{H}_i$ throughout this paper, because they are easier to describe. However, the results are applicable to distributions $\mathcal{L}_i$.

A slicing floorplan is a rectangular floorplan that can be recursively partitioned into two floorplans by a horizontal or vertical line. Throughout the paper, we will focus on slicing floorplans because they can be demonstrated better. Results are readily generalizable to non-slicing floorplans. (For a discussion on how to transform non-slicing floorplans to slicing ones see [8].)

In traditional floorplanning, when two slices of dimensions (w_1, h_1) and (w_2, h_2) are clustered vertically into a larger block $(w_{1,2}, h_{1,2})$ (or in other words, block (1,2) is sliced vertically into slices 1 and 2, the dimensions of block (1,2) can be calculated using the following two equations:

$$w_{1,2} = w_1 + w_2 \tag{1}$$
$$h_{1,2} = max(h_1, h_2) \tag{2}$$

When the same slices are clustered horizontally,

$$w_{1,2} = max(w_1, w_2) \tag{3}$$
$$h_{1,2} = h_1 + h_2 \tag{4}$$

Different clusterings of the initial modules results in different floorplans. Different algorithms aimed at minimizing the area of the final floorplan have been proposed. As an example, see [16, 5, 11, 9]. We have revised the simulated annealing method to solve the floorplanning problem with uncertainty. The simulated annealing algorithm starts with an arbitrary sliceable floorplan. Polish expressions can be used to represent sliceable floorplans. An example of such expression together with the corresponding floorplan is shown in Figure 1. The annealing algorithm takes a *normalized* polish expression as input, and performs the following "moves" on it to get another expression:

1. Exchange two operands when there are no other operands in between.
2. Complement a series of operators between two operands, i.e., change horizontal signs (H in Figure 1) to vertical ones (V in Figure 1) and vice versa.
3. Exchange adjacent operand and operator if the resulting expression is still a normalized polish expression.

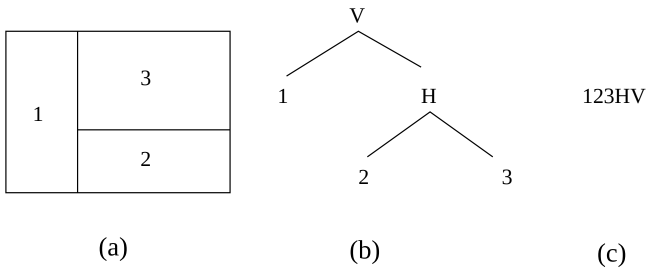

Figure 1. **(a)** a floorplan **(b)** slicing tree **(c)** corresponding polish expression

Details of the annealing algorithm can be found in [16, 17, 5, 11, 9]. Here we will highlight the modifications we have made to the algorithm to deal with uncertainty. To take uncertainty into consideration, the vertical clustering of module 1, as characterized by distributions $\mathcal{W}_1$ and $\mathcal{H}_1$, and module 2, as characterized by distributions $\mathcal{W}_2$ and $\mathcal{H}_2$, will yield the following distributions of widths and heights for module $(1, 2)$:

$$\mathcal{W}_{1,2} = \mathcal{W}_1 \oplus \mathcal{W}_2 \tag{5}$$
$$\mathcal{H}_{1,2} = \mathcal{H}_1 \circledcirc \mathcal{H}_2 \tag{6}$$

The operations $\oplus$ and $\circledcirc$ are *distribution addition* and *distribution maximum* operations, respectively, and are defined as follows:

$$\mathcal{D}_1 \oplus \mathcal{D}_2 = \Big\{ \big(d_{1i} + d_{2j} \ , \ p(d_{1i}) * p(d_{2j})\big) \ \Big| \ (d_{1i}, p(d_{1i})) \in \mathcal{D}_1 \text{ and } (d_{2j}, p(d_{2j})) \in \mathcal{D}_2 \Big\} \tag{7}$$

$$\mathcal{D}_1 \circledcirc \mathcal{D}_2 = \Big\{ \Big(d_{1i}, p(d_{1i}) * \sum_{j: d_{2j} \leq d_{1i}} p(d_{2j})\Big) \ \Big| \ (d_{1i}, p(d_{1i})) \in \mathcal{D}_1 \text{ and } (d_{2j}, p(d_{2j})) \in \mathcal{D}_2 \Big\} \bigcup \Big\{ \Big(d_{2j}, p(d_{2j}) * \sum_{i: d_{1i} < d_{2j}} p(d_{1i})\Big) \ \Big| \ (d_{1i}, p(d_{1i})) \in \mathcal{D}_1 \text{ and } (d_{2j}, p(d_{2j})) \in \mathcal{D}_2 \Big\} \tag{8}$$

