UNIVERSITY OF CALGARY

A Pre-Placement Individual Net Length Estimation Model and an Application for Modern Circuits

by

Amin Farshidi

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

August, 2011

UNIVERSITY OF
CALGARY

Abstract

A-priori individual interconnect length estimates can be used to make placement efficient and reduce delay and power consumption in a circuit. However, finding lengths of individual interconnects before their terminals have been placed can be a daunting task. In this thesis, the main characteristics that need to be considered while designing an individual a-priori length estimation technique for today's integrated circuits are discussed. Then, a technique for estimating individual interconnect lengths for mixed-size circuits based on Radial Basis Functions (RBFs) is proposed. The advantage of using RBFs, is that the given data do not need to be fitted into a pre-defined model such as a polynomial. In this thesis, a method is proposed to calculate a suitable variance parameter of the RBFs. An application of the length estimator is proposed where a predictor-corrector framework for clustering that can be used to improve the results of placement is implemented.

Acknowledgements

I would like to thank all people who helped me in production of this thesis. First, I would like to express my thanks to my supervisor, Dr. Behjat, for her suggestions in making this thesis. I also thank my co-supervisor, Dr. Westwick, for his comments and suggestions. In addition, I appreciate the valuable comments from the committee members. I wish to thank all of my previous teachers who helped me a lot in my education.

I would like to thank my family which is the most important factor of my success. I wish to thank my mom and dad who always guide me into the way of success. To my wife, Mehrnaz, thanks for making a happy environment for me during the writing of this thesis. Finally, thanks to my colleagues, Logan, Yangyang, Bardia and Aysa, for helping me with the proofreading of the thesis.

Table of Contents

List of Tables

List of Figures

List of Terms

Acronyms:

BC	:	Best-Choice Clustering
CCR	:	Cell Cluster Ratio
CSU	:	Constructive Selective Uniform
EC	:	Edge Coarsening
ESC	:	Edge Separability-Based Clustering
FC	:	FirstChoice Clustering
FGC	:	Fine Granularity Clustering
FM	:	Fiduccia and Mattheyses
HC	:	Hyperedge Coarsening
HEM	:	Heavy-Edge Matching
IC	:	Integrated Circuit
MHC	:	Modified Hyperedge Coarsening
RBF	:	Radial Basis Function
VLSI	:	Very Large Scale Integration

Sub-scripts:

i	:	Index of cells and nets, centers and RBFs
j	:	Index of cells and nets
k	:	Index of cells and nets
l	:	Index of external nets

Scalars:

a	:	Coefficient
$aveA_{stdCell}$	:	Average area of standard cells in a circuit
b	:	Coefficient
$base$	:	Base length for each net configuration
c	:	Center of an RBF
$cell$	:	Cell
Clu	:	Cluster
d_c	:	Grid distance
e	:	Edge
G_h	:	Hypergraph
h_{std}	:	height of a standard cell
k_{NC}	:	A constant for finding cluster scores in NC
k_{PinEC}	:	A constant for finding connectivity in PinEC clustering
LB_{Clu}	:	Lower bound on the cluster size

M	:	Number of actual data points
N	:	Number of centers
n_{center}	:	Number of uniform centers
n_{grid}	:	Number of center grids
n_{var}	:	Number of variables
net	:	Net
num_{d2PL}	:	Number of possible layout configurations for degree-two net
num_{d2tot}	:	Total number of degree-two nets in a circuit
num_{ext}	:	Number of external connections
$num_{totcell}$	:	Total number of cells in a circuit
P	:	Probability
p_d	:	Polynomial degree
p_x	:	Location of a pin on horizontal axis
p_y	:	Location of a pin on vertical axis
$pclu$	:	Potential cluster
per_{dec}	:	Percentage of nets with decreased length after clustering
per_{inc}	:	Percentage of nets with increased length after clustering
Pr	:	Measure of overall properties of a circuit
r	:	Radius used in CSU center placement method
$ratio_{inc-dec}$	:	Ratio of nets with increased and decreased lengths
SF	:	Measure of safeness of a cluster
UB_{Clu}	:	Upper bound on the cluster size
UF	:	Utilization factor
UF_h	:	Horizontal utilization factor
UF_v	:	Vertical utilization factor
w	:	Weight of an RBF
w_{ave}	:	Average width of a set of cells
x	:	Model variable
α	:	A factor for considering fixed cells effects
λ	:	A factor for finding the new base length
σ	:	Standard deviation

Sets:

E	:	Set of all nets
R	:	A restricted subset of nets
S_C	:	A typical set of cells
$set_{cellNbr}$	:	An arbitrary set of neighbor cells of a cell
set_{nbr}	:	Set of cells adjacent to a cell
V	:	Set of all vertices

Vectors:

$\mathbf{c}$	:	Vector of coordinates of a center

$\mathbf{c_p}$	:	Vector of coordinates of a potential center
$\ell_{\mathbf{act}}$	:	Vector of actual after placement lengths
$\ell_{\mathbf{est}}$	:	Vector of estimated lengths
$\mathbf{w}$	:	Vector of weights of RBFs
$\mathbf{x}$	:	Vector of model variables
$\mathbf{y_{act}}$	:	Vector of actual data points
$\mathbf{y_{est}}$	:	Vector of function estimates

Matrices:

$\mathbf{C}$	:	Matrix of center locations of RBFs
$\mathbf{C_p}$	:	Matrix of potential centers of RBFs
$\mathbf{X}$	:	Matrix of model variables
$\mathbf{\Phi}$	:	Matrix of RBFs

Functions:

$2ndlvl(\cdot)$	:	Second level effect variable of a net
$2PinCong(\cdot)$	:	Degree-two congestion metric of a net
$area(\cdot)$	:	The area of a cell
$attint_{nc}(\cdot)$	:	The variable $attint_{nc}$ of an edge
$base_L(\cdot)$	:	Base length variable of a net
$comN(\cdot)$	:	Set of common nets of a set of cells
$compL_{clu}(\cdot)$	:	Comparison of after and before clustering lengths of a net
$conn(\cdot,\cdot)$	:	The connectivity between two cells
$CorrCoef(\cdot,\cdot)$	:	Correlation coefficient between two sets
$COV(\cdot,\cdot)$	:	Covariance of two sets
$d(\cdot)$	:	Distance of a data point to a potential center
$dg(\cdot)$	:	Degree of a net or edge
$estAveL_h(\cdot)$	:	Estimated mean of a group of nets
$estVarL_h(\cdot)$	:	Estimated variance of a group of nets
$f(\cdot)$	:	A function that approximates the expected length increase of a net
f_B	:	A specific third-order polynomial function
$h(\cdot)$	:	The height of a cell
$ispl(\cdot)$	:	Intrinsic shortest path length of a net
$\ell_{act}(\cdot)$	:	The actual after placement length of a net
$\ell_{act_{AC}}(\cdot)$	:	The actual length of a net after clustering
$\ell_{act_{PC}}(\cdot)$	:	The actual length of a net before clustering
$\ell_{est}(\cdot)$	:	The estimated length of a net
$\log(\cdot)$	:	The natural logarithm function
$macroBase_L(\cdot)$	:	Macro base length of a net
$max\{\cdot,\cdot\}$	:	The maximum of two numbers
$mc(\cdot,\cdot)$	:	Mutual contraction between two cells

$mc_g(\cdot)$	:	Mutual contraction of a group of connected cells
$min(\cdot)$	:	The minimum of a set of numbers
$N_{2oth}(\cdot)$	:	The variable N_{2oth} of a net
$n_c(\cdot,\cdot)$	:	Number of polynomial coefficients
$Nettint_{nc}(\cdot)$	:	The variable $Nettint_{nc}$ of a net
$nld_{ratio}(\cdot)$	:	Ratio of after and before clustering lengths of a net
$num_C(\cdot)$	:	Number of cells in a cluster
$num_{cut}(\cdot)$	:	Number of nets cut by a cluster
$num_{d2nbr}(\cdot)$	:	The number of degree-two nets in the neighborhood of a net
$num_N(\cdot)$	:	Number of nets in a cluster
$num_{totnbr}(\cdot)$	:	The number of all nets in the neighborhood of a net
$qs(\cdot)$	:	A quadratically-growing sinusoidal function
$RBF_{NN}(\cdot)$	:	Radial basis function neural networks
$rspl(\cdot)$	:	Restricted shortest path length of two cells
$rspl_s(\cdot)$	:	Restricted shortest path length of set of cells
$score_{clu}(\cdot)$	:	The score assigned to a potential cluster
$score_{net}(\cdot)$	:	The score assigned to a net
$set_{2c}(\cdot)$	:	Set of cells in the second level neighborhood of a net
$set_c(\cdot)$	:	Set of cells connected to a net
$size(\cdot)$	:	The size of a cluster
$w(\cdot)$	:	The width of a cell
$wg(\cdot)$	:	Weight of a net, edge, cell or connection
$x_i(\cdot)$	:	A model variable of a net
$y_{est}(\cdot)$	:	Function estimate of a data point
$\phi(\cdot)$	:	A radial basis function
$\|\|\cdot\|\|$	:	Euclidean norm
$\cap$	:	Intersection
$\cup$	:	Union
Σ	:	The summation of a set of numbers
Π	:	The multiplication of a set of numbers

Definitions:

Adjacent (Neighbor)	:	Cells that are connected with at least one net
Cell	:	A circuit component, e.g. logic gate
Edge	:	An interconnect that only connects two cells
Macro cell	:	A cell that is more complex and larger than typical standard cells
Mixed-size design	:	A design that contains macro cells in addition to standard cells
Net (Hyperedge)	:	An interconnect that connects two or more cells
Net degree	:	Number of cells that are connected to the net
Net length	:	Length of wire needed for routing a net

Second level neighbor	:	A cell that can be reached from cells of a net by just one net
Standard cell design	:	A design that only contains standard cells
Standard cells	:	Cells at logic design level that have same heights
Total wire length	:	Sum of lengths of wires needed for routing a circuit

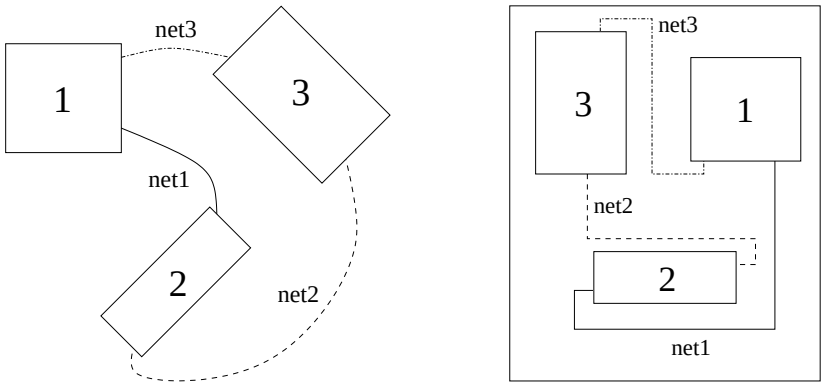
Chapter 1

Introduction

In today's Integrated Circuit (IC) designs, millions of circuit components are integrated in a small area, for example, Intel's Core i7 processor released in 2010 contains 2.79 million transistors in each mm^2 [3]. The design of these ICs is a complex process that includes determining the system specifications, such as performance, functionality and physical properties of the circuit. Physical design is one of the main steps in IC design where the physical properties such as the exact locations of components and wires of the circuit, are determined [4, 5].

The physical design includes three major stages: partitioning, placement and routing [4,5]. In the partitioning stage, the circuit, represented by a netlist, is divided into several partitions that are relatively independent, i.e. as few wires as possible connect them. In the placement stage, the exact locations of the circuit components are determined. Then, the paths for all of the nets of the circuit are determined in the routing stage. A more comprehensive introduction of these stages is presented in Appendix A.

Several approaches are also employed during physical design to improve the performance of these major stages. One such approach is clustering, where the size of the circuit is reduced to empower the placement to better handle modern IC designs. Another approach to improve the performance of the placement stage involves using an a-priori individual net length estimation technique, the main focus of this thesis. In this technique, the lengths of individual nets from circuit are estimated and this information is used to improve the quality of the placement. The data encountered in the physical design are highly non-linear and do not follow any well-known trends. This makes net length estimation a very complex task. In this thesis, it is proposed to use Radial



(a) The pre-placement circuit

(b) The after placement circuit

Figure 1.1: Example of net lengths before and after placement

Basis Functions (RBFs) to capture these non-linearities. In addition, several important characteristics of modern ICs that have not been considered in the existing net length estimation literature, are covered in the proposed technique. Then, an application of the proposed individual net length estimation technique in clustering is designed and implemented, and it is shown that the quality of the placement stage can be improved by applying the proposed technique.

The rest of this chapter is organized as follows: In Section 1.1, the a-priori individual net length estimation problem is introduced. In Section 1.2, the research motivations and major contributions of this thesis are presented. Finally, the thesis structure is presented in Section 1.3.

1.1 A-priori Individual Net Length EstimationTechnique

Net length estimation is defined as approximating the lengths of nets used for routing a circuit. A-priori net length estimation techniques are used to evaluate the net lengths before placement. In Figure 1.1, an example circuit is shown before and after placement. It can be seen that when the locations of the cells and their pins are determined, Figure

1.1(b), the calculation of net lengths is simple. However, it is very complicated to estimate the lengths of nets before the circuit components have been placed, Figure 1.1(a). For example, the nets net_1 and net_3 are long after placing the circuit since their pins are placed in distant positions.

A-priori individual net length estimation techniques aim to predict the lengths of the nets in a circuit before the locations of its cells have been determined. Therefore, it can be a complex task. Existing a-priori net length estimation techniques mostly output the total wire length or the average wire length of the nets [6,7]. However, there exist several techniques that estimate individual net lengths [1, 2, 8]. In these techniques, different characteristics of the nets and components of circuits are translated into several model variables. Then a predefined model such as a polynomial or exponential model that includes these model variables is used for net length estimation. The outputs of these techniques are the lengths of individual nets of the circuit.

1.2 Research Motivations and Contributions

As more and more components are placed in ICs, resources in the circuit, especially routing resources, are becoming more scarce. In addition, the design rules, i.e. rules that need to be observed so that the IC can be fabricated, are becoming numerous and increasingly complicated. In such an environment, it is not unusual for a design process to go on for days, before it is decided that the circuit can not be fabricated due to insufficient resources, usually routing resources, or violation of some design rules, such as timing, resulting in loss of valuable time and millions of dollars.

If some of these challenges are known prior to the physical design stage, proper action can be taken to avoid design rule violations. For example, if it is known a-priori that certain nets with tight timing are going to become too long, corrective actions for these

nets can be taken.

Individual a-priori net length estimation technique can be a valuable tool for this type of problem. However, most existing techniques do not produce reliable results since they try to fit a predefined model, such as a quadratic polynomial, to a highly non-linear set of data. Using a predefined basis for net length estimation results in inaccuracy.

In this thesis, it is proposed to use RBFs to estimate individual net lengths. RBFs provide the ability to capture the details of a set of highly non-linear given data with manageable model complexity [9]. In addition, most of the existing net length estimation techniques cannot handle mixed-size circuits with macro cells. However, today's mixed-size circuits include macro cells that are much bigger than standard cells [10]. These circuits also contain several cells that are not moveable, i.e. fixed cells. Furthermore, the existing length estimation techniques do not consider the type of placer that is going to be used to place the circuit, in their estimates. In this thesis, these shortcomings are considered and covered in the proposed estimation technique.

One potential application of the proposed pre-placement net length estimation technique is in clustering. Clustering algorithms are used to improve the performance of the placement stage [11]. However, sometimes clustering may have negative impacts on the individual net lengths. Therefore, the prediction and correction of these negative effects can help to improve the clustering and subsequently the placement performance. In this thesis, a predictor-corrector framework is proposed to capture and correct for the negative effects of clustering on individual net lengths. In the first step of this framework, a structure is considered to predict the nets that may be negatively affected during clustering. Then, a corrector step is designed to improve the results of clustering by avoiding those negative impacts.

