Nostradamus: A Floorplanner of Uncertain Designs

K. Bazargan\textsuperscript{1}, S. Kim\textsuperscript{1}S and M. Sarrafzadeh\textsuperscript{1}

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 where a good floorplan is needed, the information about all modules is not available, or even worse, part of the provided information is inaccurate. Examples of such applications are designing a large system where the floorplan is needed early in the design process, but not all the modules have been designed. Another example is the field of reconfigurable computing where it is not known what modules will be needed on the reconfigurable chip as the program is being executed. Floorplanning with uncertainty is the problem of obtaining a good floorplan when the information about module dimensions is not complete. In this paper, the floorplanning problem with uncertainty is formulated. Correlation between input characteristics and output characteristics is studied. Also, it is established that traditional floorplanners are incapable of efficiently 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.

Keywords—Floorplanning, area estimation, ECO, design planning, reconfigurable computing

I. INTRODUCTION

In recent years, the improvements in VLSI fabrication technology has enabled designers to place larger, faster and more complex circuits on chips. Smaller feature sizes in deep-submicron technology have led to more compaction of the design, which in turn improves the performance and decreases the area of the chip. The decrease in chip area helps designers to accommodate more logic on the chip using the same area as before. Consequently, designs are getting more complex.

With current CAD tools, the design process time has increased to years for such complex circuits. Products consist of many components, each being designed by a team of engineers, and requiring different lengths of time to be designed. Some of the components, or modules, must be designed from scratch, whereas some others are modifications of components from previous products. Generally, in comparison to the more complex modules, less time is needed for simpler modules or those similar to ones in the design library.

As designers try to optimize design measures such as area, delay and power more aggressively, there is an urgent need for the CAD tools to be able to generate floorplans as early as possible in the design process. Traditionally, CAD engineers wait until all modules and their interconnecting structure are designed in detail before they proceed to floorplanning, placement and routing of the circuit. There are two reasons why this strategy no longer meets the needs of CAD engineers:

1. Early cost/performance estimations are necessary to avoid wasting resources. Estimation of different design measures is not possible without a floorplan. After the architecture of a chip is defined, area and performance measures should be used to assess the potential market share of the product and to decide if major changes in the architecture are necessary. The sooner the designers can decide on such issues, the less resources are wasted.

2. By generating the floorplan earlier, some groups can start their work earlier and hence the total design time will decrease. Some important and time consuming processes can only be done after floorplanning. An example of such processes is routing the clock net. As another example, in more aggressive designs designers need to find the pattern of heat dissipation on the chip to measure the performance of the circuit more accurately. Again, this task can only be done after the floorplanning process.

Another example in which generating a floorplan is needed before all the information about the modules is present happens in adaptive/reconfigurable computing systems. In such systems, the set of modules which should be loaded on the reconfigurable chip will be known as the program runs on the adaptive host. By guessing, at compile time, what modules will be needed at run time and coming up with a potentially good floorplan beforehand, we might be able to gain performance improvements.

Our goal is to devise a method which is capable of generating a floorplan early in the design process, i.e., when some modules have not been completely designed yet. Using some estimations about the final area of different modules, our system should be able to generate a reliable floorplan. A reliable floorplan is one whose area at the time it is generated does not differ significantly from its area when all modules are designed completely. Furthermore, the difference between the area of the reliable floorplan and the area of the optimal floorplan, when the information about module dimensions is complete, should be as small as possible.

Besides generating a reliable floorplan, we should answer an important question: how should the input, i.e. estimation of module dimensions, look like so that we can generate a reliable floorplan more easily? In other words, is it easier to have a set of modules, most of which have deterministic dimensions and a few with high uncertainty, or it is more desirable to have a set of modules, most of which
have uncertain dimensions but with a small degree of uncertainty? How much uncertainty is acceptable to generate a reliable floorplan? A similar question is: When to start generating the reliable floorplan? Generating the floorplan very early has the disadvantage that the information about module dimensions is very inaccurate. On the other hand, waiting for a long time to start floorplanning lengthens the design process. We would like to address these questions by examining the relationship between input uncertainty and the uncertainty in the dimensions of the reliable floorplan.

Our approach, floorplanning with uncertainty, tries to achieve the above goals. More specifically, we deal with the problem of floorplanning where the interconnection structure is specified, but the dimensions of modules are not known. Instead of using fixed dimensions for modules, we employ probability distributions to represent the final width and height of each module. The distributions are estimations of what the dimensions of modules would be in the final product.