Equation 7 implies that in order to add two random variables with distributions $\mathcal{D}_1$ and $\mathcal{D}_2$, we should create a new distribution whose elements are pairwise "addition" of elements from the two distribution lists. This "addition" of two elements is done by adding values and multiplying their probabilities. At the end, if two elements of the distribution list $\mathcal{D}_1 \oplus \mathcal{D}_2$ have the same value, we should replace them with a new one whose value is the same as their values and its probability is the addition of their probabilities.

As an example, suppose $\mathcal{W}_1 = \{(5,.3),(7,.5),(8,.2)\}$ and $\mathcal{W}_2 = \{(2,.9),(3,.1)\}$. When we cluster them vertically, the resulting distribution will be:

$$\begin{aligned}\mathcal{W}_{1,2} &= \{(5+2,.3*.9),(5+3,.3*.1),(7+2,.5*.9),\\ &\quad (7+3,.5*.1),(8+2,.2*.9),(8+3,.2*.1)\}\\ &= \{(7,.27),(8,.03),(9,.45),(10,.05),(10,.18),\\ &\quad (11,.02)\}\\ &= \{(7,.27),(8,.03),(9,.45),(10,.23),(11,.02)\}\end{aligned}$$

The distribution list of $\mathcal{D}_1 \textcircled{m} \mathcal{D}_2$ consists of elements which are "maximum" of pairwise elements of the two distribution lists. The value of the "maximum" of two elements is the maximum of their values and its probability is the product of their probabilities. Equation 8 is a formal way of describing the relation.

As an example, suppose $\mathcal{H}_1 = \{(1,.1),(2,.2),(7,.7)\}$ and $\mathcal{H}_2 = \{(4,.4),(6,.6)\}$. If we cluster them vertically, the resulting distribution would be:

$$\begin{aligned}\mathcal{H}_{1,2} &= \{(7,.7*.4),(7,.7*.6),(4,.4*.1),(4,.4*.2),\\ &\quad (6,.6*.1),(6,.6*.2)\}\\ &= \{(7,.7*(.4+.6)),(4,.4*(.1+.2)),\\ &\quad (6,.6*(.1+.2))\}\\ &= \{(7,.7),(4,.12),(6,.18)\}\end{aligned}$$

3. FLOORPLANNING METHODS FOR UNCERTAIN DATA

In this section, we will describe how to enable traditional floorplanning algorithms to handle uncertainty. We propose three methods to convert uncertain data to deterministic ones so that they can be fed in traditional algorithms. We will show why these methods do not yield accurate floorplans. In Subsection 3.2. we will describe an efficient approach.

3.1. Traditional Method

The traditional floorplanning algorithms take a list of module dimensions which contains one entry for each module's width and one for its height. In order to enable these algorithms to solve the problem with uncertain data, we have no way but to select one candidate for width and one for height of each cell. We propose three methods of performing this task:

1. *Optimistic method:* For each cell, pick the *minimum* value from the width list and minimum value from the height list and use them as width and height of the cell.
2. *Conservative method:* For each cell, pick the *maximum* value from the width list and the maximum value from the height list and use them as width and height of the cell.
3. *Expected value method:* For each cell, calculate the *expected value* of the width distribution and the expected value of the height distribution and use them as width and height.

After this transformation, we can run any traditional floorplanning program and hope for an accurate floorplan. Intuitively, the optimistic and the conservative methods will not yield realistic results and designers cannot rely on them to make high-level decisions. The actual floorplan will most probably have smaller area than the optimistic method and larger area than the conservative method.

As an example of poor performance of the optimistic and conservative methods, consider the following distributions of modules 1-3:

$$\begin{array}{lll} \mathcal{W}_1 = \{(3,.1),(5,.9)\} & & \mathcal{H}_1 = \{(1,1.0)\}\\ \mathcal{W}_2 = \{(2,1.0)\} & & \mathcal{H}_2 = \{(4,1.0)\}\\ \mathcal{W}_3 = \{(3,.85),(5,.15)\} & & \mathcal{H}_3 = \{(3,1.0)\}\end{array}$$

The optimistic method will choose 3 as the width of modules 1 and 3. The resulting floorplan is shown in Figure 2a. The estimated area is 20. However, if the actual width of module 1 happens to be 5 (when it is completely designed), the area of this floorplan will be 28, as shown in Figure 2b.