The main contributions of this thesis are as follows:

- In this thesis, a new RBF-based pre-placement individual net length estimation

technique is proposed. For the first time in net length estimation, RBFs are applied to enable the net length estimation to better deal with the highly non-linear data encountered in physical design. The performance of the proposed technique is improved significantly when RBFs are applied.

- A new method is proposed and implemented for tuning the shapes of RBFs in order to properly adapt them to the given data. This method makes the RBFs effective for net length estimation since it calculates their variance parameter based on the input data.

- A new model variable which considers the effects that different placers have on the net lengths is proposed. The performance of different placement algorithms can affect the accuracy of the estimated lengths. Therefore, it is proposed to cover these effects by defining a new model variable.

- A new variable is defined to consider the effects of the presence of fixed cells. These cells exist in today's integrated circuits and affect the length of nets in their neighborhood, significantly. Considering these effects makes the proposed technique more effective for estimating the net lengths of modern circuits.

- A new predictor-corrector framework for clustering is proposed. The negative side effects of clustering algorithms on the individual net lengths are predicted in this framework. Then, these negative effects are corrected by further clustering the negatively-affected nets. The experiments validate the effectiveness of the proposed framework in improving the clustering results.

1.3 Thesis Structure

The rest of this thesis is organized as follows:

- **Chapter 2**

In this chapter, a comprehensive background on the existing interconnect length estimation techniques is presented. In addition, radial basis functions are introduced as effective tools for net length estimation.

- **Chapter 3**

Several existing clustering algorithms for circuit partitioning and placement are explained.

- **Chapter 4**

In this chapter, the first contributions of this thesis are presented. First, new model variables to consider the effects of different placers and the presence of fixed cells on the individual net lengths are proposed. These variables are used in net length estimation techniques to be proved to be effective. Then, an RBF-based net length estimation technique is proposed and implemented. A new method for tuning the shapes of the RBFs is suggested as well.

- **Chapter 5**

Another main contribution of this thesis is presented in Chapter 5. A predictor-corrector framework is proposed and applied to improve the results of clustering. The proposed framework is then validated by performing several experiments.

- **Chapter 6**

In this chapter, the thesis is summarized and the potential future research directions are suggested.

- **Appendix A**

In this appendix, further introduction to the physical design problem is given.

Chapter 2

Background: Net Length Estimation and Radial Basis Functions

In this chapter, backgrounds on net length estimation techniques and radial basis functions are presented. This chapter is organized as follows: In Section 2.1, the existing net length estimation techniques are studied. Then, in Section 2.2, Radial Basis Functions (RBFs) are discussed in detail. Finally, a summary is presented in Section 2.3.

2.1 Existing Net Length Estimation Techniques

2.1.1 Introduction

With the continuing growth of the amount of interconnects or nets in Integrated Circuits (ICs), circuit delay and power consumption are becoming highly affected by the net lengths. In addition, circuit area significantly depends on the total net length required for completing the connections of a circuit, since a major part of the physical space of the circuit is used for interconnection routing. Therefore, estimating the lengths of nets a-priori to the physical design is becoming an important tool in empowering designers in dealing with tough issues before the design is complete [6].

Several net length estimation techniques are proposed to provide insight about net length during or before physical design [1,2,6–8,12–24]. This insight helps the designer to improve the quality of placement and routing of a circuit. For example, if the estimated length for a specific net on a critical path is large, in the placement stage the spatial proximity of the cells in the net can be emphasized.

2.1.2 Net Length Estimation Types

There are two typical ways to classify the existing net length estimation techniques: the stage the estimation is performed at and the level of details of the estimates.

In the first classification, net length estimation techniques are distinguished in three groups depending on the physical design stage where the estimation happens at: *a-priori*, *on-line* and *a-posteriori* [14]. In the following, each one is briefly described.

A-priori net length estimation [1, 2, 6–8, 12, 13, 15–20, 24] is used to evaluate the net length before components of a circuit have been placed. These estimates may be used to obtain rough measures of routability. In addition, since estimation is performed before placement, the quality of placement results can be improved by clustering the circuit based on estimated lengths [24].

On-line net length estimation techniques, which happen during placement, are used when the estimated net lengths are desired during the placement stage. These estimates are used to stop the placement process early when it becomes clear that the process is resulting in a bad placement solution. In [14] an example of *on-line* net length estimation techniques is presented.

A-posteriori net length estimation techniques try to estimate the net lengths after the placement is obtained, but before the routing stage has been performed. These estimated lengths can be used to choose among several competing placement solutions [14]. In [21–23], several examples of *a-posteriori* net length estimation techniques are proposed.

In the other classification, net length estimation techniques are divided into three classes based on how detailed their outputs are, as *global*, *semi-individual* and *individual* net length estimation techniques.

Global techniques estimate the average length of all nets, the total wire length of a circuit or distributions of net lengths. These techniques do not output any individual

estimated net lengths. Examples of *global* net length estimation techniques are proposed in [6, 7, 12, 13, 15, 16, 20].

Semi-individual techniques are the ones that obtain data about groups of nets in a circuit. These data are mostly statistical data such as the average or variance of a group of nets with similar properties. In [17, 25] instances of *semi-individual* estimation techniques are presented.

Individual net length estimation techniques try to estimate the length for each individual net in a circuit. Since reporting lengths for each net provides much more detailed information compared to *global* and *semi-individual* techniques, *individual* net length estimators are more complicated and need much more processing time and memory space. For example, in [1, 2, 8, 24], *individual* net length estimation techniques are developed.

Considering that the focus of the current research is *a-priori individual* net length estimation techniques, the existing *a-priori individual* and *semi-individual* techniques are further described in the following.

2.1.3 Statistical Net Length Estimation Technique

In [17], one of the *a-priori semi-individual* net length estimation techniques is proposed. The input to this technique is a netlist of a standard cell circuit design and the outputs are some statistical data about the individual net lengths. These statistical data include the mean and variance of the net lengths.

In [17], a weight function is defined for each edge and is used to make the net length estimation model. This weight is calculated as:

$$wg_h(e_i) = dg(net_i) + attint_{nc}(e_i),$$

where $wg_h(e_i)$ is the weight of edge e_i and $dg(net_i)$ is the degree of net net_i from which e_i is taken. Degree of a net is the number of cells connected to that net. Variable

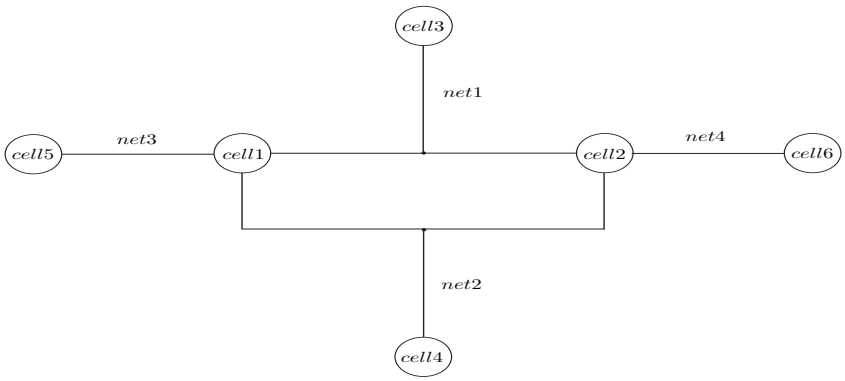


Figure 2.1: An example for calculation of wg_h for an edge

$attint_{nc}(e_i)$ is defined for edge e_i and is equal to the number of nets connected to the cells of e_i, excluding the number of common nets between those cells.

In Figure 2.1, a small circuit is presented. To calculate the weight for edge e_1 that connects $cell_1$ and $cell_2$ and belongs to net net_1, first, the degree of net_1 and $attint_{nc}(e_1)$ are calculated as 3 and 2, respectively. Then, the weight for edge e_1 is calculated as $wg_h(e_1) = 3 + 2 = 5$.

Once the weight for each net is calculated, nets with similar weights, are grouped together, and the average and variance of the lengths of the nets in each group are predicted. Several circuits are used to make this model. These circuits only contain standard cells, i.e. cells at logic design level that have same heights. It is shown that the mean net length and the variance of net lengths can be modeled as linear functions of the weights of the nets. The average and variance for each group of nets with weight

wg_h is estimated as:

$$estAveL_h(wg_h) = a_A + b_A \times Pr_1 \times wg_h$$

$$estVarL_h(wg_h) = a_V + b_V \times Pr_2 \times wg_h,$$

where $estAveL_h(wg_h)$ and $estVarL_h(wg_h)$ are the estimated average and variance of the lengths of a group of nets with weight wg_h, respectively. a_A, b_A, a_V and b_V are the model parameters and their values are reported in [17]. Furthermore, Pr_1 and Pr_2 are the measures of the overall properties of a circuit related to the net density or congestion.

The shortcomings of this technique are:

- It is designed for standard cell circuits and is not suitable for today's mixed-size circuits.
- This technique outputs the mean and variance of groups of nets and does not give any information about individual net lengths.

2.1.4 Polynomial Estimation Model For Standard Cell Circuit Designs

One of the most detailed techniques for a-priori individual net length estimation for standard cell designs is proposed in [1]. In this technique, several variables are proposed and used for net length estimation. All of these variables can be calculated before the cells of a circuit are placed. Each variable represents a certain property of the net, its connected cells or of the circuit as a whole. Then, these variables are used to make a polynomial model that estimates the length of each net.

The definitions and roles of these variables in the model are shown in Table 2.1 and further described in the following:

base length: Base length is a measure of net length that considers the effects of the sizes of the cells, i.e. their heights and widths (the dimensions of cells when they are seen

Table 2.1: The variables used in the model in [1] and the covered effect

Variable Name	Modeling Role
base length	effect of cell sizes
N_{2oth}	effect of degree-two nets of the circuit
degree-two congestion metric	effects of degree-two nets in the neighborhood of a net
degree-three congestion metric	effects of degree-three nets in the neighborhood of a net
degree-four congestion metric	effects of degree-four nets in the neighborhood of a net
degree-five congestion metric	effects of degree-five nets in the neighborhood of a net
degree-six congestion metric	effects of degree-six nets in the neighborhood of a net

from the top view), connected to a net for a typical standard cell circuit. The degree of a net, which is the number of cells connected to that net, is used to find the base length since it is shown that the nets with higher degrees are usually longer than those with lower degrees [1, 2, 8]. The base length, $base_L$, of the net net_j is calculated as:

$$base_L(net_j) = \frac{1}{2} dg(net_j) \left(h_{std} + \frac{w_{ave}}{UF} \right), \tag{2.1}$$

where h_{std} is the height of a standard cell, w_{ave} is the average width of all the cells connected to net_j and UF is the utilization factor. The utilization factor is a user-specified factor that determines what portion of a row width is to be used for cell placement.

According to (2.1), base length is equal to the adjusted average of the height needed for placing all the cells connected to the net vertically on the top of each other and row width required for placing them horizontally beside each other. Considering that in a standard cell design all cells have the same height, the height of a standard cell is used in base length calculations. In addition, in [1], all cells are considered to have the same width. This assumption results in the same average width, w_{ave}, for all of the nets. Therefore, the base length value is equal to the degree of the net multiplied by a constant value. This results in the same value of the base length variable for all nets with the

same degree.

N_{2oth}: Degree-two nets constitute the largest percentage of nets in a design, about 60% in a typical integrated circuit [26]. Furthermore, most placement algorithms are designed to put the cells connected by degree-two nets, as close together as possible. Therefore, a variable is defined to consider the effects of the presence of these nets on the length of any individual net. This variable, that is denoted as N_{2oth}, gives an estimate of how the degree-two nets cause congestion in the path used to route other nets.

N_{2oth} is then defined as:

$$N_{2oth}(net_j) = (num_{d2tot} - num_{d2nbr}(net_j)) \times \frac{num_{totnbr}(net_j)}{num_{totcell}},$$

where $num_{d2nbr}(net_j)$ and $num_{totnbr}(net_j)$ are the number of degree-two nets and the number of all the nets in the neighborhood of net net_j, respectively. Also, the total number of degree-two nets in the circuit is denoted by num_{d2tot} while $num_{totcell}$ represents the total number of cells in the circuit. In this equation, the number of degree-two nets in the neighborhood of the net is subtracted from the total number of degree-two nets in the circuit. This means that the effects of the presence of these nets on the net lengths are ignored. The reason is that these effects are already considered in the degree-two congestion metric which is discussed in the following paragraph.

degree-two to degree-six congestion metrics: Degree-two to degree-six congestion metrics, $2PinCong$ to $6PinCong$, are proposed to estimate how the length of a net is affected by the presence of nets with degree two to six, respectively, in the neighborhood of that net. For instance, the degree-two congestion metric of any net is defined to consider the effects of all degree-two nets that exist in the neighborhood of that net.

To calculate these metrics, the number of nets of a certain degree in the neighborhood of a net and the common layouts that these nets can take are found. The common layouts that a net can take are the possible paths through which a net can be routed. The number

of these possible paths is calculated by investigating the possible layout configurations, which the cells connected to that net can take.

For example, the equation for degree-two congestion metric, $2PinCong$, of net_j can be derived as:

$$2PinCong(net_j) = num_{d2nbr}(net_j) \times num_{d2PL},$$

where $num_{d2nbr}(net_j)$ is the number of degree-two nets in the neighborhood of net net_j and num_{d2PL} is the number of possible layout configurations for a typical degree-two net. It should be mentioned that although the placement results are not available when estimation is performed, the possible layouts of nets can be found by investigating common placement algorithms and so are known before the placement stage. The equations for degree-three to degree-six congestion metrics are derived using similar methods.

model development: The a-priori net length estimation model proposed in [1] consists of the proposed variables: base length, N_{2oth} and degree-two to degree-six congestion metrics. Therefore, the estimated net length using this model, ℓ_{est_B}, for net net_j is:

$$\ell_{est_B}(net_j) = f_B(base_L, N_{2oth}, 2PinCong, 3PinCong, 4PinCong, 5PinCong, 6PinCong).$$

In this equation, f_B is a specific third-order polynomial with 19 terms. These terms are the product of from one to three variables.

The coefficients of this polynomial are calculated using Ordinary Least Square Fitting (OLSF). OLSF is a method which minimizes the residual squared error to fit the best model [27]. The actual after placement net lengths of a limited number of nets from a few benchmark circuits are used as training data in finding the polynomial coefficients. These coefficients are then used to estimate the lengths of a different set of nets. The output estimated lengths are relatively correlated to the actual after placement lengths.

2.1.5 Mutual Contraction Technique

The concept of Mutual Contraction (MC) was first proposed in [18]. This metric that evaluates the proximity of the connected cells in a circuit is used in [18, 19, 25] as a technique for estimating the wire lengths. These techniques provide *a-priori individual* estimates which are applied in pre-placement clustering.

