A Sequence Pair-Based Non-Slicing Floorplanner for Uncertain Designs

Jer-Ming Hsu
Department of computer and Information Science
National Chiao Tung University
Hsinchu 300, Taiwan

Yao-Wen Chang (corresponding author)
Department of Electrical Engineering and Graduate Institute of Electronics Engineering
National Taiwan University
Taipei 106, Taiwan
Email: ywchang@cc.ee.ntu.edu.tw
Tel: +886-2-2364-1972
Fax: +886-2-2363-8247

Abstract

Floorplanning is an important step in an early phase of VLSI design. For faster design convergence, there is an urgent need to start floorplanning as early as possible, even when not all modules are designed. Therefore, it is desirable to consider floorplanning with uncertainty to obtain a compact and reliable floorplan when the dimensions and interconnections of modules are not fully determined. In this paper, we propose a sequence-pair based floorplanner for uncertain designs. We first derive a probability-based longest path algorithm for a constraint graph to evaluate the enclosing width and height distributions. We then use simulated annealing to minimize the expected value and the variance of the enclosing area. Our floorplanner with uncertain data can consider non-slicing floorplans and interconnection cost which are not addressed before, and can generate a reliable and compact floorplan. Experimental results show that with 10%–30% uncertainty we have 98% confidence that the area derivation is within only 2% and the average dead space is less than 7%.

1 Introduction

Floorplanning is one of the most important steps in an early phase of VLSI design. Some critical physical parameters, such as area, delay, and power, can be approximate after this stage. For faster design convergence, there is a urgent need to obtain a floorplan as early as possible. As mentioned in [4], by generating a floorplan early in the design process, i.e., when some modules have not been
completely designed yet, the total design time can be decreased since some time-consuming processes can only be performed after a floorplan is obtained. Therefore, it is desirable to consider the problem of floorplanning with uncertainty to obtain a compact and reliable floorplan even when the dimensions and interconnections of modules are not fully determined.

In practice, some components must be designed from scratch, whereas some others are modifications of components from previous designs, and thus designers can specify some estimations about the final area of each incompletely designed module and its corresponding probability. The conventional floorplanning algorithms take a list of module information for which each module has a deterministic dimension. Therefore, these algorithms cannot handle the problem that some modules have multiple possible dimensions, represented by width and height probability distribution functions (PDF’s). Intuitively, the reasonable approach is to use the expected values of the width distribution and height distribution as width and height respectively. However, we will show later that the expected value method cannot generate a reliable floorplan. Nostradamus [4] is the first floorplanner that can cope with uncertain dimensions of modules, and is effective to obtain a compact floorplan. However, it is based on the slicing structure [14] which does not correspond to most real designs. Further, it does not consider interconnection cost which is a crucial metric for contemporary interconnect-driven VLSI design methodology.

In this paper, we propose a sequence-pair [7] based floorplanner for uncertain designs. Our floorplanner for uncertain data can consider non-slicing floorplans [1, 2, 3, 7, 9], and interconnection cost which are not addressed before, and can generate a reliable and compact floorplan. To cope with the uncertainty, we derive a probability-based longest path algorithm for a constraint graph to evaluate the enclosing width and height distributions. Besides, we use a distribution sampling technique to avoid the exponential growth of running time. After applying the PLP algorithm, we can obtain possible placements and interconnection costs. Our floorplanner with the estimated uncertain data can effectively generate a compact and reliable floorplan. By a reliable floorplan, we mean one whose area at the time it is generated does not differ significantly from its area when all modules are designed completely. Experiments on PLP show that with 10%–30% uncertainty, we have 98% confidence that the area derivation is within only 2% and the average dead space is less than 7%. Thus, PLP can generate a reliable and compact floorplan.