The conservative method chooses 5 as the width of modules 1 and 3. Figure 2c shows the floorplan generated by the conservative method. The area estimated by this method is 28, which is the same as the area of the floorplan when the actual values are used, but larger than the optimal floorplan. (See Figure 2d.) Although in this example both optimistic and conservative methods generated the same floorplan, this is not true in general.

The expected value method chooses 4.8 as the width of module 1, and 3.3 as the width of module 3. The floorplan generated by this method is the optimal floorplan for the final design, although the estimated area is larger than the actual optimal floorplan. Figure 2e shows the estimated floorplan generated by this method, and Figure 2f shows the optimal floorplan after all the modules have been designed.

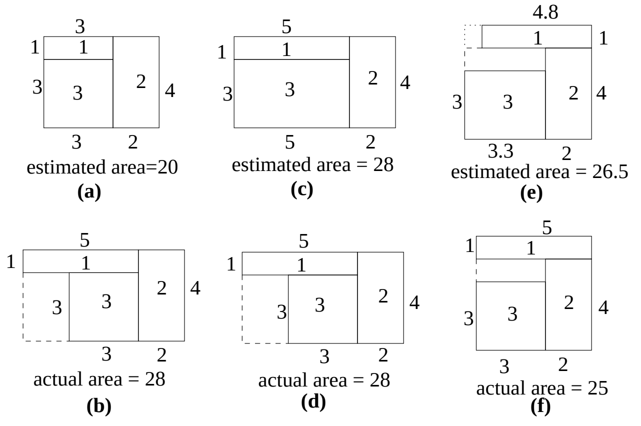

Figure 2. The numbers adjacent to the borders show the width and height of the modules. **(a)** floorplan by optimistic method assumes that $w_1 = w_3 = 3$ in final chip. **(b)** the actual size of optimistic floorplan after the modules are designed. **(c)** The floorplan generated by conservative method and its estimated area. w_1 and w_3 were estimated to be 5. **(d)** The actual conservative floorplan size after the modules are designed. **(e)** The floorplan generated by expected value method. It assumes that the final w_1 will be 4.8 and the final w_2 will be 3.3. **(f)** Minimum size floorplan for the actual dimensions

One might run both the optimistic and the conservative methods on the problem and look at both results to make design decisions. This cannot be a general solution. The difference between the two floorplan sizes might be so huge that no judgment can be made using those results. Using the expected value for widths and heights seems to be more reasonable. In subsequent sections, we will show that even this method does not generate compact floorplans.

3.2. The Nostradamus Floorplanner

Our algorithm, Nostradamus, is a modification of the traditional simulated annealing floorplanning algorithm [17]. We start with the distribution lists of width and height of cells, and a sliceable floorplan. Then we use Equations 5 and 6 and their horizontal counterparts to calculate distributions of the clustered modules. The clustering and calculating the corresponding distributions will be performed in a hierarchical manner until we get the distributions of width and height of the whole floorplan.

Reporting the distributions of the floorplan dimensions will provide more information to the user than simply reporting a width value and a height value, as is the case in the optimistic, the conservative or the expected value method described in Section 3.1..

If the only objective of the user is to get the distribution and try to minimize the area, the cost function during the annealing process would be $E(\mathcal{W}) * E(\mathcal{H})$, where $E(\mathcal{W})$ and $E(\mathcal{H})$ are respectively the expected value of width and height of the whole floorplan.

We can add variance to cost function and try to minimize both expected value and the variance of the area. The cost function which can be used is $\lambda * E(Area) + (1 - \lambda) * var(Area)$ where $E(Area)$ is the expected value of the floorplan area and $var(Area)$ is the variance of it. In order to get area distribution, one has to "multiply" [1] the width and height distributions. Since this process is both time and memory consuming, and should be done at every move in the annealing process, Nostradamus uses the following simpler cost function instead:

$$cost = \lambda * E(\mathcal{W}) * E(\mathcal{H}) + (1 - \lambda)(var(\mathcal{W}) * var(\mathcal{H})) \quad (9)$$

In the next section we will discuss details of our studies.