Mutual contraction can be calculated between two cells before the cells are placed. To calculate this metric, first, the weight, wg_e, of each edge, e_i, connecting two cells should be calculated as:

$$wg_e(e_i) = \frac{2}{dg(net_i)(dg(net_i) - 1)},$$

where $dg(net_i)$ represents the degree of the net net_i which includes the edge e_i. The weight of all connections, wg_t, between cell, $cell_i$ and all the cells in its neighborhood is defined as follows:

$$wg_t(cell_i) = \sum_{cell_j \in set_{nbr}} wg_c(cell_i, cell_j),$$

where $wg_c(cell_i, cell_j)$ is the weight of the edge connecting the cells $cell_i$ and $cell_j$. In fact, $wg_c(cell_i, cell_j) = wg_e(e_k)$, if e_k is the edge connecting $cell_i$ and $cell_j$. The set_{nbr} represents the set of cells adjacent to cell $cell_i$. Then, the relative weight of a connection, wg_{rel} between cells $cell_i$ and $cell_k$ can be calculated as follows:

$$wg_{rel}(cell_i, cell_k) = \frac{wg_c(cell_i, cell_k)}{wg_t(cell_i)}.$$

In [18], the mutual contraction between cells $cell_i$ and $cell_k$ is defined as the tuple $(wg_{rel}(cell_i, cell_k), wg_{rel}(cell_k, cell_i))$. To simplify the comparison between connections, the scalar metric for mutual contraction, $mc(cell_i, cell_k)$, is calculated as:

$$mc(cell_i, cell_k) = wg_{rel}(cell_i, cell_k) \times wg_{rel}(cell_k, cell_i).$$

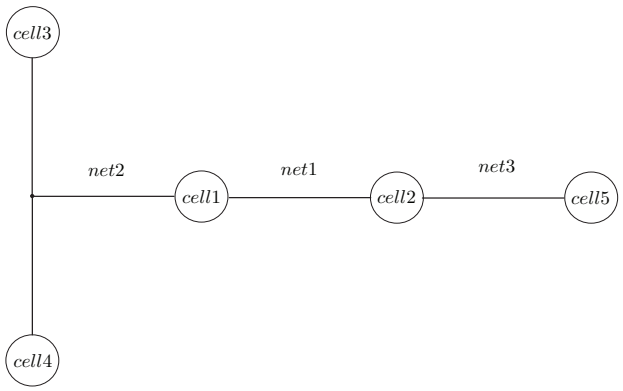


Figure 2.2: An example for mutual contraction calculation for an edge

As an example, a small circuit is given in Figure 2.2 where it is desired to calculate the mutual contraction of the connection between two cells $cell_1$ and $cell_2$. The weight of edge, e_1, connecting $cell_1$ and $cell_2$ is calculated as $wg_c(cell_1, cell_2) = wg_e(e_1) = \frac{2}{dg(net_1)(dg(net_1)-1)} = 1$. The weight of all connections between $cell_1$ and all of its neighbors is $wg_t(cell_1) = \frac{2}{2(2-1)} + \frac{2}{3(3-1)} + \frac{2}{3(3-1)} = \frac{5}{3}$. Then, the relative weight of $cell_1$ and $cell_2$ is $wg_{rel}(cell_1, cell_2) = \frac{1}{\frac{5}{3}} = \frac{3}{5}$. Similarly, $wg_{rel}(cell_2, cell_1) = \frac{1}{2}$. So, $mc(cell_1, cell_2) = \frac{3}{5} \times \frac{1}{2} = \frac{3}{10}$.

This definition of mutual contraction is only suitable for nets that only connect two cells, degree-two nets. Therefore, in [18], mutual contraction is expanded to be applicable for nets with degree higher than two. First, the relative weight of the connections of a cell, $cell_i$, to an arbitrary set of its neighbor cells, $set_{cellNbr}$, is denoted as $wg_{rel_g}(cell_i, set_{cellNbr})$ and calculated as follows:

$$wg_{rel_g}(cell_i, set_{cellNbr}) = \frac{\sum_{cell_j \in set_{cellNbr}} wg_c(cell_i, cell_j)}{wg_t(cell_i)}.$$

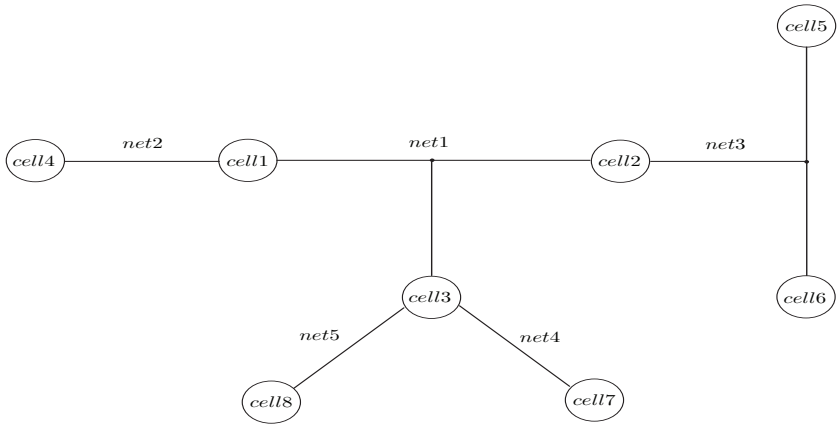


Figure 2.3: An example for mutual contraction calculation for a net

Then, the mutual contraction of a set of connected cells is defined as the multiplication of all the relative weights:

$$mc_g(set_{cellNbr}) = \Pi_{\ cell_k \in set_{cellNbr}} wg_{rel_g}(cell_k, set_{cellNbr} - \{cell_k\}). \tag{2.2}$$

To calculate the mutual contraction for net net_1 connecting three cells $cell_1$,$cell_2$ and $cell_3$ in Figure 2.3, (2.2) is used. If $set_{cellNbr} = \{cell_1, cell_2, cell_3\}$ then $wg_{rel_g}(cell_1, set_{cellNbr} - \{cell_1\}) = \frac{wg_c(cell_1,cell_2)+wg_c(cell_1,cell_3)}{wg_t(cell_1)} = \frac{\frac{1}{3}+\frac{1}{3}}{\frac{1}{3}+\frac{1}{3}+1} = \frac{2}{5}$. In a similar way, $wg_{rel_g}(cell_2, set_{cellNbr} - \{cell_2\}) = \frac{2}{3}$ and $wg_{rel_g}(cell_3, set_{nbr} - \{cell_3\}) = \frac{1}{4}$. Therefore, the mutual contraction of net_1 is calculated as $mc_g(net_1) = mc_g(\{cell_1, cell_2, cell_3\}) = \frac{2}{5} \times \frac{2}{3} \times \frac{1}{4} = \frac{1}{15}$.

Several applications of mutual contraction are presented in [2, 18, 19, 25]. In [18], an application of mutual contraction is shown where it is used in an estimation-based clustering algorithm. Even though it is shown that MC is correlated to the net lengths, it is not used to estimate the net lengths in [18]. However, it is used as a metric for scoring and comparing potential clusters.

Net length estimation is another application in which mutual contraction is frequently used. In [25], this metric is used to estimate the lengths of nets with degree two, and it is shown that mutual contraction is inversely proportional to the length of the net. The higher the mutual contraction, the lower the net length is. However, the correlation between actual after placement net lengths and mutual contraction is relatively low for nets with higher degree, so this metric is not adequate for estimating the lengths of these nets. This shortcoming is considered in [2] and is discussed in Section 2.1.7.

In [19], another application of mutual contraction in a placement algorithm is presented. In this work, constraints such as upper bounds on net lengths, are applied on the lengths of nets during the placement stage. The upper bounds are calculated using mutual contraction metric. Therefore, any increase in an individual net length during the placement stage is constrained to this boundary.

The disadvantages of a net length estimation technique which is only based on mutual contraction, such as the one proposed in [25], are:

- Length of a net is not just affected by the mutual contraction of its connected cells.
- Mutual contraction is not highly correlated to the actual after placement lengths of nets with degree more than two.

In fact, the length of a net is affected by several circuit characteristics, such as the presence of other nets in the circuit and the sizes of the cells connected by that net. These characteristics, that are not considered in a net length estimate only based on mutual contraction, are more likely to be seen in mixed-size circuits. Furthermore, mixed-size circuits contain nets with very high degrees. Therefore, a technique such as the one in [25], is not well suited to today's mixed-size circuits.

2.1.6 Intrinsic Shortest Path Length Technique

The Intrinsic Shortest Path Length (ISPL) is proposed in [8] and used for estimating net lengths before the placement stage. Therefore, ISPL is considered as an *a-priori individual* net length estimation technique. The advantage of this technique is that it deals with nets with different degrees and it is not required to convert nets with higher degrees into edges.

To simplify the definition of the intrinsic shortest path length, in [8], in the first step, the Restricted Shortest Path Length (RSPL), $rspl$, between two cells $cell_1$ and $cell_2$ of a circuit is defined as the minimum-weight set of nets that connect $cell_1$ and $cell_2$. In the other words, supposing a hypergraph $G_h = (V, E)$ where V is the set of vertices or cells and E is the set of all nets, and supposing a subset of nets $R \subset E$, the restricted shortest path length between two cells $cell_1$ and $cell_2$ when R is restricted, is defined as the sum of the weights of all nets in the minimum-weight set of nets connecting $cell_1$ and $cell_2$ in the hypergraph $G'_h = (V, E - R)$. For a typical set of cells, $S_C \subset V$ and a set of nets R, this definition is expanded as:

$$rspl_s(S_C|R) = max_{cell_1, cell_2 \in S_C} rspl(cell_1 \rightarrow cell_2|R), \tag{2.3}$$

where $rspl_s$ is the restricted shortest path length of a set of cells and $rspl(cell_1 \rightarrow cell_2|R)$ is the RSPL between $cell_1$ and $cell_2$when R is restricted. In the other words, RSPL of a set of cells is equal to the maximum RSPL of all the possible pairs of cells.

In [8], the intrinsic shortest path length is then defined as the smallest weighted path through the connectivity graph between pairs of cells connected by a net supposing that net is removed. It means that for a net, net_j, the ISPL is maximum of the restricted shortest path length of its connected cells when the net itself is restricted. So:

$$ispl(net_j) = rspl_s(S_{C_j}|net_j), \tag{2.4}$$

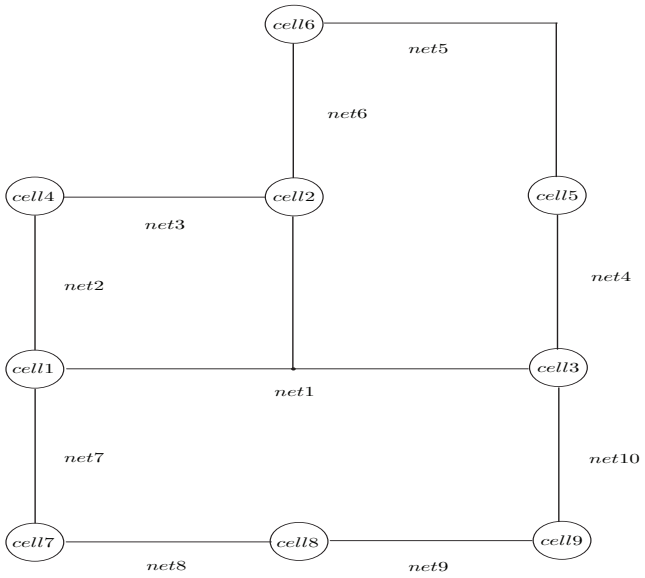


Figure 2.4: An example for ISPL calculation

where *ispl* is the intrinsic shortest path length of a net and S_{C_j} is the set of cells connected by net_j.

In Figure 2.4, a simple circuit including 9 cells and 10 nets, is presented as an example. It is desired to calculate the ISPL for net net_1. According to (2.4), $ispl(net_1) = rspl_s(cell_1, cell_2, cell_3|net_1)$. Then, from (2.3), $ispl(net_1) = max(rspl(cell_1 \rightarrow cell_2|net_1), rspl(cell_1 \rightarrow cell_3|net_1), rspl(cell_2 \rightarrow cell_3|net_1))$. Since $rspl(cell_1 \rightarrow cell_2|net_1) = 2$, $rspl(cell_1 \rightarrow cell_3|net_1) = 4$ and $rspl(cell_2 \rightarrow cell_3|net_1) = 3$, then $ispl(net_1) = 4$.

The intrinsic shortest path length is then used to develop an individual a-priori net length estimation technique. After considering the distribution of net lengths and ISPL for several circuits, an exponential relation between the estimated length and ISPL of net_j is introduced as:

$$\ell_{est_{ISPL}}(net_j) = a_1 e^{a_2 ispl(net_j)},$$

where $\ell_{est_{ISPL}}(net_j)$ is the length of net_j estimated by ISPL technique and a_1 and a_2 are

the coefficients. These coefficients are calculated by performing curve fitting using a set of after placement net lengths as training data.

This estimation technique produces results in high correlation between the estimated length, net degree and the actual wire length of degree-two nets when using a modified version of ICCAD04 benchmark circuits [28] where all cells have unit area. However, in today's mixed-sized circuits, there are nets with degrees much higher than two as well as cells of different sizes. Therefore, considering all cells to have unit size and only degree-two nets can result in large errors in the predicted lengths of a real circuit.

2.1.7 Polynomial Estimation Model For Mixed-Size Circuits

The a-priori individual net length estimation technique, proposed in [2], is the basis of the model proposed in this thesis. The estimation technique proposed in [2] can be performed before or after the clustering stage. In addition, this technique is designed to deal with mixed-size circuits by proposing several variables that consider the effects of the presence of cells with different sizes on individual net lengths.

This estimation model is a second-order polynomial that is expressed as:

$$\ell_{est_F}(net_j) = \sum_{i=1}^{n_{var}} \sum_{k=i}^{n_{var}} a_{ik} x_i(net_j) x_k(net_j) + \sum_{i=1}^{n_{var}} b_i x_i(net_j) + c. \tag{2.5}$$

In this equation, $\ell_{est_F}(net_j)$ represents the length of net net_j estimated by the model proposed in [2]. The variables $x_i(net_j)$ are the values of the variables of the model for net net_j, and n_{var} is the total number of variables that are utilized in the model. These variables are defined to provide coverage of various parameters that can affect the length of a net in a typical integrated circuit. The values of all of these variables can be calculated before the placement stage. Since nine variables are used in this technique, n_{var} is set to nine. In (2.5), coefficients a_{ik}, b_i and c are the coefficients of the polynomial that need to be calculated using estimation methods. The methods that are used to

obtain the values of these coefficients are further explained in the following. Then, the definitions of the model variables are presented.

To find the coefficients of the polynomial model of (2.5), curve fitting is performed using after placement lengths of a limited number of nets as a training data set. Ordinary Least Square Fitting (OLSF), which fits a model based on minimizing the residual squared error, is used to find the optimal polynomial coefficients. However, OLSF has the drawback that the estimates can have very large variances resulting in lowering the accuracy of the estimation [29]. In [29], Least Absolute Shrinkage and Selection Operator (LASSO) technique is introduced that addresses the mentioned problem by setting a selected number of coefficients that are not deemed to be effective in the prediction, *ineffective terms*, to 0. LASSO minimizes the sum of squared errors subject to the absolute values of the coefficients being less than a constant, the LASSO constant. This technique increases the accuracy of the estimates since it only considers the polynomial terms which have the most effect on the model.

In [2], the Algorithm Poly shown in Figure 2.5 is used to calculate the optimal values of polynomial coefficients of (2.5). This algorithm uses a combination of OLSF and LASSO to improve the quality of the polynomial estimation model. The inputs to the algorithm are a training data set of actual after placement net lengths (usually 30% of the total number of nets), values of model variables which are calculated before placement, and a set of LASSO constants. The outputs of Algorithm Poly are the optimal polynomial coefficients, a_{ik}, b_i and c, which are used for estimating the individual net lengths.

In Step 1, a LASSO constant is chosen and taken out of the given set of LASSO constants. Then, in Step 2, LASSO is used to discard ineffective terms of the polynomial. LASSO removes ineffective terms by setting them to zero. In Step 3, OLSF is used to find the coefficients only for the effective terms of the polynomial. Then, in Step 4, the estimated length of each net in the training data set is calculated using the calculated

polynomial coefficients. In Step 5, the correlation coefficients of these estimated lengths to the training data set of the actual lengths are calculated as a metric to compare different iteration results. The algorithm returns to Step 1 if any LASSO constant remains in the set of LASSO constants, in Step 6. Finally, in Step 7, the correlation coefficients from all iterations are compared. The optimal polynomial coefficients are the coefficients which yield the best correlation. These polynomial terms are used in the final estimation model.