2 Problem Formulation

In this paper, we follow the formulation presented by [4]. Let \( M = \{ m_1, m_2, ..., m_n \} \) be a set of \( n \) rectangular modules. Each module \( m_i \in M \) is associated with a two tuple \( (W_i, H_i) \) which denotes a pair of height and width distribution. Each distribution list contains pairs of potential heights (or widths) of a module and their probabilities.

\[ \text{A formal definition of a reliable floorplan is given in Section 3.4.} \]
We have

\[ \mathcal{W}_i = \{ (w_{ij}, p_{w_{ij}}) | 1 \leq j \leq r_i \}, \]

\[ \mathcal{H}_i = \{ (h_{ij}, p_{h_{ij}}) | 1 \leq j \leq s_i \}. \]

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 \(w\) and \(h\) respectively. Therefore, each module \(i\) has a distribution list of such triplets, represented by

\[ \mathcal{L}_i = \{ (w_{ij}, h_{ij}, p_{ij}) | 1 \leq j \leq k_i \}. \]

However, directly using \(\mathcal{L}_i\) in our model is too cumbersome. Thus, we explain how to transform \(\mathcal{L}_i\) into independent \(\mathcal{W}_i\) and \(\mathcal{H}_i\) and show that the combinations of \(\mathcal{W}_i\mathcal{H}_i\) still correspond to the area distribution of module \(i\). By \(\mathcal{L}_i\), we can easily obtain an area distribution, \(\mathcal{A}_i = \{ (a_{ij}, p_{ij}) | 1 \leq j \leq k_i \}\), where \(a_{ij} = w_{ij}h_{ij}\). An element \((w_{ij}, h_{ij}, p_{ij})\) of \(\mathcal{L}_i\) is divided into a two tuple \((w_{ij}, p_{w_{ij}})\) for \(\mathcal{W}_i\) and \((h_{ij}, p_{h_{ij}})\) for \(\mathcal{H}_i\), where \(p_{h_{ij}}\) is a chosen value to make \(\mathcal{W}_i\mathcal{H}_i\) close to \(\mathcal{A}_i\). To determine \(p_{h_{ij}}\), we can interpolate the expected area \(h_{ij}E[\mathcal{W}_i]\) on the area distribution \(\mathcal{A}_i\). Therefore, we attain the corresponding \(\mathcal{W}_i\) and \(\mathcal{H}_i\) for each \(\mathcal{L}_i\) and use them throughout this paper.

The objective of floorplanning with uncertain designs is to minimize the expected value of area \(\mathcal{A}_{tot}\) and wirelength \(\mathcal{W}_{tot}\) induced by the assignment of the topology of \(m_i\)’s, where \(\mathcal{A}_{tot}\) is a probability distribution measured by possible enclosing rectangles of placement and \(\mathcal{W}_{tot}\) is also a probability distribution measured by the summation of half of bounding boxes of the center-to-center interconnections among all modules.

### 3 Floorplanning with Uncertain Designs

In this section, we shall present our non-slicing floorplanning algorithm for uncertain designs. As mentioned earlier, our algorithm adopts the sequence pair representation for which constraint graphs are constructed to evaluate the cost of a placement. In [7], the longest path algorithm is applied to a directed acyclic constraint graph to compute the enclosing dimension. To cope with uncertainty, we propose a probability-based longest path (PLP) algorithm to evaluate the enclosing width and height distributions. In the following subsections, we detail the algorithm and techniques to reduce the run-time complexity.

#### 3.1 Probability-based Longest Path (PLP) Algorithm

For uncertain designs, since a module may have many possible dimensions, the weight of the corresponding node in a horizontal (vertical) constraint graph gives the width (height) distribution of the module. Figure 1(b) shows the probability-based horizontal constraint graph for the placement of Figure 1(a). An edge \(n_i \rightarrow n_j\) in a horizontal (vertical) constraint graph denotes that module \(m_i\) is on the left (bottom) side of module \(m_j\). Since the weight of a node is a distribution, there could be many possible
longest paths. Our objective is to obtain the enclosing width (height) distribution of a floorplan which is corresponding to the length distribution of longest paths of horizontal (vertical) constraint graph.

Thus, we propose the probability-based longest path (PLP) algorithm to estimate the length distribution of longest paths. To describe the algorithm, we need two probability distribution operations: distribution merge and distribution addition. For simplicity, we use the notation for horizontal constraint graphs while vertical part is similar. Let $\mathcal{L}_i$ and $\mathcal{R}_i$ denote node $n_i$’s left and right boundary random variable respectively ($1 \leq i \leq n$.) The distribution merge operation is used to find $\mathcal{L}_i$, while the distribution addition operation is used to find $\mathcal{R}_i$. The distribution addition operation was first presented in [4], which is defined as follows:

$$
\mathcal{D}_1 \oplus \mathcal{D}_2 = \{ (d_{i1} + d_{j1}, p(d_{i1})p(d_{j1})) | (d_{i1}, p(d_{i1})) \in \mathcal{D}_1, (d_{j1}, p(d_{j1})) \in \mathcal{D}_2 \} 
$$