4. CORRELATING OUTPUT UNCERTAINTY TO INPUT UNCERTAINTY

To understand the nature of problem and how Nostradamus behaves to different input parameters, we have generated different sets of input data each with a different distribution. For example, data sets b1 – b4 contain uniform distributions of values within a range for each module's width/height. The range of values has fixed length and its starting point is selected randomly. As another example, set f1 contains distributions $\{(a - l, .05), (a, .9), (a + l, .05)\}$. l is a fixed number, and a is selected randomly for each width/height of each module.

Each input set contains width and height distributions for 50 modules. At first, we set λ to 1 (i.e., no effort in minimizing the variance) and try to find the relationship between input parameters and output parameters.

We ran Nostradamus with $\lambda = 1$ on each of the input sets. The final floorplan generated by Nostradamus is called "output" or "Nostradamus floorplan".

Figure 3 shows the width and height distributions of some Nostradamus floorplans. The figure contains distributions for two data sets. These floorplans are generated with $\lambda = 1$.

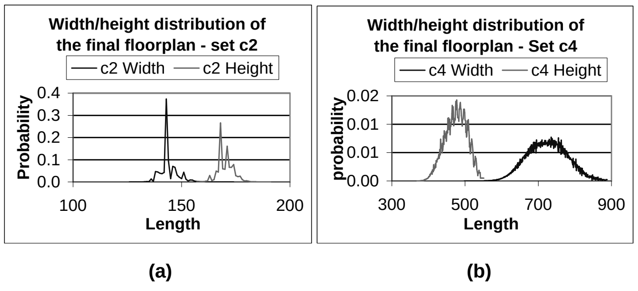

Figure 3. Nostradamus floorplan's width and height distribution for data sets c2 and c4 when $\lambda = 1$. Note that X and Y scales of the graphs are not the same.

A designer (or, a design team) would like to have a floorplan with as minimum a deviation as possible. A floorplan similar to c4 is practically useless. (See Figure 3b.) The user is provided with a very *flat* curve showing the possible values for width/height. The designer cannot make a concrete decision based on such floorplans.

[1]By "multiplication" we mean an operation similar to $\oplus$ which multiplies the values instead of adding them.

Figure 4 shows the relationship between input width deviation and Nostradamus floorplan's width deviation. Different data sets can be found on the x-axis of the graph. All the input deviations of data sets are scaled by 3.13 to make the comparison between input and output deviation easier. As can be seen, the output deviation follows input deviation very closely. Although we have not included the deviation graphs of height distributions here, the output height deviation also follows input height deviation.

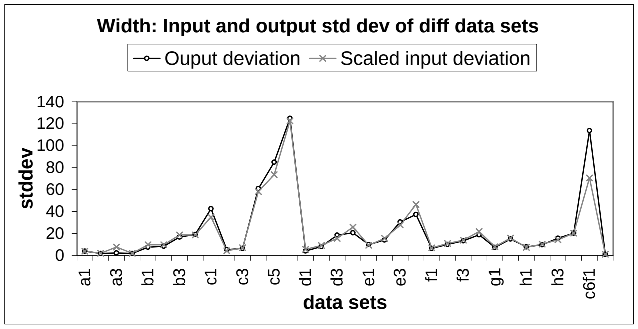

Figure 4. Width standard deviation of input sets and the corresponding width deviation of the Nostradamus floorplan. X-axis shows different input sets. $\lambda = 1$.

5. EXPERIMENTAL RESULTS

In this section experimental results on the effectiveness of the proposed floorplanner will be shown. In particular, we will show the effect of parameter λ on the final floorplan. We will also show that the input characteristic, e.g., the number of modules with uncertainty, has a profound impact on the final result.

We will describe a simulation environment, developed to measure the effectiveness of Nostradamus and compare it with other techniques proposed earlier (e.g., the conservative method).

5.1. Effect of λ on Nostradamus Floorplans

Different values of λ in Equation 9 generate different floorplans with distinct area and varying variance. Among all possible floorplans, it is not obvious which floorplan the user prefers: one with minimum expected value or one with minimum variance, or a solution in between. All floorplans generated for data sets g1 and h1 are shown in Figure 5. It shows different points with different expected value and standard deviation. The numbers next to the points show the value of λ which was used to generate the floorplan.