The correlation coefficient is calculated using the following equation [30]:

$$CorrCoeff(\ell_{\mathbf{est_F}}, \ell_{\mathbf{act}}) = \frac{\mathbf{COV}(\ell_{\mathbf{est_F}}, \ell_{\mathbf{act}})}{\sigma_{\ell_{\mathbf{est_F}}} \sigma_{\ell_{\mathbf{act}}}},$$

where $\ell_{\mathbf{est_F}}$ and $\ell_{\mathbf{act}}$ represent the vectors of the estimated lengths and training data set of the actual lengths, respectively. $COV(\ell_{\mathbf{est_F}}, \ell_{\mathbf{act}})$ is the covariance of the estimated and actual lengths and $\sigma_{\ell_{\mathbf{est_F}}}$ and $\sigma_{\ell_{\mathbf{act}}}$ are the standard deviations of the estimated and actual lengths, respectively.

Inputs: Training data, Values of model variables, LASSO constant set
Output: a_{ik}, b_i and c
1. Choose a LASSO constant
2. Use LASSO to discard ineffective terms of (2.5)
3. Use OLSF to find polynomial coefficients only for the effective terms
4. Estimate length of each net using calculated polynomial coefficients
5. Calculate correlation coefficient of estimated lengths and training data
6. If LASSO constant set is not $\emptyset$, go to Step 1
7. Set a_{ik}, b_i and c to the polynomial coefficients yielding the best correlation coefficient

Figure 2.5: Algorithm Poly: High level algorithm for polynomial coefficients calculation

The model proposed in (2.5) includes global and local net and cell parameters that can be calculated before placement and are relatively independent. These variables have been used or introduced by different researchers, and are gathered as the most relevant variables in [2]. In addition, in [2], three new variables are introduced. After a variable

analysis procedure, in [2], these variables are carefully chosen among all the existing variables proposed for net length estimation. In Table 2.2, each variable and its role is briefly described. Further description of the variables and their origins are given in the

Table 2.2: The variables used in the model in [2] and the covered effect. Variables shown with bold letters are proposed in [2]

Variable	Name	Modeling Role
x_1	net degree	number of cells of a net
x_2	**macro base length**	minimum half-perimeter net length
x_3	**second level effect**	sizes of 2^{nd} level neighbors
x_4	N_{2oth}	effect of degree-two nets
x_5	inv. mutual contraction	connectivity between cells of a net
x_6	$\boldsymbol{Nettint_{nc}}$	common and uncommon nets between cells of a net
x_7	degree two to four	effects of other nets in the
x_8 , x_9	congestion metrics	neighborhood of a net

following:

net degree (x_1): Net degree is the number of cells connected to a net. Many researchers have used this variable as an indicator for estimating the length of nets, e.g. [1, 2, 8]. It is shown that a higher net degree indicates a higher length. This is a correct assumption when considering nets of greatly different degrees. However, for nets with the same or close degrees other characteristics should be considered and net degree is not adequate for these cases.

macro base length (x_2): Macro base length is a measure of net length calculated using the actual widths and heights of cells connected to a net. This variable is developed in [2] to improve the suitability of the base length metric for mixed-size circuits. The base length variable, which is initially proposed in [1], is described in detail in Section 2.1.4 and is calculated using (2.1). The base length is defined to cover the effects of the sizes of cells that are connected to a net, on the length of that net, but only for standard cell circuit designs. In calculations for variable, all the cells of the circuit are considered

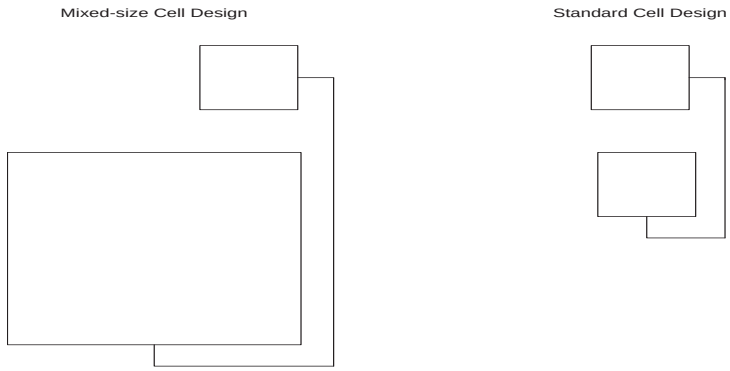


Figure 2.6: Examples of degree-two nets from standard and mixed-size cell designs

to have equal heights and widths. This results in the same base length value for nets with equal degrees.

In mixed-size circuits, cells with large differences in their dimensions can exist as a mixed-size design includes both standard cells and macro cells. In Figure 2.6, small examples of mixed-size and standard cell designs are presented. It is shown that even though both nets are degree-two nets, the base lengths of the nets are very different.

To cover the effects of the presence of mixed-size cells on the individual net lengths, the macro base length is proposed. In [2], instead of the standard cell height used in the base length equation, to calculate the macro base length the average value of actual cell heights is used. The macro base length is equal to the adjusted average of the height required for placing all the cells of the net vertically and the row width required for placing them horizontally right next to each other. The averages are multiplied by a utility factor to allow sufficient space allocation for wiring space. Therefore, the macro base length, $macroBase_L$, of a net net_j is:

$$macroBase_L(net_j) = \frac{1}{2}\left(\frac{\sum_{cell_i \in set_c(net_j)} h(cell_i)}{UF_v} + \frac{\sum_{cell_i \in set_c(net_j)} w(cell_i)}{UF_h}\right),$$

where $set_c(net_j)$ is the set of cells connected to net_j and $h(cell_i)$ and $w(cell_i)$ are the height and width of the cell $cell_i$ connected to net net_j, respectively. UF_h and UF_v

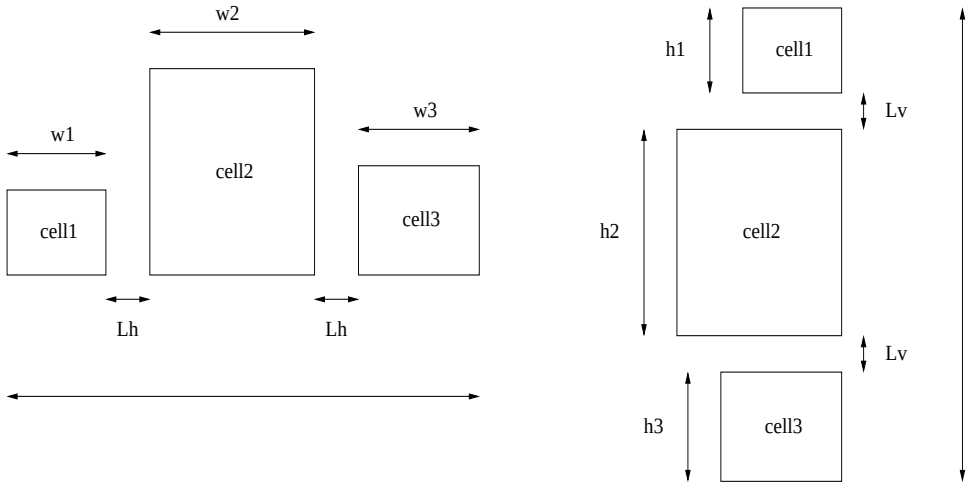


Figure 2.7: An example for calculation of macro base length

represent the utility factors for horizontal and vertical cell placements, respectively.

In Figure 2.7, an example is presented to clarify the calculation of the macro base length. L_h and L_v are the minimum lengths needed for wiring in the horizontal and vertical directions, respectively. These lengths are determined by dividing the half perimeters of the cells by the relevant utility factors. Then, the macro base length for net net_1 connecting cells $cell_1$, $cell_2$ and $cell_3$ is calculated as $macroBase_L(net_1) = \frac{1}{2}(\frac{h_1+h_2+h_3}{UF_v} + \frac{w_1+w_2+w_3}{UF_h})$.

The development of the macro base length results in the ability of the model to distinguish between nets with the same degree. Therefore, the value of macro base length is not just based on net degree but depends on the actual sizes of cells connected to a net. For example, the length of a degree-two net which is connected to a macro cell is usually higher than the length of a degree-two net that is only connected to standard cells. This concept is covered using the macro base length in net length estimation.

second level effect (x_3): The sizes of cells in the second level neighborhood of a net affect the net length, particularly when macro cells are in that second level neighborhood. Therefore, a variable for considering the second level effect is introduced in [2] to cover

the effect of the sizes of the second level neighbor cells of a net, which are cells that do not belong to the net, but are directly connected to cells of that net. This variable is calculated as the sum of the half perimeters of the cells in the second level neighborhood:

$$2ndlvl(net_j) = \sum_{cell_i \in set_{2c}(net_j)} \left(h(cell_i) + w(cell_i)\right),$$

where $set_{2c}(net_j)$ and $2ndlvl(net_j)$ are the set of cells in the second level neighborhood and second level effect variable of net net_j, respectively. Also, $h(cell_i)$ and $w(cell_i)$ are the height and width of a cell $cell_i$ in $set_{2c}(net_j)$, respectively.

In [2], this variable is used in the proposed a-priori net length estimation model. The variable analysis results show that the second level effect of a net is significantly correlated to the actual after placement net lengths. Using the second level effect and macro base length variables makes the net length estimation model suitable for mixed-size circuit designs.

N_{2oth}, (x_4): N_{2oth} is originally proposed in [1] and explained in Section 2.1.4.

Inverse Mutual Contraction (x_5): Mutual contraction, introduced in [18], is described in Section 2.1.5. In [2], during the variable selection analysis, mutual contraction is shown to be relatively correlated to the actual after placement lengths. Therefore, this variable is used as one of the variables in (2.5).

$Nettint_{nc}$, (x_6): $Nettint_{nc}$ is equal to the number of nets connected to the cells of a net excluding the nets that are connected to two or more cells of that net. This variable is developed based on $attint_{nc}$ that is originally introduced in [17] for degree-two nets, i.e. edges, and is explained in Section 2.1.3. In [2], the variable is expanded to include nets of all degrees as shown in the following:

$$Nettint_{nc}(net_j) = num_{N_{set_c(net_j)}} - num_{comN(set_c(net_j))},$$

where $num_{N_{set_c(net_j)}}$ is the number of all the nets that are connected to at least one of the

cells in the neighborhood of net net_j and $num_{comN(set_c(net_j))}$ is the number of common nets between the cells connected to net_j.

degree-two to degree-four congestion metrics (x_7), (x_8), (x_9): Degree-two to degree-six congestion metrics are proposed originally in [1] and explained in Section 2.1.4. During the variable selection procedure in [2], degree-five and degree-six congestion metrics are not chosen to be included in the estimation model. In fact, these variables show relatively low correlation to the actual after placement net lengths in comparison with the other variables. Therefore, degree-two to degree-four congestion metrics are included in the net length estimation model.

2.2 Radial Basis Functions Background

2.2.1 Introduction

As computers are utilized widely in all engineering areas, the efficient evaluation and implementation of mathematical functions become important. In many cases, it is not possible to implement the exact mathematical functions. For example, sometimes, the number of terms required to represent functions is not finite. In addition, it can be very time consuming to evaluate the exact representation of a function. Therefore, the needs to approximate mathematical functions grow rapidly [27].

One way to approximate a function is to model the data based on a pre-specified basis. This basis can be a general function such as a polynomial. This method is usually used when the data follow the general trend of a pre-defined function. However, approximating a function which represents a set of highly non-linear data that do not follow any well-known trend, is a daunting task.

Individual a-priori net length estimation techniques, such as [1, 2, 8], use pre-defined models such as exponential models, quadratic or third order polynomials, for net length

estimation. However, fitting the data with a pre-defined basis is not suitable for the complex and large circuits used today since the given data for estimation are highly non-linear.

In this research, Radial Basis Functions (RBFs) are used to estimate the individual lengths since the function for net length estimation is not known in practice and as the data suggest, is highly non-linear. RBFs are the means to estimate the multivariate functions. RBFs provide better ability to capture the details of a set of highly non-linear given data with manageable model complexity [9].

The rest of this section is organized as follows: In Section 2.2.2, different types of RBFs are introduced. The applications of radial basis functions are explained in 2.2.3. In Section 2.2.4, an algorithm is introduced which is used for estimation using RBFs. Finally, different methods for determining the locations of centers of RBFs are presented in Section 2.2.5

2.2.2 Radial Basis Functions Types

Several radial basis functions exist and are used for estimation. In fact, any function which takes the distance between any given point as the input and a certain point, the so called center, is a radial basis function. So, the general form of a radial basis function, ϕ_i, for a given data point, x, is:

$$\phi_i(x) = \phi(||x - c_i||^2),$$

where, $\phi(\cdot)$ represents a function and c_i is the center associated with that function.

There are a few radial basis functions which are commonly used by researchers such as Gaussian, multiquadric, inverse multiquadric and thin plate spline functions [9]. These RBFs are discussed in the following paragraphs:

Gaussian Radial Basis Functions: Gaussian RBFs are widely used in several different research areas [9]. One reason is that the normal distribution usually fits well to the

training data. The second reason is that the parameters of a Gaussian RBF, i.e. its mean and variance, can be intelligently found. In this thesis, Gaussian RBFs are used.

The general form of a Gaussian radial basis function, ϕ_g, for a center c and a given point x is as follows:

$$\phi_g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{||x-c||^2}{2\sigma^2}\right),$$

where σ^2 is the variance of the Gaussian function.

Multiquadric Radial Basis Functions: The general form of these RBFs for a center c and a given point x is shown in the following:

$$\phi_m(x) = \sqrt{a_m^2 + ||x-c||^2},$$

where ϕ_m is the multiquadric RBF and a_m is its parameter. This parameter is set by studying the given data which should be estimated using RBFs.

Inverse Multiquadric Radial Basis Functions: The general form of the inverse multiquadric RBFs, ϕ_{im}, for a center c and a given point x is:

$$\phi_{im}(x) = \frac{1}{\sqrt{a_{im}^2 + ||x-c||^2}},$$

where the function parameter a_{im} is found based on the input data.

Thin Plate Spline Radial Basis Functions: The general form for thin plate spline RBFs, ϕ_t, for a center c and a given point x is as follows:

$$\phi_t(x) = ||x-c||^2 \log\left(||x-c|| + a_t\right),$$

where a_t is the RBF parameter which is calculated by carefully considering the training data set. It should be noted that this parameter should be bigger than zero to deal with the case that the input point is exactly located on a center.

2.2.3 Radial Basis Functions Applications

Recently, radial basis functions have been used in many applications. Radial basis functions are widely used by scientists for estimation and interpolation. Some applications of RBFs are in neural networks [31, 32], non-linear control systems [33–35] and pattern recognition [36, 37]. Furthermore, RBFs are used in signal processing [38], image processing [39] and global optimization [40].

RBFs are applied in neural networks. These networks consist of several RBF neurons, i.e. RBFs each associated with one center, that are applied in parallel [9]. All these neurons are connected to the output neuron. The general form of an RBF network, $RBF_{NN}(\cdot)$ for each input point x is as follows:

$$RBF_{NN}(x) = \sum_{i=1}^{N} w_i \phi_{N_i}(||x - c_i||^2),$$

where c_i and w_i are the center and weight associated with RBF neuron ϕ_{N_i}, respectively.

In [31, 32], radial basis function neural networks are used for load forecasting in the electricity market. In these papers, the capability of RBFs in non-linear data estimation is used to predict the short-term load in power systems.