There are many ways to estimate the final dimensions of the modules. For example,

1. Previous designs: In the early phases of the design some modules have not been designed, however, the designers have some idea of how large the module would be. Estimates from previous designs can be used to guess the dimensions of the new modules. 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, 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 $A$ and $1.3A$ with probability $0.6$.

2. Evolving library. At the floorplanning stage the existing library might be incomplete, because some modules are in their early phases of design, and will change as the designing process goes on. More instances of a module will be designed which have better area or performance than the previous versions. As a result, the area of the module at the time of floorplanning will not be the same as the time all modules have been designed completely. The final area of the module can be estimated based on existing implementations of it and based on requirements of the new implementation. For example, if the new module is to be faster than existing ones, its area can be estimated to be larger.

The input to the process of floorplanning with uncertainty is a list of width and height values for each module, with some probability assigned to each value. A straightforward approach to floorplanning with uncertainty would be to use the expected value of a module's width and height as a guess for its final dimensions, and use that information in the floorplanning process. Experiments show that this is very naive and inaccurate (see Section III). A more conservative approach, where the maximum possible area of a module 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 floorplan's area. Minimization of the expected value of the area leads to reduction in cost since the cost of a chip decreases exponentially with decreases in the area of the chip. Minimizing the deviation of the area helps us achieve a reliable floorplan. The area of such a floorplan does not change much with changes in uncertain module dimensions because its deviation is small.

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" the uncertainty of a module's dimensions by placing it near a module with fixed dimensions. For example, if the width of a module is supposed to be between 100 and 150, with uniform probability, placing it below another module with width 160 would "hide" the uncertainty. The proposed floorplanning scheme achieves exactly that.

We also study the relationship between variance in module dimensions used as input to the floorplanning process, to the variance of the dimensions of the final floorplan. It is shown that in most cases the two are highly correlated. Numerous results, some counter-intuitive, are demonstrated.

The paper is organized as follows: In Section II we formulate the problem of floorplanning with module uncertainty and explain how it differs from the traditional floorplanning problem. In Section III we show how to enable the traditional floorplanning algorithms to deal with uncertainty in module dimensions. We will also show why these methods cannot generate a reliable floorplan. Then we will present our method, called Nostradamus, which tries to minimize the expected value and standard deviation of the floorplan. In Section IV we formulate the correlation of standard deviation in input data to that of the final floorplan. Section V includes detailed experimental results. Different combinations of characteristics of the floorplan are targeted. We have developed a process to simulate the real-life design cycle to compare floorplans generated by our method to those generated by traditional methods. Section VI explains how to extend Nostradamus so that it can handle a more realistic model for uncertainty. Experimental results using this model are shown as well. Section VII contains conclusion and future work.

II. Problem Formulation

Effective approaches to the floorplanning problem have been proposed in the past two decades. Among them are simulated annealing methods [12], [19], bottom-up and top-down hierarchical methods [4], [8], [17], linear and quadratics programming based techniques [14], force directed paradigms [15], clustering methods [5] and techniques based on geometric dualization of the netlist [3], [6], [21], [22]. For other research related to floorplanning, see, e.g., [9], [16], [20]. For a detailed list of existing floorplanning methods see textbooks on physical design, e.g., [7], [11], [13].
The traditional floorplanning problem takes as input a set of modules (blocks), their widths and heights, and netlist information. It tries to find a floorplan such that the total area or delay or power, or a combination of them, is minimized. Our work focuses only on area minimization, however, once the floorplan is generated other measures can be easily calculated and hence minimized.

In this paper we consider floorplanning with uncertain data. Data with uncertainty will consist of width distribution lists $W_i$ ($1 \leq i \leq n$) and height distribution lists $H_i$ ($1 \leq i \leq n$), where $n$ is number of modules. Each distribution list contains pairs of numbers: width (or height) of a module and its probability.

$$W_i = \{(w_{i1}, p(w_{i1})), (w_{i2}, p(w_{i2})), \ldots, (w_{in}, p(w_{in}))\} \quad \sum_{j=1}^{m_i} p(w_{ij}) = 1$$

$$H_i = \{(h_{i1}, p(h_{i1})), (h_{i2}, p(h_{i2})), \ldots, (h_{in}, p(h_{in}))\} \quad \sum_{j=1}^{m_i} p(h_{ij}) = 1$$

As a more realistic, but more complex model, we could represent the dimensions of a module as a list of triplets $(w, h, p)$, where $p$ is the probability that the width and height of the module are respectively $w$ and $h$ in the final product. Each module will have a list of such triplets, e.g., the distribution list for module $i$ is represented as:

$$L_i = \{(w_{i1}, h_{i1}, p_{i1}), (w_{i2}, h_{i2}, p_{i2}), \ldots, (w_{in}, h_{in}, p_{in})\}$$

We will use distributions $W_i$ and $H_i$ throughout the paper, except in Section VI in which we will use distributions $L_i$. We will show how we have modified our program to handle these distributions as well.