(1)

The distribution list of $\mathcal{D}_1 \oplus \mathcal{D}_2$ consists of elements which are pairwise “addition” of the elements from the two distribution list $\mathcal{D}_1$ and $\mathcal{D}_2$. Thus, it is trivial that $\mathcal{R}_i$ can be derived by

$$
\mathcal{R}_i = \mathcal{L}_i \oplus \mathcal{W}_i,
$$

(2)

where $\mathcal{W}_i$ denotes the width distribution of node $n_i$.

In the following subsection, we detail the distribution merge operation for $\mathcal{L}_i$.

### 3.1.1 Distribution Merge Operation

In the above section, we have shown how to obtain right boundary distribution by distribution addition operation.

To derive the left boundary distribution of node $n_i$, we define distribution merge operation that manipulates the set of all right boundary distributions of incoming nodes of $n_i$. In the following, we shall use a example to elucidate the operation rather directly deriving the equation. As shown in
Figure 1(b), the node $a$ has incoming node $e$ and $c$ whose right boundary distributions are $\mathcal{R}_e = \{(5, 0.2), (6, 0.6), (7, 0.2)\}$ and $\mathcal{R}_c = \{(3, 0.3), (5, 0.6), (6, 0.1)\}$. Thus, $\mathcal{L}_a$ has three possible values 5, 6, and 7 where $\mathcal{L}_a = 3$ is impossible since the node $e$ will dominate node $c$ if $\mathcal{R}_c = 3$. The probability that the longest path will pass through node $e$ to node $a$ is denoted as $P_{e\rightarrow a}(x)$. It is intuitive that $P_{e\rightarrow a}(x) = P(\mathcal{R}_e = x)P(\mathcal{R}_c < x)$ in that node $e$ should dominate node $c$. Therefore, the probability of the left boundary $x$ of node $a$ is equal to sum up $P_{e\rightarrow a}(x)$ and $P_{c\rightarrow a}(x)$. For example, the probability of $\mathcal{L}_a = 7$ equals the summation of $P(\mathcal{R}_e = 7)P(\mathcal{R}_c < 7)$ and $P(\mathcal{R}_e = 7)P(\mathcal{R}_c < 7)$ which is $0.2 \times 1 + 0 \times 1 = 0.2$. The formal definition of the probability that the longest path will pass through $n_i$ to $n_k$ is defined as follows:

$$P_{n_i\rightarrow n_k}(x) = P(\mathcal{R}_i = x) \prod_{j \in I_{n_k}}^{n_j > i} P(\mathcal{R}_j < x) \prod_{j \in I_{n_k}}^{n_j < i} P(\mathcal{R}_j \leq x),$$

where $I_{n_k}$ denotes the set of all $n_k$’s incoming nodes. $\mathcal{R}_j \leq x$ of the above equation is used to compensate the probability when some nodes have same right boundary.

To represent the left boundary distribution of $n_k$, $\mathcal{L}_k$, we should merge all possible longest path through $n_i$ to $n_k$ for each $n_i \in I_{n_k}$ into single random variable $\mathcal{L}_k$ as follows:

$$P(\mathcal{L}_k = x) = \sum_{n_i \in I_{n_k}} P_{n_i\rightarrow n_k}(x).$$

Let $\odot$ denote the merge operation of a set of distributions.

$$\mathcal{L}_k = \odot(\mathcal{R}_i, \ldots, \mathcal{R}_j),$$

where $\mathcal{R}_i, \ldots, \mathcal{R}_j$ denote the set of right boundary distributions for all $n_k$’s incoming nodes.

$\mathcal{L}_k$ generated by the distribution merge operation is a legal random variable, i.e., the summation of its probabilities equals 1. We have the following property.