2.2.4 Estimation Using Radial Basis Functions

Radial Basis Functions (RBFs) are a class of functions which can be used to estimate the values of a function, given a set of measurements, without having to fit them into a pre-defined function, such as a polynomial [9, 27, 41]. In RBF-based estimation, a set of distribution functions centered throughout the data space are used to estimate the function value at any data point. Each distribution function captures the local properties of the data around each center. The estimated value at a certain point is the weighted sum of all the distribution functions evaluated at that point. These weights are calculated using the given set of measurements. Using RBFs allows the estimation to be performed

on a very complex set of data, i.e. data with high non-linearity, without having to smooth the non-linearity to fit a pre-specified model [27].

In contrast to RBFs, in order to increase the accuracy of a polynomial estimation model, such as the ones used for length estimation [1, 2], the degree of the polynomial model needs to be increased. However, the number of polynomial coefficients, $n_c(\cdot,\cdot)$, increases rapidly with the polynomial degree, p_d, and number of variables, n_{var}. This increase can be calculated using the following relation:

$$n_c(p_d, n_{var}) = \binom{p_d + n_{var}}{n_{var}}.$$

This relation results in a large number of coefficients for large number of variables and large polynomial degrees. The large number of coefficients that need to be calculated to form an estimation model shows that the strategy of increasing the polynomial degree to improve the estimation accuracy is not practical, especially for large numbers of variables and large sizes of data.

Radial basis functions can solve this problem by avoiding the need to use a specific form for the estimated function and instead using a combination of independent distributions. This combination is a highly non-linear function and can be used to estimate complex data while each independent distribution, RBF, is not complex.

For example, consider a quadratically-growing sinusoidal function, $qs(\cdot)$:

$$qs(x) = x^2 \sin(2.5\pi x^4).$$

$qs(\cdot)$ is a non-linear function and its values when $x \in [0, 1]$ are plotted in Figure 2.8, using the solid gray line. The function is evaluated at a set of random data points selected between 0 and 1 and used to reconstruct the least-square estimates using second and third order polynomials. These least-square estimates are shown in Figure 2.8 using dotted gray and solid black lines, respectively. It can be seen that the second and third order polynomial estimates do not follow $qs(\cdot)$ very closely. The correlation coefficients of these

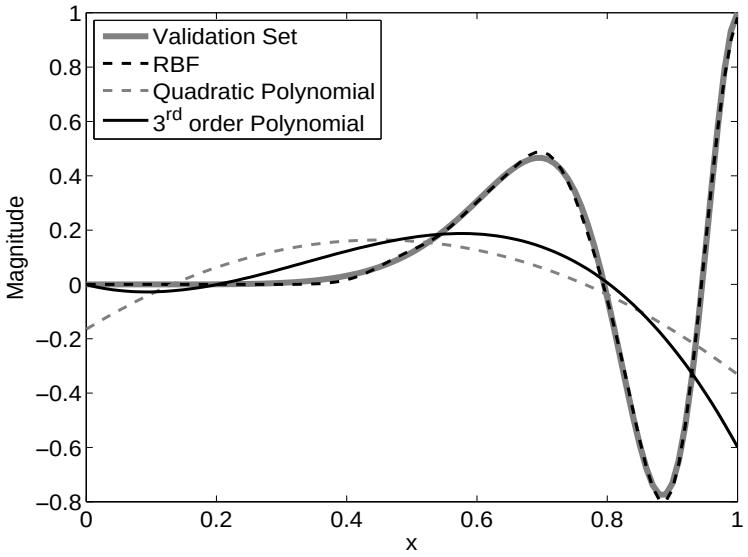


Figure 2.8: Approximation of a quadratically-growing sinusoidal function using polynomial and RBF-based models

estimates with the actual function are 14.5% and 11.7%, respectively. However, if RBFs with four centers are used for the estimation using the same data set, the correlation coefficient is increased to 99.9%, and the plot of the estimated values, as shown in Figure 2.8 using a black dotted line, follows the plot of $qs(\cdot)$ closely.

To produce function estimates using RBFs, Algorithm RBF shown in Figure 2.9 is used. The inputs to this algorithm are a set of M actual data points, their corresponding

Algorithm RBF: RBF-based estimation
Inputs: Sets of actual data, $\boldsymbol{y_{act}}$, model variables, $\boldsymbol{X}$ and center locations, $\boldsymbol{C}$
Output: Set of estimated values, $\boldsymbol{y_{est}}$
1. For each center, $\boldsymbol{c_i} \in \boldsymbol{C}$:
1.1. Evaluate RBFs using (2.6)
2. Solve (2.7) to find weights for RBFs
3. Find estimated values using (2.8)

Figure 2.9: High-level algorithm for RBF-based estimation

model variables and a set of N centers located over the data space. These centers can

be determined using different center placement methods which are discussed in Section 2.2.5. The outputs of the algorithm are the estimated values, $\boldsymbol{y_{est}}$.

Each center, $\boldsymbol{c_i} \in \boldsymbol{C}$, is associated with an RBF, $\phi_i(\cdot)$, whose inputs are the distances between any point and the center. Hence, in Step 1 of the Algorithm RBF, for a given point, $\boldsymbol{x}$, $\phi_i(\boldsymbol{x})$ is calculated as:

$$\phi_i(\boldsymbol{x}) = \phi(||\boldsymbol{x} - \boldsymbol{c_i}||^2), \tag{2.6}$$

where $\phi(\cdot)$ represents a distribution function.

In Step 2, the weights associated with RBFs, are determined by solving the following matrix equation:

$$\boldsymbol{\Phi w} = \boldsymbol{y_{act}}, \tag{2.7}$$

where $\boldsymbol{y_{act}} \in \Re^{M\times 1}$ is the vector of M actual data points, $\boldsymbol{\Phi} \in \Re^{M\times N}$ is the matrix of radial basis functions evaluated for each of N centers and M data points and $\boldsymbol{w} \in \Re^{N\times 1}$ is a vector of weights for each of the N distribution functions. Usually, $M \gg N$, so that (2.7) is an overdetermined system of equations that can be solved using OLSF, resulting in some robustness.

Finally, in Step 3, the estimated value for any given point, $y_{est}(\boldsymbol{x})$, is calculated as the weighted sum of all the RBFs:

$$y_{est}(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \phi_i(\boldsymbol{x}), \tag{2.8}$$

where w_i are the weights associated with RBFs.

2.2.5 Center Placement

Center selection is a crucial part of RBF-based estimation. Centers should be selected to cover the whole range of the data. Sometimes the input data points are scattered over a wide range of values. Most previous RBF applications use a uniform center placement

method [9] which places centers in a set of uniformly distributed grid points over all data space dimensions. Each dimension is associated with one variable in the estimation model. In Figure 2.10, an example with two variables, that are normalized to be between 0 and 1, is shown to illustrate the concept of center placement. In this figure, the solid

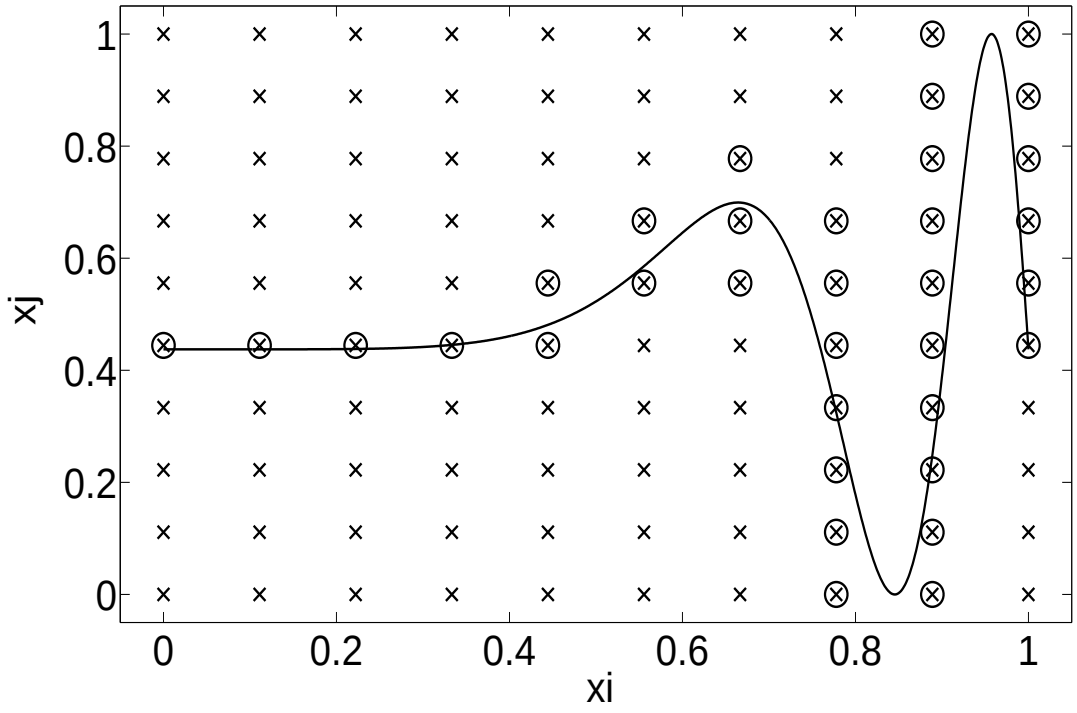


Figure 2.10: Centers generated using traditional uniform and the proposed constructive selective center placement methods for two-dimensional training data

black line is the training data set. The intersections of the center grid lines, shown with $\times$ in Figure 2.10, are the positions of the centers. In general, having normalized all the model variables between 0 and 1, the grid distance, d_c, between two adjacent centers on each dimension, when using the uniform center placement method, is

$$d_c = \frac{1}{n_{grid} - 1},$$

where n_{grid} is the number of center grids for each variable, 10 in Figure 2.10. The relation between n_{grid} and the total number of centers, n_{center}, placed by uniform center placement method, is as follows:

$$n_{center} = {n_{grid}}^{n_{var}}. \tag{2.9}$$

According to (2.9), if the estimation model includes a large number of model variables, the total number of centers created by uniform center placement method becomes extremely large. Hence, an increase in the number of center grids increases the number of centers exponentially. It should be mentioned that estimation with a large number of centers requires a huge memory space and a large amount of computing time.

In addition, there exist centers which do not have any data points in their neighborhood. These centers do not contribute to the estimation and hence, can be removed. The lower the number of centers, the lower the required processing time and memory space are. However, because of the large number of centers, center removal can also be very time consuming.

A new center placement method, called the *constructive selective uniform (CSU) center placement method* , is proposed in the literature which constructs an essential set of centers for estimation using the given data. The advantage of the constructive selective uniform center placement method over a traditional uniform center placement method is that it creates fewer centers without losing the accuracy of the estimation results. This leads to less required memory space and shorter processing time. Furthermore, using a constructive center placement method instead of a center removal technique results in further savings in computational time and memory usage.

The CSU center placement method locates the centers based on the distribution of the data. Initially, a space with n_{var} dimensions and n_{grid} center grids on each dimension is considered. The intersections of center grid lines are the locations of potential centers.

For each given data point, only the centers which are within a certain radius, r, are used in the estimation model. The algorithm for this new center placement method, Algorithm Center, is shown in Figure 2.11. Each step of the algorithm is explained in what follows.

Algorithm Center: Center placement

Inputs: Model variables ($\boldsymbol{X}$), n_{grid}
Output: Set of center locations, $\boldsymbol{C}$

1. Normalize all model variables to be between 0 and 1
2. Calculate grid distance, $d_c = \frac{1}{n_{grid}-1}$
3. Calculate the radius r using d_c
4. For each data point in model variables, $\boldsymbol{x_i} \in \boldsymbol{X}$:
   - 4.a. Find all the neighboring potential centers, $\boldsymbol{C_{p_i}}$
   - 4.b. For each $\boldsymbol{c_{p_{ij}}} \in \boldsymbol{C_{p_i}}$:
     - 4.b.i. Calculate distance of data point to potential center, $d(\boldsymbol{c_{p_{ij}}}) = ||\boldsymbol{c_{p_{ij}}} - \boldsymbol{x_i}||$
     - 4.b.ii. If the distance, $d(\boldsymbol{c_{p_{ij}}}) \leq r$, add $\boldsymbol{c_{p_{ij}}}$ to $\boldsymbol{C}$

Figure 2.11: High-level algorithm for CSU center placement method

Model variables ($\boldsymbol{X}$) and number of center grids on each dimension, n_{grid}, are the inputs of this algorithm. The set of center locations, $\boldsymbol{C}$, is the output. In Step 1, all the model variables are normalized between 0 and 1. Then, in Step 2, the grid distance is calculated. In Step 3, the radius r is calculated using the value of the grid distance. The proposed calculation of r is given in the following paragraph. Finally in Step 4, for each data point, the neighboring potential centers are found and the potential centers whose distances to the data point are less than r, are added to the center locations, $\boldsymbol{C}$.

To calculate the value of r, two extreme situations should be considered. One situation is when a datum is located on a potential center location. In this case, since one center is sufficient to cover that datum in the estimation model, $r = 0$ can be used. This is the minimum value that can be chosen for r. Another situation is when a datum is located on a spot where the distances to all the neighboring potential center locations are the same. For example, if only two variables are used, i.e. two-dimensional space,

this extreme situation occurs when the data point is located in the middle of a square with the closest potential centers on its corners. Each center is in a distance of $\frac{\sqrt{2}}{2}$ in this situation. Considering that one center is sought, in n_{var} dimensions, r should be $\frac{\sqrt{n_{var}}}{2}d_c$ which is the maximum value for the radius r. Therefore, a value within the range of $0 \leq r \leq \frac{\sqrt{n_{var}}}{2}d_c$ should be chosen.

If the maximum value of r is chosen in the constructive selective uniform center placement method, the number of centers becomes too large for estimation and results in over-fitting. On the other hand, if the minimum value of r is set, the CSU center placement method places a low number of centers and leads to less accurate estimation results. Therefore, r is chosen to be the average of the extreme values and set to be $\frac{\sqrt{n_{var}}}{4}d_c$.

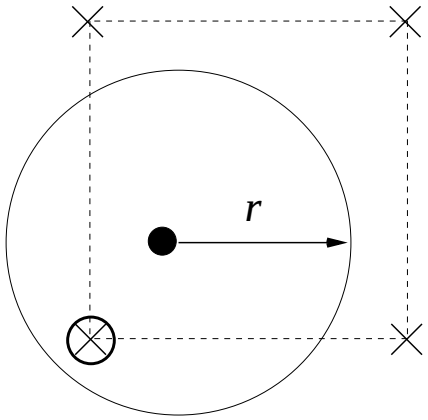


Figure 2.12: Two-dimensional example of the constructive selective uniform center placement method with one data point using radius r

In Figure 2.12, an example of CSU center placement method for estimation with two variables, two dimensions, is shown for a data point . The data point is shown by ● and the potential center locations are shown by ×. In addition, the selected center is represented by ○. It can be seen that any center which is within a distance of r from the data point is selected and used in the estimation model.

To better illustrate the effectiveness of the new center placement method, in Figure 2.10, both the traditional uniform and CSU center placement methods are used to generate centers for the data. The centers generated by the uniform technique are shown as $\times$ and the centers generated by the proposed center placement method are shown by $\circ$. For the data that are easy to estimate, the data in the left half of the figure, less centers are placed by CSU center placement method while when the data trend changes, the data in the right half of the figure, more centers are placed to provide the ability of capturing the trends.

2.3 Summary

In this chapter a background on estimation is presented. Major a-priori individual net length estimation techniques are explained in detail. In addition, performances of several existing techniques are studied and compared.

Furthermore, a background on Radial Basis Functions (RBFs) is presented in this chapter. RBFs are used for estimation and interpolation. The types and characteristics of RBFs along with an algorithm for estimation using RBFs are introduced. Finally, different methods for determining the locations of the centers of RBFs are discussed.