Here, we concentrate on slicing floorplans. A slicing floorplan is a rectangular floorplan that can be recursively partitioned into two floorplans by a horizontal or vertical line. Results are readily generalizable to non-slicing floorplans. For a discussion on how to transform non-slicing floorplans to slicing ones see [10].

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, as shown in Figure 1), the dimensions of block $(1,2)$ can be calculated using the following two equations:

$$w_{1,2} = w_1 + w_2$$

$$h_{1,2} = \max(h_1, h_2)$$

When the same slices are clustered horizontally,

$$w_{1,2} = \max(w_1, w_2)$$

$$h_{1,2} = h_1 + h_2$$

Different clusterings of the initial modules result in different floorplans. Various algorithms aimed at minimizing the area of the floorplan have been proposed. As an example, see [7], [11], [13], [18], [19]. We have revised the simulated annealing method used in [19] 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 expressions together with the corresponding floorplan is shown in Figure 2. 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 2) to vertical ones (V in Figure 2) and vice versa.
3. Exchange adjacent operand and operator if the resulting expression is still a normalized polish expression.

Details of the annealing algorithm can be found in [7], [11], [13], [18], [19]. Here we will only 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 $W_1$ and $H_1$, and module 2, as characterized by distributions $W_2$ and $H_2$, will yield the following distributions of width and height for module $(1,2)$:

$$W_{1,2} = W_1 \oplus W_2$$

$$H_{1,2} = H_1 \oplus H_2$$

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

$$D_1 \oplus D_2 = \{ (d_{i1} + d_{j2}, p(d_{i1}) \ast p(d_{j2})) \}$$

![Fig. 1. Block (1,2) sliced vertically into blocks 1 and 2](image)

![Fig. 2. (a) a floorplan (b) slicing tree (c) corresponding polish expression](image)
(d_{1i}, p(d_{1i})) \in D_1 \text{ and } (d_{2j}, p(d_{2j})) \in D_2 \right) \quad (7)
\begin{align*}
D_1 \otimes D_2 &= \left\{ \left( d_{1i}, p(d_{1i}) \right) \mid \sum_{j : d_{2j} \leq d_{1i}} p(d_{2j}) \right\} \\
&\cup \left\{ \left( d_{2j}, p(d_{2j}) \right) \mid \sum_{i : d_{1i} \leq d_{2j}} p(d_{1i}) \right\} \quad (8)
\end{align*}

Equation 7 implies that in order to add two random variables with distributions $D_1$ and $D_2$, we should create a new distribution whose elements are pairwise “addition” of elements from the two distribution lists. The “addition” of two elements is done by adding width/height values and multiplying their probabilities. At the end, if some elements of the distribution list $D_1 \otimes D_2$ have the same width/height, we should replace them with a new one whose width/height value is the same as theirs and its probability is the addition of their probabilities.

As an example, suppose $W_1 = \{(5,3),(7,5),(8,2)\}$ and $W_2 = \{(2,9),(3,1)\}$. When we cluster them vertically, the resulting distribution will be:

$W_{1,2} = \{(5+2,3+9),(5+3,3+1),(7+2,5+9), (7+3,5+1),(8+2,2+9),(8+3,2+1)\}$

$= \{(7,27), (8,03), (9,45), (10,05), (10,18), (11,02)\}$

The distribution list of $D_1 \otimes D_2$ consists of elements which are “maximum” of pairwise elements of the two distribution lists. The width/height 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 $H_1 = \{(1,1),(2,2),(7,7)\}$ and $H_2 = \{(4,4),(6,6)\}$. If we cluster them vertically, the resulting distribution would be:

$H_{1,2} = \{(7,7*4),(7,7*6),(4,4*1),(4,4*2), (6,6*1),(6,6*2)\}$

$= \{(7,7*(4+6)), (4,4*(1+2)), (6,6*(1+2))\}$

$= \{(7,7),(4,12),(6,18)\}$

III. 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 used as input to the traditional algorithms. We will show why these methods do not yield reliable floorplans. In Subsection III-B we will describe an efficient approach for floorplanning with uncertainty.