**Property 1** Random variable is closed under the distribution merge operation.

3.1.2 The Algorithm

The constraint graph constructed directly from a sequence pair is a transitive closure. The transitive closure implies that the graph contains many redundant edges which can be replaced by other two or more edges. (See Figure 2 for a transitive closure in which a thin edge denotes a redundant edge.) Too many redundant edges may significantly slow down our PLP algorithm. Thus, we develop a transitive edge reduction algorithm as in Figure 3 for a directed acyclic graph to remove the redundant edges. The algorithm first traverses each node $s$ in increasing topological order, and then selects a node $t$ connected

\[\text{The transitive closure of a graph } G \text{ is defined as the graph } G' = (V, E'), \text{ where } E' = \{n_i \rightarrow n_j \mid \text{there is a path from } n_i \text{ to } n_j \text{ in } G\}.\]
to $s$ in decreasing topological order. Notably the edge traverse order is critical to remove all redundant edges. Then, iteratively deletes an edge $s \rightarrow t$ if there exist two edges $s \rightarrow k$ and $k \rightarrow t$ for some node $k$.

![Graph](image)

Figure 2: A transitive closure in which a thin edge denotes a redundant edge.

<table>
<thead>
<tr>
<th>Algorithm: Reduce_Transitivity($G$)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> $G$—acyclic directed graph</td>
</tr>
<tr>
<td><strong>Output:</strong> $G$</td>
</tr>
<tr>
<td>1 foreach node $s$ in $G$ with ascending topology order</td>
</tr>
<tr>
<td>2 $A \leftarrow$ outgoing nodes of $s$;</td>
</tr>
<tr>
<td>3 $A' \leftarrow$ sorting $A$ by descending topology order;</td>
</tr>
<tr>
<td>4 foreach node $t$ in $A'$</td>
</tr>
<tr>
<td>5 foreach node $k$ in $A$</td>
</tr>
<tr>
<td>6 if($t \neq k$ and $s \rightarrow k$ and $k \rightarrow t$)</td>
</tr>
<tr>
<td>7 erase_edge($s \rightarrow t$, $G$);</td>
</tr>
<tr>
<td>8 break;</td>
</tr>
</tbody>
</table>

Figure 3: The transitive edge reduction algorithm for a directed acyclic graph.

After applying the transitive edge reduction algorithm to remove redundant edges, we use the PLP algorithm to evaluate the distribution of the longest path length. Given a weighted, directed acyclic graph $G = (V, E)$ with source $S$, target $T$, and the weight function $w : V \rightarrow \mathcal{W}$ (where $\mathcal{W}$ denotes the width random variable), the PLP algorithm returns the length distribution of longest paths. The algorithm traverses each node in increasing topological order. For each node $n_k$, we apply the distribution merge and addition operations to obtain $\mathcal{L}_k$ and $\mathcal{R}_k$. Consequently, since target $T$ is the last node of graph $G$, $\mathcal{L}_T$ represents the length distribution of longest paths of graph $G$.

### 3.1.3 Sampling Distribution

Both distribution superposition and addition operations might cause an exponential growth in the number of distribution elements. Even though we usually run the PLP algorithm off-line, it is not tolerable if the exponential growth occurs frequently. To relieve the situation, we can replace those elements whose probabilities are negligible with the expected value and the corresponding summation of probability.
Algorithm: Prob_Longest_Path($G$)
Input: $G$—acyclic directed graph
Output: $D$—random variable
1: $S$ is the source node of $G$
2: $T$ is the target node of $G$
3: $I n_k$ is a set of incoming nodes of $n_k$
4: $G = \text{Top}_\text{Sort}(G)$;
5: $G = \text{Reduce}_\text{Transitivity}(G)$;
6: foreach node $n_k$ in $G$ with ascending topology order
7:  if ($S \not\rightarrow n_k$)
8:      $n_i, \ldots, n_j \in I n_k$
9:      $\mathcal{L}_k = \oplus(\mathcal{R}_i, \ldots, \mathcal{R}_j)$;
10:  if ($T = n_k$)
11:      $D = \mathcal{L}_k$;
12:     return $D$
13: else
14:    $\mathcal{R}_k = \mathcal{L}_k \oplus W_k$;

Figure 4: The PLP algorithm.

More aggressively, we can use a *uniform sampling* technique to divide a distribution into $k$ clusters, where the elements in a cluster have similar values and their summation of probability is equal to $1/k$. By using a larger value of $k$, we can get a more accurate result; however, it is a tradeoff between quality and run-time. Our empirical results indicate that $k = 20$ is usually enough to get accurate results close to those without sampling. With the sampling technique, we can show that our PLP algorithm runs in $O(n^2k^2)$ time, avoiding the expensive computation.