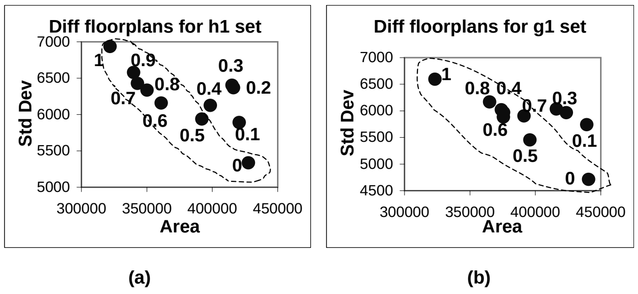

Figure 5. Different points show different expected value/standard deviation of Nostradamus floorplans. User might choose any of the floorplans inside the dominant set, shown by dotted curves.

The dotted curves highlight *the dominant set.* A floorplan is in the dominant set if and only if there is no other floorplan which has smaller area and smaller standard deviation than it. All members of the dominant set are candidate floorplans; none of them can be easily preferred over the other. For example, referring to Figure 5a, a designer might choose the floorplan generated with $\lambda = 0.8$ instead of $\lambda = 0.9$ since paying a small penalty in the expected area, the worst case (e.g., the variance) can be controlled.

5.2. Design Simulation Environment

To see how well Nostradamus might work in practice, we have developed a simulation environment as demonstrated in Figure 6. At first, width/height distributions are generated for each of the modules. In reality, the design team or the design manager provides these data based on the scenarios described in Section 1. In our experiments, random data and the controlled data described in Section 4. have been used.

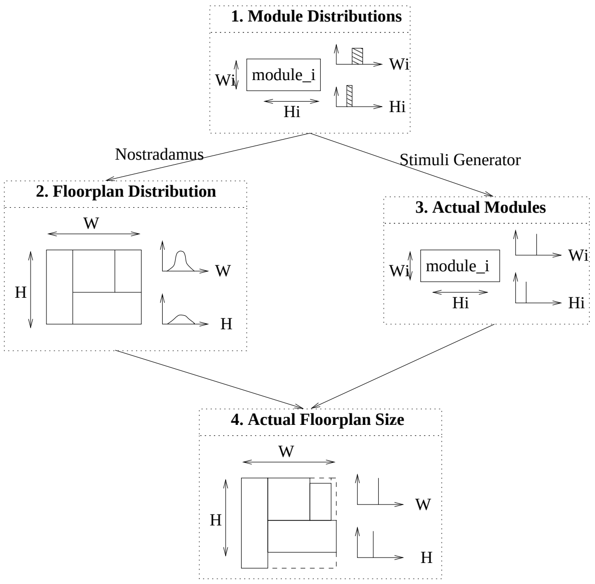

Figure 6. Design simulation process. The floorplan generated by Nostradamus from module distributions is sized by actual width and height generated by stimuli generator. Nostradamus can be replaced by other methods such as the optimistic method so that we can compare the resulting floorplans.

Then we run Nostradamus on probabilistic data. Different values of λ are tried out and an arbitrary floorplan in the dominant set is selected. This process is shown as box 2 in Figure 6.

We also ran the optimistic, conservative and expected value methods (see Section 3.1. for a description of these methods) on the same input data. Each of the methods generates a floorplan. Although we have not shown them, the boxes corresponding to these steps are similar to box 2 in Figure 6, that is, a floorplan distribution is generated.

After running Nostradamus and the other methods, we test the process by choosing a stimuli. The stimuli is generated by finding a point in the distribution list of each module in accordance with the dictated probabilities. The stimuli represent the values of the completed, and completely specified design. See box 3 in Figure 6. Considering these dimensions, we calculate the area of the sized floorplans generated by Nostradamus, optimistic, conservative and expected value methods. This is shown in box 4 in Figure 6.

5.3. Comparing Different Methods

Another important parameter in input data is "uncertainty percentage". We say an input data set is x% uncertain if x% of its modules have probabilistic dimensions and the other modules have deterministic dimensions, i.e., the rest have exactly one width and one height value.

Figures 7 to 10 show the actual area of optimistic, conservative, expected value and Nostradamus floorplans for different uncertainty values of input data. The actual area is calculated using the method described in Section 5.2..

As can be seen, when all (or almost all) module dimensions' are uncertain then it is not possible to "hide" uncertain ones inside the rest. Therefore, all methods generate more or less the same result, as shown in Figure 10. In the other extreme, when all (or almost all) module dimensions' are known, again it is not possible to "hide" uncertain ones inside the rest since there are not many uncertain ones. Again, in this case, all methods generate more or less the same result, as shown in and Figure 7. When uncertainty is between, 30 to 70 percent, the proposed method is very effective in generating compact floorplans by anticipating the final result, as shown in Figures 9 and 8.