Chapter 3

Background: Clustering Algorithms

3.1 Introduction

As transistor sizes decrease, the power consumption and delay caused by circuit interconnects have become a larger part of the total circuit delay and power consumption. Furthermore, the number of cells and interconnects in ICs is significantly growing [42]. To handle the increasing sizes of circuits, clustering algorithms are used to minimize the amount of wires required for routing interconnects. Clustering algorithms are applied in different stages of physical design such as partitioning and placement.

Clustering has been used in physical design for more than a decade. The leading partitioning algorithms such as hMetis [43] have used clustering to improve results since 1997. In the ISPD 2005 [10] and ISPD 2006 [44] placement contests, clustering techniques were used in almost all of the placement algorithms. The clustering algorithms that are applied in the placement stage are the focus of this research. These algorithms reduce the sizes of the circuits in order to improve the placement results. This results in empowering the placers to cope with the exponentially increasing sizes of circuits. Examples of these clustering algorithms are presented in [11, 43, 45–49].

The rest of this chapter is organized as follows: In Section 3.2, the clustering problem is defined. In Section 3.3, different types of existing clustering algorithms are identified. Then, the scoreless and score-based clustering algorithms available in the literature are presented in Sections 3.4 and 3.5. In Section 3.6, the idea of a feedback framework for correcting the negative effects of clustering algorithms is described. Finally, a summary is presented in Section 3.7.

3.2 Clustering Problem Definition

The clustering problem in physical deign is defined as grouping highly connected cells of a circuit together, such that certain requirements are satisfied. Examples of these requirements can be lower and upper bounds on the sizes of clusters. Suppose that $G_h = (V, E)$ is a hypergraph where V is the set of cells or vertices and E is the set of nets. A set of clusters Clu_1, Clu_2, ..., Clu_k is found such that the following requirements are met:

$$
\begin{aligned}
k &\leq num_{totcell}, \\
Clu_i &\neq \{\}, \quad \forall i = 1, ..., k \\
Clu_i &\subset V, \quad \forall i = 1, ..., k \\
\cup_{i=1}^{k} Clu_i &= V, \\
Clu_i \cap Clu_j &= \{\}, \quad \forall i = 1, ..., k, \quad i \neq j \\
size(Clu_i) &\leq UB_{Clu}, \quad \forall i = 1, ..., k \\
size(Clu_i) &\geq LB_{Clu}, \quad \forall i = 1, ..., k
\end{aligned}
$$

where k is the number of clusters. The number of all the cells in the circuit is represented by $num_{totcell}$ and UB_{Clu} and LB_{Clu} are the upper and lower bounds on the sizes of clusters, respectively. Also, $size(Clu_i)$ represents the size of cluster Clu_i which can be either the number of cells in the cluster or the sum of the areas of all cells in the cluster.

In a typical clustering algorithm, clusters with higher quality, i.e. clusters that result in a placement solution with less total wire length, are sought. The number of nets that are cut by all of the clusters can be used as a measure of the quality of a set of clusters. A net is cut by a cluster if that net is connected to at least one cell in the cluster while at least one of its connected cells does not belong to that cluster. Considering that the aim of clustering is to group highly connected nets together, a set of clusters with a minimum

number of cut nets is desired. Therefore, in addition to the requirements mentioned above, one of the main goals of clustering is defined as the minimization of the number of nets that are cut by the set of clusters:

$$min(\sum_{i=1}^{k} num_{cut}(Clu_i)),$$

where $num_{cut}(Clu_i)$ is the number of nets cut by cluster Clu_i.

Once the clusters have been made, the nets that only connect cells from one cluster are absorbed into that cluster, i.e. eliminated. However, nets that connect at least one cell which does not belong to the cluster are kept, and counted as cut nets. Then, each group of clustered cells is replaced by a bigger cell which represents the cluster. This procedure may be repeated for a circuit to reduce its size to a desired size. Such a clustering algorithm is called multilevel clustering.

To control the reduction of circuit size in each level of a multilevel clustering, the Cell Cluster Ratio (CCR) is defined and calculated as:

$$CellClusterRatio(CCR) = \frac{number\ of\ cells\ in\ clustered\ circuit}{number\ of\ cells\ in\ original\ circuit}.$$

This metric is mostly expressed as a percentage and used to stop clustering at each level when the circuit size is reduced to the desired size. Multilevel clustering is used to perform multilevel placement or partitioning. In multilevel placement, first, all levels of clustering are performed. Then, the first level of placement is performed by placing the cells of the highest level of the clustered circuit. In the next level of the placement stage, the circuit is unclustered for one level and the exact locations of the unclustered cells are determined. This procedure is repeated for all clustering levels until all the cells have been placed. It has been shown that using clustering for circuit placement results in decreased runtime and increases the quality of the placement results as compared to the placement results without using clustering [50, 51].

3.3 Types of Clustering Algorithms

There exist several classifications of clustering algorithms. One possible classification of clustering algorithms is based on whether or not they perform comparison of the potential clusters, i.e. score-based versus scoreless. Examples of score-based algorithms are best-choice clustering [47], Net Cluster [11], edge-separability [48], and fine granularity clustering [49]. Although score-based algorithms are more desirable in terms of the quality of the results, they are usually more time consuming than the scoreless approaches. Edge coarsening [43], FirstChoice [52], PinEC [45], hyperedge coarsening [53] and heavy-edge matching [46] are examples of scoreless clustering where clusters are made without comparison to each other. These algorithms can have low runtimes and reduce the circuit sizes effectively, but the quality of clusters might not be very high, because high quality potential clusters can be disregarded since their cells are already assigned to other clusters.

Another classification of clustering algorithms can be made based on if clusters are made by focusing on grouping cells together, e.g. FirstChoice, PinEC, best-choice, or by collapsing nets, e.g. hyperedge coarsening and Net Cluster.

3.4 Scoreless Clustering Algorithms

Scoreless clustering is an approach in which potential clusters are formed by considering the local neighborhood of cells. In this approach, these potential clusters are finalized and are not compared with any other potential clusters. These clusters are locally the best possible clusters while they may not be the best overall clusters. The lack of comparison of finalized clusters with other potential clusters results in lowering the quality of clustering results. However, scoreless clustering approaches are relatively fast which makes them suitable for specific circumstances. The existing scoreless clustering approaches that are

most commonly used in physical design are presented in the following subsections.

3.4.1 Edge Coarsening

One of the basic clustering algorithms is called Edge Coarsening (EC) and presented in [43, 52]. This algorithm is based on randomly considering cells in a circuit. In this algorithm, first, a randomly selected cell is considered. If this cell is already clustered, it will be skipped and another cell is selected randomly. If the selected cell is not clustered, it is chosen as the seed cell. The seed cell is a cell that is considered as a base for formation of a cluster or potential cluster. Then, the weights of connections of this seed cell to its unclustered adjacent cells are calculated. The cell with the highest connectivity to the seed cell is grouped with the seed cell in a cluster.

Edge coarsening forms clusters of two cells in each iteration. This formed cluster is locked and does not change during the following iterations. To find the weight of connection between two cells, first, the weight, wg_n, of a typical net net_j is defined:

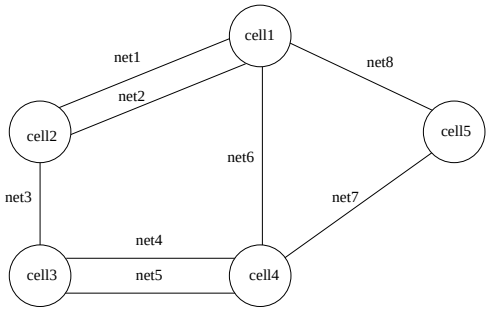
$$wg_n(net_j) = \frac{1}{dg(net_j) - 1},$$

where $dg(net_j)$ is the degree of net_j, i.e. number of cells that are connected to the net. Then, the weight of connection between two cells $cell_i$ and $cell_j$, $wg_{con}(cell_i, cell_j)$, is calculated as:

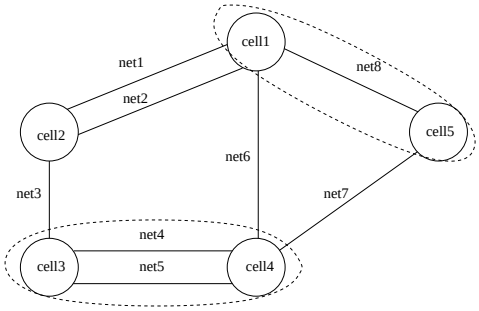
$$wg_{con}(cell_i, cell_j) = \sum_{\{net_k | net_k \rightarrow \{cell_i, cell_j\}\}} wg_n(net_k) = \sum_{\{net_k | net_k \rightarrow \{cell_i, cell_j\}\}} \frac{1}{dg(net_k) - 1}, \quad (3.1)$$

where $net_k \rightarrow \{cell_i, cell_j\}$ means that net_k is connected to the cells $cell_i$ and $cell_j$.

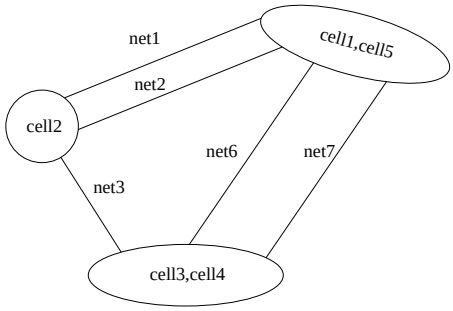
In Figure 3.1(a), a small circuit is presented. This circuit contains 5 cells that are connected by 8 nets. When EC is used for clustering, a seed cell is selected randomly. In this example, the first seed cell is randomly chosen to be $cell_5$ which is an unclustered cell. Therefore, the weight of connections between this seed cell and its unclustered



(a) Original Circuit



(b) Formed Clusters (shown with dashed lines)



(c) Final Clustered Circuit

Figure 3.1: Example for clustering using EC

adjacent cells, $cell_1$ and $cell_4$, are calculated as $wg_{con}(cell_5, cell_1) = \frac{1}{dg(net_8)-1} = \frac{1}{2-1} = 1$ and $wg_{con}(cell_5, cell_4) = \frac{1}{dg(net_7)-1} = \frac{1}{2-1} = 1$. Since these weights are the same, $cell_1$ can be randomly chosen to form a cluster with $cell_5$. In the next iteration, another random seed cell is selected which can be, for example, $cell_3$. This seed cell does not belong to any clusters and is connected to $cell_2$ and $cell_4$ that do not belong to any cluster. In a similar way, the connectivity weights are calculated as $wg_{con}(cell_3, cell_2) = \frac{1}{2-1} = 1$ and $wg_{con}(cell_3, cell_4) = \frac{1}{2-1} + \frac{1}{2-1} = 2$. As the connection weight between $cell_3$ and $cell_4$ is higher, they are put in a new cluster. The next random seed cell is, for example, $cell_1$ which is already clustered. This seed cell is skipped and $cell_2$ is randomly chosen as the next seed cell. Even though this cell has not already been clustered, all of its adjacent cells are part of other clusters and cannot form a new cluster with $cell_2$. This cell is skipped too and $cell_4$, which is the only unvisited cell remaining, is selected as the seed cell. However, this cell is already clustered and is skipped by EC algorithm. In Figure 3.1(b), the formed clusters are shown using dashed lines and in Figure 3.1(c), the clustered circuit is shown that contains 3 cells and 5 nets. It should be mentioned that nets net_4, net_5 and net_8 are absorbed into the clusters.

3.4.2 Hyperedge Coarsening

In [52,53], Hyperedge Coarsening (HC) is proposed. In this clustering algorithm, instead of considering cells in a random manner, nets or hyperedges are visited. In the first step, the nets of a circuit are ranked by their degrees in ascending order. To deal with nets with the same degree, the sum of the areas of all cells connected to each net is used to rank the nets in increasing order.

In the next step, the first ranked net, i.e. net with the lowest degree, is selected as a seed net. The seed net is a net that is considered as a base for the formation of a cluster or potential cluster. Similar to EC clustering, if the seed net is already clustered, it will

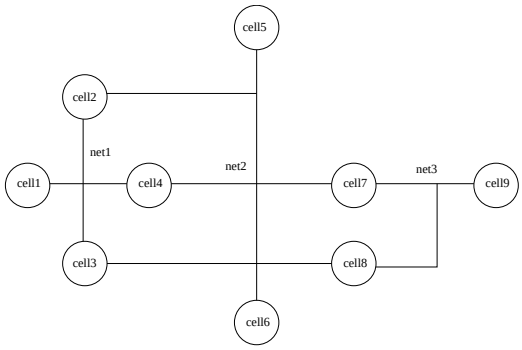
be skipped and the next ranked net will be visited. A net is considered as a clustered net if at least one of its connected cells belongs to a formed cluster. If the seed net is not clustered, all of its connected cells are grouped as a new cluster. This procedure is repeated for all the nets in the circuit. The procedure stops when the desired cell cluster ratio is obtained or all the nets of the circuit have been visited.

This algorithm can be repeated for the clustered circuit by considering the formed clusters as the cells of a new circuit. To prevent the production of unbalanced clusters, an upper bound on the sizes of clusters is set. It should be noted that size of a cluster is the sum of areas of cells in that cluster.

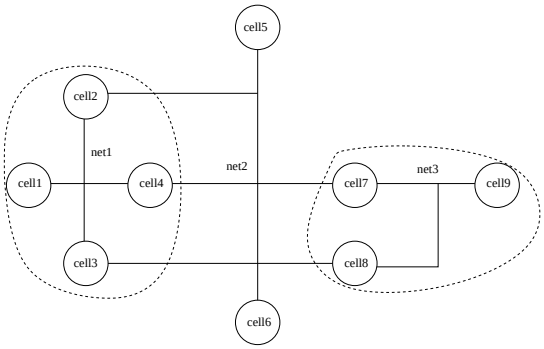
In the HC clustering algorithm, in contrast to EC clustering that makes clusters with only two cells, the clusters may contain different numbers of cells, depending on the degree of the net used to form the cluster.

In Figure 3.2(a), an example circuit with 9 unit-size cells and 3 nets is presented. The hyperedge coarsening algorithm is used to form clusters with an upper bound of 5 units on the cluster size. In the first step, the nets are sorted based on their net degrees in increasing order, as follows: net_3, net_1 and net_2. The first ranked net which is net_3 with degree 3, is visited as the first seed net. This net can be clustered since none of its cells belongs to any cluster. All of the cells connected to this net are grouped in a new cluster that absorbs net_3. The size of this cluster is 3 unit-size cells which does not violate the constraint on the cluster size. The next seed net is net_1 which is connected to 4 cells: $cell_1$, $cell_2$, $cell_3$ and $cell_4$. These cells are not part of any cluster and so they form a new cluster with a size of 4 units which is less than the upper bound on the cluster size. The last seed net is net_2 which is a net with degree 7. Some of the cells connected to this net are already clustered. Therefore, net_2 is skipped and the clustering is concluded since all of the nets have been visited.