3.2 Estimation of Interconnection

The wirelength is usually estimated by the summation of half of bounding boxes of the center-to-center interconnection multiplied by their connectivity among all modules. The half of bounding box can divided into horizontal and vertical components. Since the method to compute the horizontal and vertical components of wirelength is the same, we discuss the horizontal part only. After applying the PLP algorithm on the horizontal constraint graph, we can obtain the right and left boundary distributions for each module, denoted by $\mathcal{R}$ and $\mathcal{L}$ respectively. We can derive the center distribution $\mathcal{C}$ by $\mathcal{C} = (\mathcal{R} \oplus \mathcal{L})/2$. The *distance distribution* between $\mathcal{C}_1$ and $\mathcal{C}_2$ is defined as follows:

$$C_1 \oplus C_2 = \{ (|d_{1i} - d_{2j}|, p(d_{1i})p(d_{2j})) | (d_{1i}, p(d_{1i})) \in C_1, (d_{2j}, p(d_{2j})) \in C_2 \}$$  \hspace{1cm} (6)$$

We use the expected value of $C_1 \oplus C_2$ as the estimated distance between two modules. Thus, the horizontal component of wirelength is the summation of the horizontal center-to-center distance among modules and is defined as follows:
\[ W_H = \frac{1}{2} \sum_{i,j} c_{ij} E[C_i \otimes C_j], \] 

(7)

where \( c_{ij} \) is the connectivity between module \( m_i \) and module \( m_j \) and \( E[C_i \otimes C_j] \) denote the expected value of \( C_i \otimes C_j \).

The vertical component of wirelength can be estimated in the same way. The total wirelength \( W \) is \( W_H + W_V \).

3.3 The Overall Algorithm

We develop a simulated annealing based algorithm [5] using the sequence pair representation for non-slicing floorplan design with uncertain data. Given an initial solution represented by a sequence pair, the algorithm perturbs the sequence pair to a new one. Then the algorithm applies the PLP algorithm to evaluate the \( W \) and \( H \) distributions. The cost function during the annealing process can be \( E[|W|]E[|H|] \). However, we not only want to minimize the area, but also to optimize its wirelength. The wirelength is evaluated by \( W_H + W_V \). Furthermore, obtaining a reliable floorplan is critical. Thus, we add variance of area and wirelength to the cost function and try to minimize both the expected value and the variance of area and wirelength. The variance of area and wirelength is defined respectively as follows:

\[ \Delta Area_{W,H} = \sigma_W \sigma_H + \mu_H \sigma^2 + \mu_W \sigma^2, \] \hspace{1cm} (8)

\[ \Delta Wire_{W,H} = \sigma^2 + \sigma^2, \] \hspace{1cm} (9)

where \( \sigma \) and \( \mu \) denote the standard derivation and mean of a distribution respectively.

Thus, we use the following cost function for simulated annealing:

\[ cost = \lambda \left[ \beta \frac{E[|W|]E[|H|]}{Area_{norm}} + (1 - \beta) \frac{\Delta Area_{W,H}}{\Delta Area_{norm}} \right] + \] \hspace{1cm} (10)

\[ (1 - \lambda) \left[ \beta \frac{W_H + W_V}{Wire_{norm}} + (1 - \beta) \frac{\Delta Wire_{W,H}}{\Delta Wire_{norm}} \right], \] \hspace{1cm} (11)

where \( \lambda \) is a user specified parameter to control the tradeoff between the area and the wirelength of a floorplan, and \( \beta \) is to control the tradeoff between the expected value and the variance of area and wirelength. In addition, the cost function is normalized by \( Area_{norm}, \Delta Area_{norm}, Wire_{norm}, \) and \( \Delta Wire_{norm} \) which are estimated maximum values respectively.

3.4 Reliable Floorplan

A reliable floorplan should have a small variance of area and wirelength. In the above section, we have shown to use the simulated annealing algorithm to minimize the expected value and variance of area and wirelength simultaneously. To show how it perform well in the real environment, we start simulations by random generating each module’s dimension accroding its distribution, then fit those modules into the
topology of floorplan determined by the PLP or sequence-pair floorplanner. Thus, the actual area and wirelength of the simulation is obtained. By perform the simulation \( n \) times, we have \( X_1\ldots X_n \) statistics as sample area or sample wirelength of the floorplan. Based on the statistics, the reliability of a floorplan can be calculated. The reliability \( r \) of a floorplan is defined as we have \( r \) confidence that the area or wirelength deviation of the floorplan is within \( d = 2\% \) of the area. We use Chebyshev’s inequality to facilitate the definition.