Finally Figure 11 shows a summary of how Nostradamus behaves under different uncertainty measures. As one might expect, the more uncertainty results in less accurate prediction capability.

6. CONCLUSION

We have proposed a floorplanner that is capable of dealing with modules' uncertainty. It generates a compact floorplan by anticipating the final dimensions of the modules. The proposed floorplanner, called Nostradamus, can be tuned for minimum expected area or for minimum variance. Experiments show that planning for uncertainty does pay off and results in compact floorplans.

Our research is a start on floorplanning with uncertainty. Obtaining analytical bounds on the area of the final floorplan, based on the amount and type of uncertainty in input, remains open. There are many other uncertainty factors, for example, uncertainty on the number and types of nets. This is an interesting and challenging problem. In this case, even the modeling aspect of the problem is non-trivial.

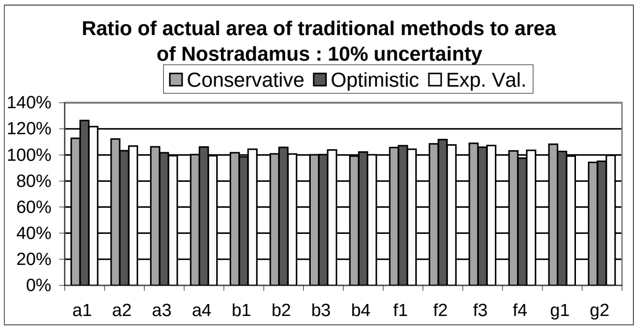

Figure 7. The ratio of actual floorplan size of traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 10%.

7. ACKNOWLEDGMENTS

This research has been sponsored in part by a grant from the Defense Advanced Research Projects Agency, and a grant from the National Science Foundation. Also, we would like to thank Dr. Naveed Sherwani (Intel) for discussions on the modeling aspect of the problem.

REFERENCES

[1] J. Bhasker and S. Sahni. "A Linear Algorithm to Find a Rectangular Dual of a Planar Triangulated Graph". *Algorithmica*, 3(2):274–278, 1988.

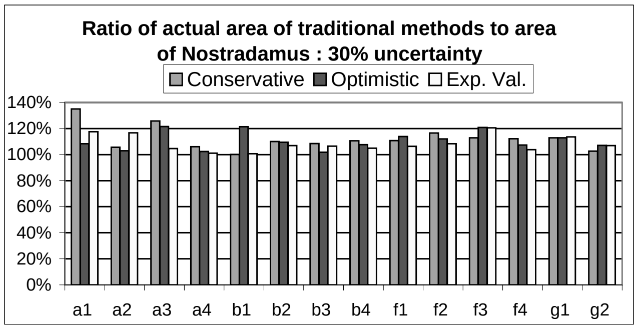

Figure 8. The ratio of actual floorplan size of traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 30%.

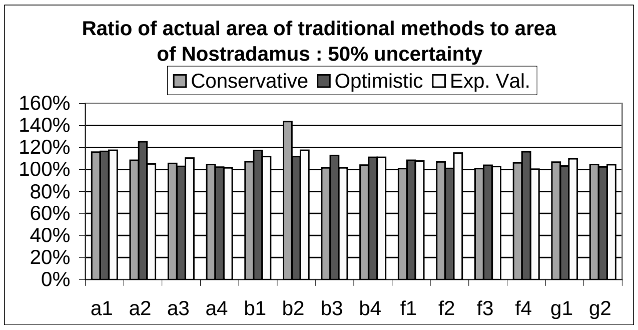

Figure 9. The ratio of actual floorplan size of traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 50%.

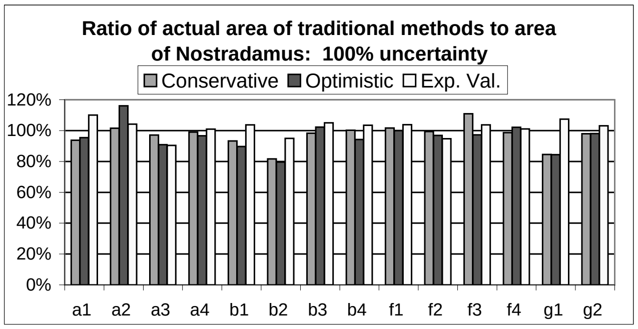

Figure 10. The ratio of actual floorplan size of traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 100%.