A. 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. These algorithms cannot solve the problem with uncertain data unless we select only one candidate for width and one for height of each module. We propose three methods for performing this task:

1. Optimistic method: For each module, pick the minimum value from the width list and minimum value from the height list and use them as width and height of the module.

2. Conservative method: For each module, 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 module.

3. Expected value method: For each module, 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 a good floorplan for the final product. Intuitively, the optimistic and the conservative methods will not yield realistic results and designers cannot rely on them to make high-level decisions such as if major changes in the architecture of the product is needed. The final floorplan (the optimal floorplan generated after all modules are designed) will most probably have larger area than the optimistic method and smaller 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:

$W_1 = \{(3,1),(5,9)\}$

$W_2 = \{(2,1,0)\}$

$W_3 = \{(3,85),(5,15)\}$

The optimistic method is not able to generate a reliable floorplan for the final product in advance, because it is not able to estimate the final dimensions of modules well. The method anticipates that at the end of the design process, the width of modules 1 and 3 will be 3, and given that, the minimum area floorplan is the one shown in Figure 3a, with area 20. The area reported by the method is an estimation of the area of the minimum-area floorplan at the time all modules have been designed completely. The proposed floorplan might not be the best at the end of the process because the final dimensions of some modules might differ from those anticipated by the method. Consequently, the area reported by the method might be far off the area of the optimal floorplan at the end of the process. In fact, if the actual width of module 1 happened to be 5 (when it is completely designed), the area of the floorplan generated by the optimistic method will be 28, as shown in Figure 3b.

The conservative method cannot generate a reliable floorplan either. The method chooses 5 as the width of modules 1 and 3, generating the floorplan shown in Figure 3c. The area estimated by the method is 28, which is the same as the area of the floorplan when the final values are
used, but larger than the optimal floorplan. (See Figure 3a.) 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 for the final design. Figure 3e shows the estimated floorplan generated by this method, and Figure 3f shows the optimal floorplan after all the modules have been designed. More dramatic results can be demonstrated by running larger examples.

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 for the final design.

B. The Nostradamus Floorplanner

Our algorithm, Nostradamus, is a modification of the traditional simulated annealing floorplanning algorithm [19]. We start with the distribution lists of width and height of modules, 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 Subsection III-A.

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 \( E(W) \times E(H) \), where \( E(W) \) and \( E(H) \) are the expected value of width and height of the whole floorplan respectively.

We can add variance to cost function and try to minimize both expected value and the variance of the area. Nostradamus uses the following cost function:

\[
\text{cost} = \lambda \times E(W) \times E(H) + \text{var}(W) \times \text{var}(H)
\]

The cost function increases as the expected area of the floorplan or its variance increase. By using larger values for \( \lambda \) we can create more compact floorplans, because \( E(W) \times E(H) \) approximates the expected value of the area. On the other hand, using smaller values of \( \lambda \) will cause the floorplanner to create more reliable floorplans. In the next section we will discuss details of our studies.

IV. CORRELATING OUTPUT UNCERTAINTY TO INPUT UNCERTAINTY

To understand the nature of the problem and how Nostradamus behaves to different input distributions, we have generated different classes of input data. We have used various distribution functions with different variances as input to Nostradamus. Each input set contains width and height distributions for 50 modules. At first we set \( \lambda \) (the cost function normalizer as defined in Subsection III-B) to 1 (i.e., no effort in minimizing the variance) and try to find the relationship between input variance and output variance.

Figure 4 shows the width and height distributions of some Nostradamus floorplans. The figure contains distributions for two data sets. These floorplans are generated with \( \lambda = 1 \). A designer (or design team) would like to have a floorplan with as minimum a deviation as possible. A floorplan similar to c4, shown in Figure 4b, is practically useless. The user is provided with a very flat curve showing the possible values for width/height, and hence he/she cannot make a concrete decision based on the area of the floorplan. Such floorplans can show that the design is still uncertain and the floorplanning should be done at a later stage.

Figures 5 and 6 show the relationship between input deviation and the deviations of the width and height of floorplans generated by Nostradamus. Different data sets can be found on the x-axis of the graph. All input deviations of data sets are scaled by a constant (in our example 3.13. The number depends on number of modules and the distribution type of the input.) to make the comparison between input and output deviation easier. As can be seen, the output deviation follows the input deviation very closely.