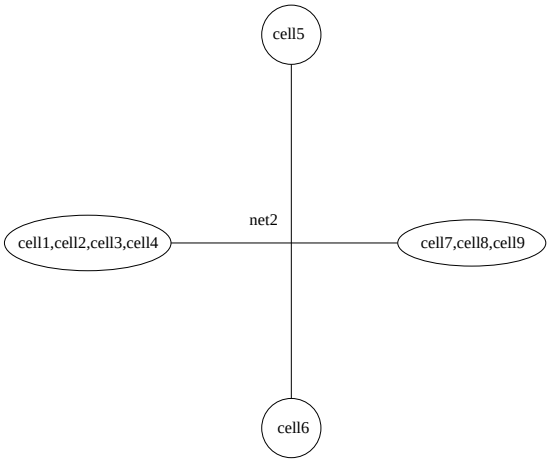
In Figure 3.2(b), the formed clusters are shown using dashed lines. Finally, in Figure



(a) Original Circuit



(b) Formed Clusters (shown with dashed lines)



(c) Final Clustered Circuit

Figure 3.2: Example for clustering using HC

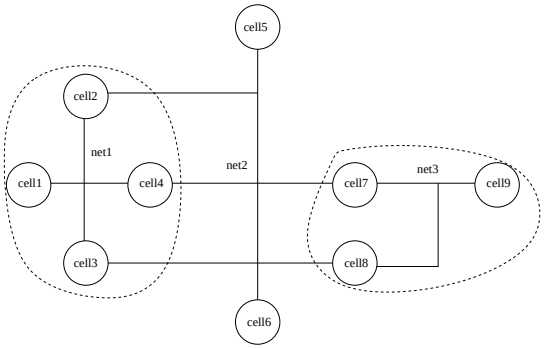
3.2(c), the circuit is presented after it has been clustered using the HC clustering algorithm. It can be seen that the circuit has been reduced to a circuit with only 4 cells and 1 net.

3.4.3 Modified Hyperedge Coarsening

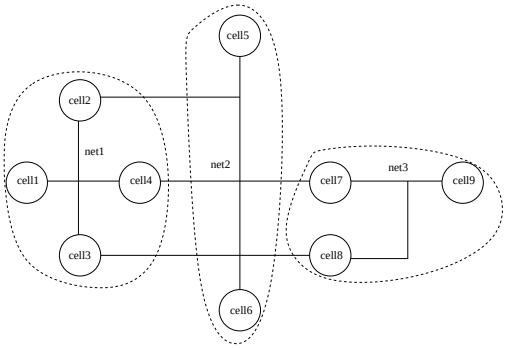
Modified Hyperedge Coarsening (MHC) was proposed in [52, 53]. This algorithm is a modified version of hyperedge coarsening (HC). As discussed in Section 3.4.2, the HC algorithm skips any seed net that is connected to at least one clustered cell. This results in skipping a large number of nets during clustering, which may decrease the efficiency of the clustering algorithm. The MHC algorithm was developed to cluster nets even when they are connected to clustered cells.

Modified hyperedge coarsening first uses the HC algorithm to cluster a circuit. Then, nets are considered again to find those nets that are not clustered because they are connected to clustered cells. The unclustered nets form new clusters in this process. The constraint on the cluster size must still be satisfied by these new clusters. This procedure continues until all the nets have been clustered or a certain cell cluster ratio is reached.

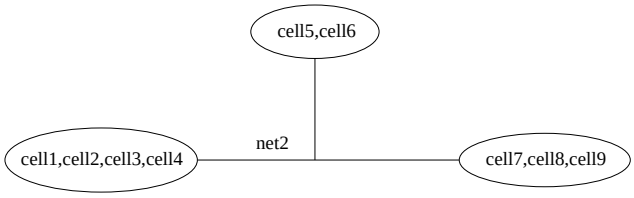
In Figure 3.3, MHC is performed on the circuit used as an example for explaining the hyperedge clustering algorithm in Section 3.4.2. As it can be seen in Figure 3.3(a), two clusters are produced in the first step of the MHC algorithm. Then, nets are visited again and net_2 is found to be the only net which is not clustered. This net along with its connected cells that have not already been clustered, i.e. $cell_5$ and $cell_6$, form a new cluster. This cluster satisfies the constraint on the cluster size since its size is 2 units which is less than the specified upper bound on cluster size, 5. The MHC algorithm stops since all the nets have been clustered. In Figure 3.3(b), the clusters formed by MHC are shown. Finally, in Figure 3.3(c), the final clustered circuit is presented. This circuit is reduced in size compared to the circuit clustered by the HC algorithm.



(a) Clusters formed in the first step of MHC (shown with dashed lines)



(b) All of the clusters formed with MHC (shown with dashed lines)



(c) Final Clustered Circuit

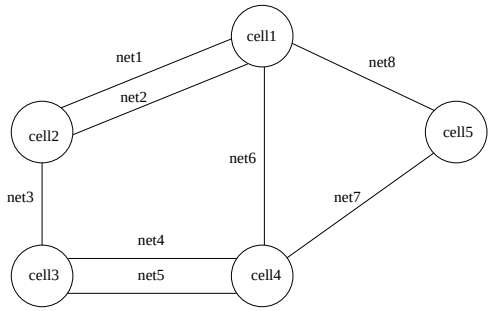
Figure 3.3: Example for clustering using MHC

3.4.4 FirstChoice Clustering

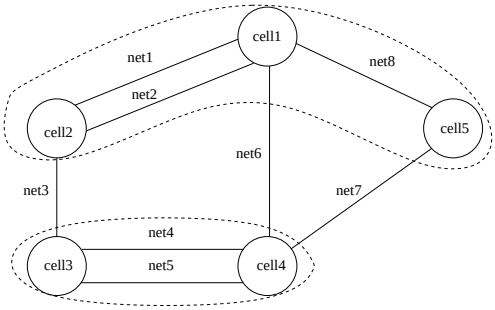
The FirstChoice Clustering (FC) algorithm is proposed in [52]. This algorithm is a modified version of the edge coarsening (EC) algorithm discussed in Section 3.4.1. Similar to EC, FirstChoice clustering is based on randomly visiting circuit's cells. If a visited cell has not yet been clustered, it is treated as a seed cell.

In EC, the weights of the connections between the seed cell and any unclustered adjacent cells are calculated and the seed cell is clustered with the neighbor cell that has the maximum weight. The calculation of connection weights is presented in (3.1). However, the only difference between EC and FC is that in the FC algorithm, the clustering status of the adjacent cells is not considered. In the other words, in FC, for each seed cell, the weights of its connections with all the adjacent cells, i.e. clustered and unclustered, are calculated. Then, the seed cell is clustered with the adjacent cell that has the maximum connection weight without considering its clustering status. Therefore, if the adjacent cell with the maximum connectivity is part of a cluster, the seed cell is added to the existing cluster, rather than forming a new cluster. Since clusters such as these may include too many cells, the clusters formed by the FC algorithm are required to satisfy a constraint on their size.

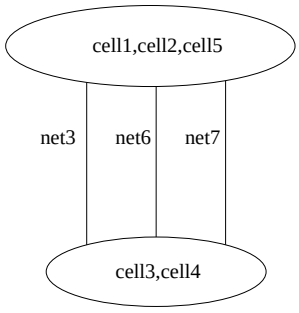
In Figure 3.4(a), the example circuit used in Section 3.4.1 to explain the EC algorithm is used to provide a fair comparison between EC and FC. The cells are randomly visited. However, this random sequence was set to be the same as that used in the example for EC, in order to have a proper comparison. Similar to the example for EC, the first visited cell is $cell_5$ which is unclustered and considered as a seed cell. The weights of its connections to cells $cell_1$ and $cell_4$ are calculated as $wg_{con}(cell_5, cell_1) = \frac{1}{dg(net_8)-1} = \frac{1}{2-1} = 1$ and $wg_{con}(cell_5, cell_4) = \frac{1}{dg(net_7)-1} = \frac{1}{2-1} = 1$. Similar to the example for EC, since the connectivities are equal, $cell_1$ is randomly chosen to be grouped in a new cluster with $cell_5$. The cluster size constraint, set to 5 in this example, is satisfied by this cluster. As



(a) Original Circuit



(b) Formed Clusters (shown with dashed lines)



(c) Final Clustered Circuit

Figure 3.4: Example for clustering using FC

in EC example, the next visited cell is $cell_3$ which can be considered as a seed cell because it is still unclustered. Calculating $wg_{con}(cell_3, cell_2) = 1$ and $wg_{con}(cell_3, cell_4) = 2$, $cell_3$ and $cell_4$ form a new cluster that meets the constraint on cluster size. The next visited cell, $cell_1$, is a clustered cell and so is skipped. Then, $cell_2$ is visited randomly and is considered as a seed cell since it is not clustered. The connectivities are calculated as $wg_{con}(cell_2, cell_1) = \frac{1}{dg(net_1)-1} + \frac{1}{dg(net_2)-1} = \frac{1}{2-1} + \frac{1}{2-1} = 2$ and $wg_{con}(cell_2, cell_3) = \frac{1}{dg(net_3)-1} = \frac{1}{2-1} = 1$. The connectivity between $cell_2$ and $cell_1$ is more than that between $cell_2$ and $cell_3$. Although $cell_1$ is already clustered and since FC is used for clustering, $cell_2$ is added to the already existing cluster that includes $cell_1$ and $cell_5$. This cluster still satisfies the constraint on the cluster size. Since the last visited cell, $cell_4$, is already clustered, it is skipped and the clustering process stops because all cells have been visited.

In Figure 3.4(b), the formed clusters are shown with dashed lines. In Figure 3.4(c), the circuit clustered by the FirstChoice clustering algorithm is shown. It can be seen from Figures 3.4 and 3.1 that when the example circuit is clustered using FC, the circuit size is reduced to 2 cells and 3 nets while the circuit clustered by EC includes 3 cells and 5 nets.

3.4.5 Heavy-Edge Matching

In [46], the Heavy-Edge Matching (HEM) clustering algorithm is proposed. This algorithm is a modified version of edge coarsening (EC) in which the priority is with forming clusters that are smaller physically. The main process of HEM clustering is similar to EC, but the definitions and calculations of connection weights are different.

A new measure for the calculation of connectivity between two cells, $cell_i$ and $cell_j$, is defined as:

$$conn_{HEM}(cell_i, cell_j) = \frac{1}{area(cell_i) + area(cell_j)} \sum_{\{net_k | net_k \rightarrow \{cell_i, cell_j\}\}} \frac{1}{dg(net_k)}, \quad (3.2)$$

where $conn_{HEM}(cell_i, cell_j)$ is the connectivity between $cell_i$ and $cell_j$ defined in HEM clustering. Also, $area(cell_i)$ and $area(cell_j)$ are the areas of $cell_i$ and $cell_j$, respectively. The net net_k is the net that is connected to both $cell_i$ and $cell_j$ and $dg(net_k)$ is the degree of net_k. It can be seen that in this connectivity measure, the effects of cell areas are considered.

Using the connectivity measure defined in (3.2), priority is given to the nets with lower degrees and the cells with smaller areas. The clustering procedure is the same as EC. The cells in the circuit are visited randomly and a cell is considered to be a seed cell if it is unclustered. The connectivities between the seed cell and any unclustered adjacent cells are calculated. The seed cell is grouped in a new cluster with the adjacent cell with maximum connectivity. This procedure is repeated for other seed cells until all the cells of the circuit have been visited or a pre-specified cell cluster ratio is reached.

3.4.6 PinEC Clustering

A variation of the heavy-edge matching (HEM) clustering algorithm is presented in [45]. In this algorithm, called PinEC clustering, a new method for connectivity calculations along with an algorithm different from HEM are used. In this algorithm, the connectivity of the nets with degree two is doubled to emphasize the priority of these nets. In addition, the clustering algorithm is modified. In this new algorithm, the cells are updated dynamically after the formation of a new cluster. Thus, once a new cluster is formed by grouping two connected cells, those two cells are removed from the set of cells in the circuit and replaced by the new cluster. In subsequent iterations, this cluster is treated as a cell with an area equal to sum of the areas of the clustered cells. Therefore, all subsequent calculations of connectivities and formation of clusters are aware of all previously formed clusters. The calculation of connectivity that includes the areas of clustered cells in order to give priority to the formation of small clusters, is defined as

follows:

$$conn_{PinEC}(cell_i, cell_j) = \frac{1}{area(cell_i) + area(cell_j)} \sum_{\{net_k | net_k \rightarrow \{cell_i, cell_j\}\}} \frac{k_{PinEC}}{dg(net_k)},$$

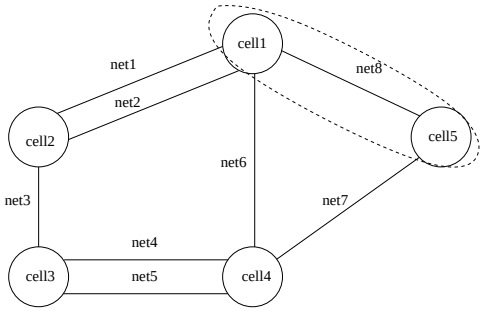
and:

$$k_{PinEC} = \begin{cases} 2, & net_k \text{ is a degree-two net} \\ 1, & \text{otherwise} \end{cases}$$

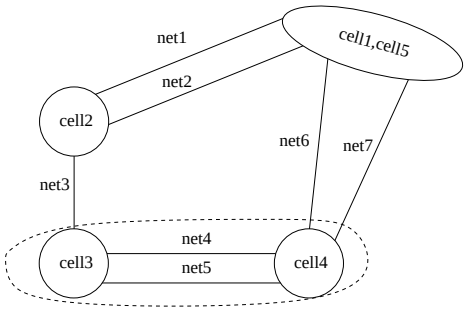
where $conn_{PinEC}(cell_i, cell_j)$ is the connectivity between cells $cell_i$ and $cell_j$ and k_{PinEC} is a constant which is equal to 2 for degree-two nets and equal to 1 for nets with other degrees.

An example of clustering using the PinEC algorithm is presented in Figure 3.5. The same circuit as the one presented in the example for EC in Section 3.4.1 is used. This circuit contains 5 unit-size cells and 8 nets and is clustered using PinEC clustering. The cells are visited randomly but in the same order as the example for EC.

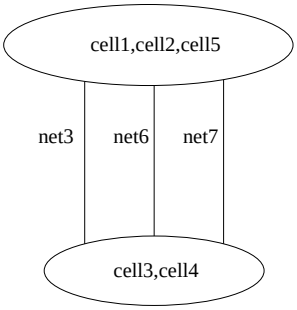
The first cell to be visited is $cell_5$ which is considered as the seed cell since it is not clustered. The adjacent cells are $cell_1$ and $cell_4$, and both are unclustered. The connectivities are calculated as $conn_{PinEC}(cell_5, cell_1) = \frac{1}{1+1}(\frac{2}{2} + \frac{2}{2}) = 1$ and $conn_{PinEC}(cell_5, cell_4) = \frac{1}{1+1}(\frac{2}{2} + \frac{2}{2}) = 1$. Since the connectivities are equal, $cell_1$ is randomly selected to form a new cluster with $cell_5$. This cluster is shown with a dashed line in Figure 3.5(a). Then, the circuit is updated by replacing $cell_1$ and $cell_5$ by a cell that includes the newly formed cluster and its area is the sum of the areas of all cells in the cluster. The updated circuit is shown in Figure 3.5(b). Then, $cell_3$ is visited and considered as the seed cell. Finding the connectivities as $conn_{PinEC}(cell_3, cell_2) = \frac{1}{1+1}(\frac{2}{2}) = \frac{1}{2}$ and $conn_{PinEC}(cell_3, cell_4) = \frac{1}{1+1}(\frac{2}{2} + \frac{2}{2}) = 1$, $cell_3$ and $cell_4$ are grouped as a new cluster. This cluster is shown with a dashed line in Figure 3.5(b). After updating the circuit, $cell_2$ is visited and considered as a seed cell. The connectivities are calculated as $conn_{PinEC}(cell_2, cell_{1,5}) = \frac{1}{1+2}(\frac{2}{2} + \frac{2}{2}) = \frac{2}{3}$ and $conn_{PinEC}(cell_2, cell_{3,4}) = \frac{1}{1+2}(\frac{2}{2}) = \frac{1}{3}$. Therefore, $cell_2$ is added to the already formed cluster that includes $cell_1$ and $cell_5$. The



(a) Original Circuit



(b) Updated Circuit



(c) Final Clustered Circuit

Figure 3.5: Example for clustering using PinEC

PinEC clustering stops since there is no unclustered cell in the circuit. The final version of the clustered circuit that includes 2 cells and 3 nets, is shown in Figure 3.5(c).