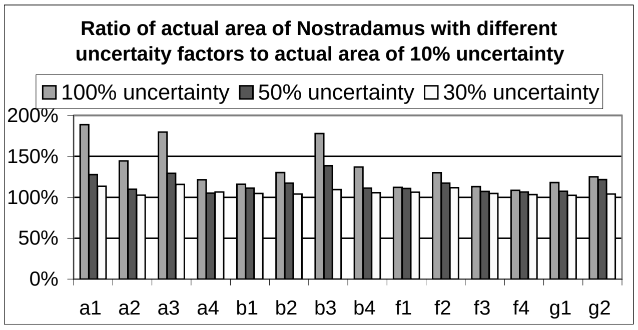

Figure 11. The ratio of actual floorplan size of Nostradamus with different percentages of uncertainty to 10% uncertainty. In all cases, $\lambda = .7$.

[2] W. M. Dai and E. S. Kuh. "Simultaneous Floor Planning and Global Routing for Hierarchical Building-Block Layout". *IEEE Transactions on Computer Aided Design*, 6(5):828–837, 1987.

[3] J. D. Gabbe and P. A. Subrahmanyam. "A Note on Clustering Modules for Floorplanning". In *Design Automation Conference*, pages 594–597. IEEE/ACM, 1989.

[4] K. Kozminski and E. Kinnen. "Rectangular Dual of Planar Graphs". *Networks*, 15:145–157, 1985.

[5] T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout.* John Wiley & Sons, 1990.

[6] T. Lengauer and R. Muller. "A Robust Framework for Hierarchical Floorplanning with Integrated Global Wiring". In *International Conference on Computer-Aided Design*, pages 148–151, 1990.

[7] M. Pedram, M. Marek-Sadowska, and E. Kuh. "Floorplanning with Pin Assignment". In *International Conference on Computer-Aided Design*, pages 98–101, 1990.

[8] M. Sarrafzadeh. "Transforming an Arbitrary Floorplan into a Sliceable One". In *International Conference on Computer-Aided Design*, pages 386–389. IEEE/ACM, 1993.

[9] M. Sarrafzadeh and C. K. Wong. *An Introduction to VLSI Physical Design.* McGraw-Hill, 1996.

[10] C. Sechen and K. W. Lee. "An Improved Simulated Annealing Algorithm for Row–Based Placement". In *Design Automation Conference*, pages 180–183. IEEE/ACM, 1988.

[11] N. A. Sherwani. *Algorithms For VLSI Physical Design Automation.* Kluwer Academic Publishers, 1993.

[12] S. Sutanthavibul, E. Shragowitz, and J. B. Rosen. "An Analytical Approach to Floorplan Design and Optimization". *IEEE Transactions on Computer Aided Design*, 10(6):761–769, 1991.

[13] K. Ueda, H. Kitazawa, and I. Harada. "CHAMP: Chip Floorplan for Hierarchical VLSI Layout Design". In *IEEE Transactions on Computer Aided Design*, pages 12–22, 1986.

[14] T. C. Wang and D. F. Wong. "An Optimal Algorithm for Floorplan Area Optimization". In *Design Automation Conference*, pages 180–186. IEEE, 1990.

[15] T. C. Wang and D. F. Wong. "A Graph Theoretic Technique to Speed up Floorplan Area Optimization". In *Design Automation Conference*, pages 62–68. IEEE, 1992.

[16] D. F. Wong, H. W. Leong, and C. L. Liu. *Simulated Annealing for VLSI Design.* Kluwer Academic, 1988.

[17] D. F. Wong and C. L. Liu. "Floorplan Design of VLSI Circuits". *Algorithmica*, 4:263–291, 1989.

[18] D. F. Wong and P. S. Sakhamuri. "Efficient Floorplan Area Optimization". In *Design Automation Conference*, pages 586–589. IEEE/ACM, 1989.

[19] K. H. Yeap and M. Sarrafzadeh. "Floorplanning by Graph Dualization: Two-concave Rectilinear Modules". *SIAM Journal on Computing*, 22(3):500–526, June 1993.

[20] K. H. Yeap and M. Sarrafzadeh. "Sliceable Floorplanning by Graph Dualization". *SIAM Journal on Discrete Mathematics*, 8(2):258–280, 1995.