From the above figures and Figure 4 we conclude that if all module dimensions in the input are very uncertain,
then the generated floorplan will also have uncertain area. However, we will show in Section V that if some modules have very uncertain dimensions but the rest have fixed width/height, then generating a reliable floorplan is possible. The reason is that when some modules have fixed dimensions, Nostradamus can hide the uncertainty of the rest (with probabilistic dimensions) by placing them adjacent to the fixed ones.

V. EXPERIMENTAL RESULTS

In this chapter experimental results on the effectiveness of the proposed floorplanner will be shown. In particular, we will show the effect of parameter $\lambda$ 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 also describe a simulation environment, developed to measure the effectiveness of Nostradamus and compare it to other techniques proposed earlier (e.g., the conservative method).

A. Effect of $\lambda$ on Nostradamus Floorplans

Different values of $\lambda$ 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 7. Points show the expected value and the standard deviation of floorplans generated with different values of $\lambda$. The numbers next to the points show the value of $\lambda$ which was used to generate the floorplan.

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 7a, a designer might choose the floorplan generated with $\lambda = 0.8$ instead of $\lambda = 0.9$ since by paying a small penalty in the expected area, he/she can get a more reliable floorplan. The area of such a floorplan is less likely to change after all modules are designed.

B. Design Simulation Environment

To see how well Nostradamus might work in practice, we have developed a simulation environment as demonstrated in Figure 8. At first, width/height distributions are generated for each of the modules. In reality, the design team or the design manager provides such data based on the scenarios described in Chapter I. In our experiments, we used random data and the controlled data described in Chapter IV. Then we run Nostradamus on probabilistic data. Different values of $\lambda$ are tried out and an arbitrary floorplan in the dominant set is selected. This process is shown as box 2 in Figure 8.

We also ran the optimistic, conservative and expected value methods (see Subsection III-A 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 8, that is, a floorplan distribution is generated.

After running Nostradamus and also running 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 8. 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 8.

C. Comparing Different Methods

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

Figures 9 to 12 show the actual area of the optimistic, the conservative, the expected value and Nostradamus floorplans for different uncertainty values of input data. The actual area is calculated using the method described in Subsection V-B.

As can be seen, when all (or almost all) modules have uncertain dimensions, 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 9. On the other extreme, when all (or almost all) modules’ dimensions are known, again it is not possible to hide uncertain ones inside the rest since there are not many uncertain ones. In this case, too, all methods generate more or less the same results, as shown in Figure 12. 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 10 and 11.

Finally Figure 13 show a summary of how Nostradamus behaves under different uncertainty measures. As one might expect, the more uncertainty results in less accurate prediction capability, and hence less power to generate a minimum area floorplan.

In conclusion, Nostradamus can be used when between 30% and 70% of the modules have deterministic dimensions. If more modules are uncertain, then the method cannot hide the uncertainty of the rest. If less modules are uncertain, then our method is not significantly better than traditional methods since there are not many uncertain modules to hide.

VI. EXTENSION OF NOSTRADAMUS

In the previous sections, we used different and independent distribution lists $W_i$ and $H_i$ (see Section II) to model the uncertainty in the dimensions of the modules. This model is simple, however, it is not typically used. The
Similarly, we can combine the distribution lists of two modules with distribution lists $\mathcal{L}_1$ and $\mathcal{L}_2$ vertically by combining the distribution lists as:

$$\mathcal{L}_{12V} = \{(w_{1i} + w_{2j}, \max(h_{1i}, h_{2j}), p_1, p_2) | (w_{1i}, h_{1i}, p_1) \in \mathcal{L}_1 \text{ and } (w_{2j}, h_{2j}, p_2) \in \mathcal{L}_2\}$$

To avoid exponential growth of the size of the lists, one can delete any triplet with probability less than a small constant. In our experiments, we did not notice significant slow down on the running time of the algorithm, so we did not prune any triplets.

We have run similar experiments to those described in Section V-C to compare the extended Nostradamus with traditional methods. The results are shown in Figures 14 – 17. The results are not as good as those of Nostradamus, but still show that the extended Nostradamus generates floorplans with better quality than the traditional methods.