If \( X \) is a random variable with finite sample mean \( \mu \) and sample variance \( \sigma^2 \), then for any value \( d > 0 \),

\[
P\{ |X - \mu| \geq d\mu \} \leq \frac{\sigma^2}{d^2\mu^2} = 1 - r
\]

\[
P\left\{ \left| \frac{X - \mu}{\mu - 1} \right| < d \right\} > 1 - \frac{\sigma^2}{d^2\mu^2} = r.
\]

Then, we have

\[
r = 1 - \frac{\sigma^2}{d^2\mu^2} \quad (12)
\]

For example, the areas of a floorplan obtains by simulations has \( \mu = 3000 \) and \( \sigma^2 = 100 \), then the area reliability is \( r = 1 - 100(0.02 * 3000)^2 = 97.2\% \).

4 Experimental Results

Our experiments use the same data sets as the experiments of Nostradamus used. However, the detail experimental results of Nostradamus are not available to us, and thus we cannot fairly compare our results with Nostradamus. But, the non-slicing structure in general has less area usage than the slicing structure. Sinice PLP can handle non-slicing structure, we believe PLP can generate more compact floorplans than Nostradamus which is a slicing floorplanner.

In following, we will compare our results with the expected value method which uses fill the width (height) of a module by the expected value of the width (height) of the module. To evaluate how well PLP might work in practice, we simulate the real environment as all modules have been designed completely by generating a fix dimension of each module according to its width and height distribution. Then, the enclosing area and total wirelength is obtained by fitting those real modules to the topology determined by PLP or the expected value method. By repeating a number of simulations, the sample area and wirelength are obtains. Then, we can use the sample area and wirelength evaluate the reliability of a floorplan and average area and wirelength. Thus, we will compare results by the average area, average wirelength, area reliability, and wire reliability of a floorplan.

An 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. To demonstrate the performance of PLP to cope with uncertainty effectively, we compare the results with different percentage of modules that have
probabilistic dimensions. Figure 5 shows the ratio of average area obtained by expected value method to average area obtained by PLP. Noteably, the ratio approaches to 100% for 10%, 30%, 50%, and 100% uncertainty. The result indicates that PLP has similar average area performance with the expected value method for all range uncertainty.

Figure 6-9 shows the reliability of PLP and the expected value method with different uncertainty values ranging from 10% to 100%.

For uncertainty value ranging from 10% to 30%, PLP has almost 100% area reliability among most cases in contrast to the expected value method that has less 10% reliability in a1, a2, a3, and f4 data sets. For uncertainty value ranging from 50% to 100%, PLP become weak to attain above 80% reliability, but still has higher reliability than the expected value method over all data sets. In conclusion, PLP can be used when between 10% to 30% of modules have probabilistic dimensions to generate a reliable floorplan.

![Graph showing the ratio of the area of the expected method to the area of PLP method with different uncertainty factors.](image)

Figure 5: Ratio of the area of the expected method to the area of PLP method with different uncertainty factors.

5 Concluding Remark

We have proposed a non-slicing floorplanner, called PLP, that can deal with uncertain dimensions of modules. Experimental result have shown that the PLP have powerful ability to hide the uncertainty and can generate quiet reliable and compact floorplans. The capability of the PLP shows its promise in generating a floorplan in early design process. We propose to explore the floorplanning problem with time consuming tasks, such as rectilinear and buffer-block planning for interconnect-driven floorplanning in the future.
**Reliability of a floorplan with 10% uncertainty**

![Graph showing comparison between PLP and expected value methods]

Figure 6: Comparison of reliability of area derivation within 2% between the PLP method and the expected value method. Level of uncertainty is 10%.

**References**


Figure 7: Comparison of reliability of area derivation within 2% between the PLP method and the expected value method. Level of uncertainty is 30%.


Figure 8: Comparison of reliability of area derivation within 5% between the PLP method and the expected value method. Level of uncertainty is 50%.

Figure 9: Comparison of reliability of area derivation within 5% between the PLP method and the expected value method. Level of uncertainty is 100%.