**Fig. 11.** Ratio of the actual floorplan size of the traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 30%.

**Fig. 12.** Ratio of the actual floorplan size of the traditional methods to the actual floorplan size of Nostradamus. Level of uncertainty is 10%.

width and the height of the modules cannot assume any combination, because their area might be fixed or within a specific region. The model we used so far cannot address such issues. A more realistic model is the triplet list $\mathcal{L}_i = \{(w_{ij}, h_{ij}, p_i)\}$ (see Section II).

To enable Nostradamus to handle such distributions, we have used operations similar to $\oplus$ and $\ominus$. When clustering two modules with distributions $\mathcal{L}_1$ and $\mathcal{L}_2$ horizontally, we combine the distribution lists as:

$$\mathcal{L}_{12H} = \{\max(w_{1i}, w_{2j}), h_{1i} + h_{2j}, p_1, p_2) | (w_{1i}, h_{1i}, p_1) \in \mathcal{L}_1 \text{ and } (w_{2j}, h_{2j}, p_2) \in \mathcal{L}_2\}$$

**Fig. 13.** The ratio of actual floorplan size of Nostradamus with different percentages of uncertainty to 10% uncertainty. In all cases, $\lambda = 0.7$.

**Fig. 14.** Ratio of the actual floorplan size of the traditional methods to the actual floorplan size of Nostradamus using $\mathcal{L}_1$ distributions. Level of uncertainty is 100%.

**Fig. 15.** Ratio of the actual floorplan size of the traditional methods to the actual floorplan size of Nostradamus using $\mathcal{L}_1$ distributions. Level of uncertainty is 50%.

**VII. Conclusion and Future Work**

We have proposed a floorplanner that is capable of dealing with uncertainty of modules’ dimensions. It generates a compact floorplan by anticipating the final width/height 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.
We also studied the correlation between the input variance and the floorplan's dimension variance. Experimental results show that the number of certain modules plays an important role in helping the system generate reliable floorplans.

We should extend Nostradamus so that it can take as input list of uncertain netlist as well as uncertain modules. In fact, we have done some research in [1], [2] where the information on the modules is complete, but the netlist data is probabilistic.

As another extension, we might be able to deal with continuous distribution functions as an alternative to the discrete model that we used. Handling continuous probability distribution functions is more efficient and will probably speed up the floorplanning process dramatically.

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 an open problem.

The source code of Nostradamus can be obtained by contacting the authors.

ACKNOWLEDGEMENTS

The authors wish to thank Dr. Naveed Sherwani of Intel for many useful discussions on this problem.

REFERENCES

Sam Jung Kim received his B.S. degree in Computer Science from Northwestern University in 1998. From 1996 to 1998, he worked at VLSI Algorithmic Design Labs in Electrical and Computer Engineering department in Northwestern. His research interests include physical design, algorithmic design, and VLSI architecture. He is currently working for Monterey Design Systems in Sunnyvale, CA.

Majid Sarrafzadeh received his B.S., M.S. and Ph.D. in 1982, 1984, and 1987 respectively from the University of Illinois at Urbana-Champaign in Electrical and Computer Engineering. He joined Northwestern University as an Assistant Professor in 1987. Since 1997 he has been a Professor of Electrical Engineering and Computer Science at Northwestern University. His research interests lie in the area of VLSI CAD, design and analysis of algorithms and VLSI architecture. Dr. Sarrafzadeh is a Fellow of IEEE for his contribution to "Theory and Practice of VLSI Design". He received an NSF Engineering Initiation award, two distinguished paper awards in ICCAD, and the best paper award for physical design in DAC for his work in the area of Physical Design. He has served on the technical program committee of numerous conferences in the area of VLSI Design and CAD, including ICCAD, EDAC and ISCAS. He has served as committee chairs of a number of these conferences, including International Conference on CAD and International Symposium on Physical Design. He was the general chair of the 1996 International Symposium on Physical Design.

Professor Sarrafzadeh has published approximately 150 papers, is a co-editor of the book "Algorithmic Aspects of VLSI Layout" (1994 by World Scientific), co-author of the book "An Introduction to VLSI Physical Design" (1996 by McGraw Hill), and the author of an invited chapter in Encyclopedia of Electrical and Electronics Engineering in the area of VLSI Circuit Layout. This is planned for publication in 1997 by John Wiley & Sons, Inc. Dr. Sarrafzadeh is on the editorial board of the VLSI Design Journal, co-editor-in-chief of the International Journal of High-Speed Electronics, and an Associate Editor of IEEE Transactions on Computer-Aided Design. Dr. Sarrafzadeh has collaborated with many industries in the past ten years including IBM and Motorola and many CAD industries.