3.5 Score-Based Clustering Algorithms

In this section, the existing score-based clustering algorithms are explained. A score-based clustering algorithm is an algorithm in which potential clusters and their corresponding scores are found. These scores measure the quality of the potential clusters. The priority of potential clusters is based on their scores. It means that a potential cluster with a higher score is formed before those with lower scores. The potential clusters are ranked based on their scores and the clusters are formed in order, until all of them have been formed or a certain cell cluster ratio has been obtained.

In some score-based clustering approaches, the scores are updated dynamically. Once a cluster has been formed, it is treated as a regular cell and the clustering algorithm continues with the awareness of all previously formed clusters.

3.5.1 Edge Separability-Based Clustering

A score-based clustering algorithm called Edge Separability-Based Clustering (ESC), is proposed in [48, 54]. Edge separability is a measure for the connectivity between two connected cells in a circuit. In this measure, all of the paths between two cells are considered in the calculation of connectivity. These paths include the nets that connect the cells directly and paths that connect the cells via other cells. This point is the feature that distinguishes ESC, since before its proposal, all of the clustering algorithms used connectivity metrics that do not consider the effects of paths that connect cells via other cells.

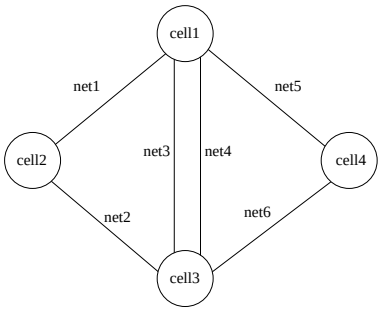
The ESC algorithm works as follows: First, the circuit is considered as the corresponding graph. This means that all the hyperedges of the circuit are converted to their

corresponding edges. In the next step, all the cells of the circuit are visited. For each cell, the edge separability with each adjacent cell is calculated and the maximum of these values is found. The cell is paired with the adjacent cell that has the maximum edge separability. Then, all of these pairs of cells are ranked in decreasing order according to the value of their edge separability. In fact, edge separability is used to score the potential clusters. Therefore, the pair of cells, i.e. potential cluster, with the maximum edge separability and first rank between all potential clusters is formed as a finalized cluster. Then, the second ranked potential cluster is formed. This procedure continues until a pre-specified cell cluster ratio is reached or all of the potential clusters are formed.

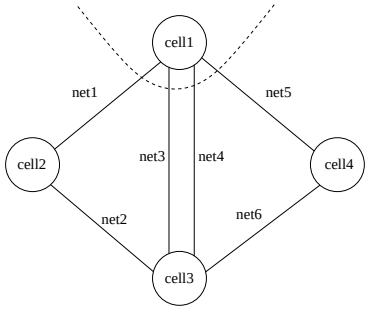
To calculate the edge separability between two cells, $cell_i$ and $cell_j$, that are connected by a net, net_k, the concept of mincut should be defined. Suppose that the circuit is divided into two partitions and that $cell_i$ and $cell_j$ are located in different partitions. These partitions are determined such that the number of nets that are cut by partitions is minimized. The mincut is the minimum number of nets that are cut by the partitions. The edge separability of net net_k is equal to the mincut of $cell_i$ and $cell_j$.

In Figure 3.6(a), a small circuit is presented. The edge separability between $cell_1$ and $cell_2$ is desired. The mincut of $cell_1$ and $cell_2$ is calculated in Figures 3.6(b) and (c). In these figures, the partitions are divided by a dashed line and the number of cut nets is calculated. It can be seen that the minimum number of cut nets is found as 2 when partitions are made as in Figure 3.6(c). Therefore, the edge separability between $cell_1$ and $cell_2$ is 2.

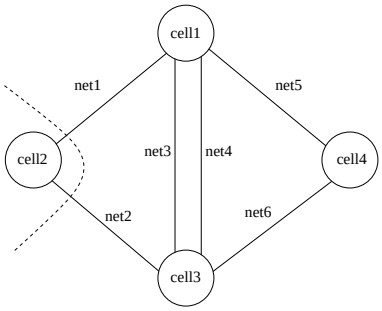
The edge separability-based clustering algorithm calculates the connectivity of all pairs of cells and compares them. This clustering algorithm compares the potential clusters globally. Therefore, this algorithm is the first global clustering algorithm which considers all of the potential clusters at one time.



(a) Example Circuit



(b) Number of cut nets = 4



(c) Number of cut nets = 2

Figure 3.6: Example for calculation of edge separability

3.5.2 Fine Granularity Clustering

Another score-based clustering algorithm is presented in [49]. Fine Granularity Clustering (FGC) can be summarized as follows: First, the circuit is converted to the corresponding graph by converting the hyperedges into edges. Then in the first phase of the algorithm, which is called the cluster formation phase, all the cells of the circuit are randomly visited. If a visited cell has not yet been clustered, it is considered as a seed cell. Each seed cell is grouped in a potential cluster along with the adjacent cell with the maximum connectivity. Then, the adjacent cell with the second maximum connectivity is attracted into the potential cluster. This procedure continues until the constraint on the maximum cluster size is reached or all the adjacent cells have been attracted to the potential cluster. The first phase of the algorithm procedure is repeated for all seed cells until all of the cells in the circuit have been visited.

In the second phase of the algorithm, the cluster refinement phase, the formed potential clusters are considered again. Cluster refinement is performed using the FM algorithm proposed in [55]. The quality of the clusters is improved by moving cells between different clusters. The constraints on the cell movements are the lower and upper bounds on the cluster sizes. In FGC, a cluster is defined as a fine cluster if it includes 2 to 6 standard cells. In the refinement phase, a clustering solution of fine potential clusters is desired that maximizes the total connectivity of the nets absorbed by the clusters.

3.5.3 Best-Choice Clustering

Best-Choice Clustering (BC) is a score-based clustering algorithm proposed in [47, 56]. Similar to FGC, best-choice clustering includes two main phases. In this clustering algorithm, the two cells with the maximum weight of connection are found and clustered.

In the first phase of the BC algorithm, the cells in the circuit are randomly visited. For each cell, the weights of connections to its adjacent cells are calculated. Then, the

visited cell is grouped with the adjacent cell with the maximum connectivity in a potential cluster. The potential clusters are ranked in a priority queue in decreasing order based on the connectivity values.

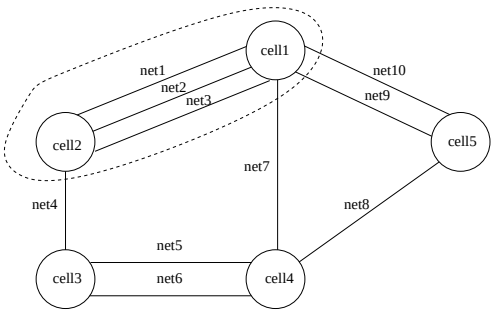
In best-choice clustering, the connectivity between cells of potential clusters is identified as the clustering score, and gives an evaluation of the quality of the potential cluster. It is calculated similar to (3.2):

$$conn_{BC}(cell_i, cell_j) = \frac{1}{area(cell_i) + area(cell_j)} \sum_{\{net_k | net_k \to \{cell_i, cell_j\}\}} \frac{1}{dg(net_k)},$$

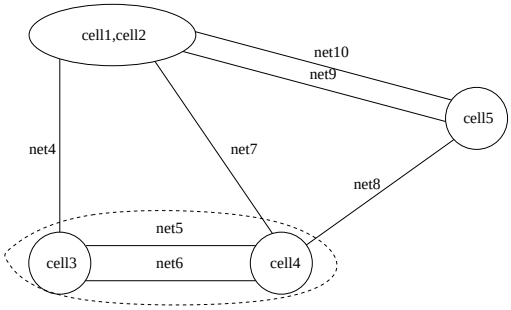
where $conn_{BC}(cell_i, cell_j)$ is the connectivity between two adjacent cells $cell_i$ and $cell_j$ and net_k is a net that is connected to both $cell_i$ and $cell_j$.

In the second phase of BC, the first ranked potential cluster in the priority queue is formed. Once the cluster has been formed, the priority queue is dynamically updated. In other words, when a potential cluster is formed, the clustered cells are replaced by a new cell with an area equal to the sum of the areas of the cells in the cluster. Then, the connectivities of this new cell with its adjacent cells are calculated and new potential clusters are identified. In the next step, the rankings of potential clusters in the priority queue are updated. The procedure of forming clusters and dynamically updating the priority queue is repeated until all of the cells of the circuit have been clustered or a certain cell cluster ratio is reached.

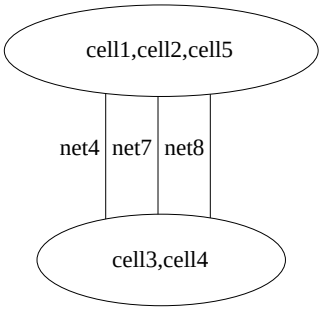
In Figure 3.7(a), a circuit is presented as an example. This circuit, that includes 5 unit-size cells and 10 nets, is desired to be clustered by the best-choice clustering algorithm. In the first phase, all of the cells are visited and the priority queue of potential clusters is formed. The first ranked potential cluster, shown with the dashed line in Figure 3.7(a), consists of $cell_1$ and $cell_2$ with the clustering score of 0.75. This potential cluster is formed and $cell_1$ and $cell_2$ are replaced by a new cell which is the newly formed cluster and the connectivities are re-calculated. In Figure 3.7(b), the updated circuit is shown.



(a) Original Circuit



(b) Updated Circuit



(c) Final Clustered Circuit

Figure 3.7: Example for clustering using BC

In the updated priority queue, the potential cluster shown with the dashed line in Figure 3.7(b), is the first ranked potential cluster with the clustering score of 0.5. This procedure continues until all of the cells of the circuit have been clustered. The example circuit clustered by the BC algorithm is presented in Figure 3.7(c), and includes 2 cells and 3 nets.

The best-choice clustering algorithm tries to cover all the advantages of the previous algorithms. The method for calculating the connectivity is taken from the heavy-edge matching algorithm, that is a scoreless clustering. Using scores for potential clusters improves the quality of clusters. This gives global information about the quality of clusters. Therefore, at each clustering iteration, the potential cluster with the globally highest quality will be formed.

3.5.4 Net Cluster Clustering

Another score-based algorithm, Net Cluster Clustering (NC), is proposed in [11, 57]. In contrast to the previous score-based clustering algorithms, that are based on clustering of cells of the circuit, the NC algorithm clusters the circuit by clustering its nets. Clustering nets means the absorption of nets into the clusters. In fact, NC is a net-based clustering algorithm in which it is desired to reduce the number of nets of the circuit.

Net cluster clustering includes three main phases. First, the potential clusters are identified and their corresponding scores are calculated. Then, the potential net-based clusters are ranked according to their scores. Finally, the potential clusters are finalized by consideration of the potential scores.

In the first phase, the potential cluster identification phase, the cells of the circuit are visited randomly. Then, the circuit is divided into two partitions. Each visited cell is considered as a seed cell and the seed cell is put in one partition called the natural partition. The other cells of the circuit are put into the other partition. The number

of nets that are cut by the partitions is used as a metric to measure the quality of the potential clusters. For each seed cell, all of the adjacent cells are examined by moving them one at a time into the natural partition. If by moving an adjacent cell into the natural partition the number of nets cut by the partitions decreases, that cell is permanently moved into the natural partition. Otherwise, the adjacent cell is kept in the other partition. This procedure is repeated for all the adjacent cells of the seed cell. Finally, this natural partition is considered as a potential cluster. However, for some seed cells there may not be any potential cluster since no reduction results from the movements of any of the adjacent cells.

The procedure of potential cluster formation is done for all the cells of the circuit and a score is assigned for each potential cluster. For each potential cluster, $pclu_i$, the score is defined as follows:

$$score_{clu}(pclu_i) = \frac{num_N(pclu_i)}{num_C(pclu_i) - 1} \times \frac{1}{\sum_{cell_j \in pclu_i} area(cell_j)},$$

where $score_{clu}(pclu_i)$, $num_N(pclu_i)$ and $num_C(pclu_i)$ are the assigned score, number of nets and number of cells of the potential cluster $pclu_i$, respectively. Also, $area(cell_j)$ is the area of $cell_j$ that is clustered in $pclu_i$.

In the second phase of NC, the nets are assigned specific scores. The scores of potential clusters are used to calculate the scores for the nets of the circuit. To calculate the score for each net, the status of that net in different potential clusters is considered. The score of each net is defined as the sum of all the scores of the potential clusters that absorb that net minus the sum of all the scores of the potential clusters that cut that net. This definition is formulated as:

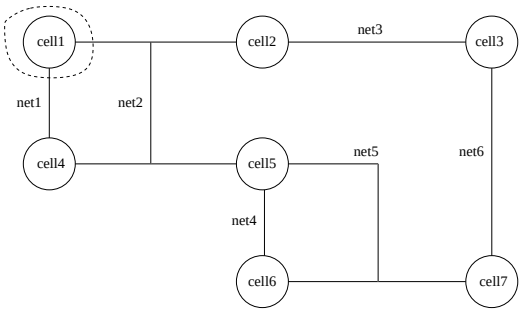
$$score_{net}(net_j) = \sum_i k_{NC}(net_j, pclu_i) \times score_{clu}(pclu_i),$$

$$k_{NC}(net_j, pclu_i) = \begin{cases} 1, & net_j \text{ is absorbed into } pclu_i \\ -1, & net_j \text{ is cut by } pclu_i \\ 0, & \text{otherwise} \end{cases}$$

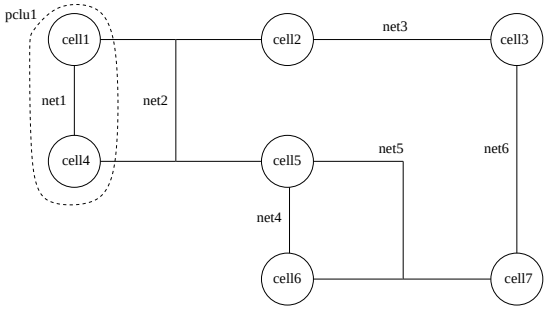
where $score_{net}(net_j)$ is the score of net net_j defined in NC. The constant $k_{NC}(net_j, pclu_i)$ is 1 if net_j is absorbed by the potential cluster $pclu_i$, -1 if net_j is cut by $pclu_i$ and 0 otherwise. In this phase of the algorithm, the nets or potential clusters are ranked in a decreasing order based on their corresponding scores and put in a priority queue. The potential cluster or net with a higher score has the priority to form a cluster.

In the third phase of NC, the potential clusters are finalized. If the priority queue is made by ranking the potential clusters, they are formed as new clusters based on their rankings. On the other hand, if nets are ranked in the priority queue, the potential cluster that includes the first ranked net is clustered first. A net is clustered by clustering its connected cells into a cluster. This procedure is repeated for all the nets until a desired cell cluster ratio has been reached.

In Figure 3.8, an example of the formation of potential clusters and the calculation of their corresponding scores is presented. The circuit, presented in Figure 3.8(a), includes 7 unit-size cells and 6 nets. For cell $cell_1$, the adjacent cells are $cell_2$, $cell_4$ and $cell_5$. In Figure 3.8(a), the initial natural partition for $cell_1$ is shown with a dashed line. The number of nets that are cut by this partition is 2 since nets net_1 and net_2 are cut. If $cell_2$ or $cell_5$ are moved to the natural partition, the number of cut nets increases. Therefore, moves such as these are not accepted by NC algorithm. However, if $cell_4$ is moved to the natural partition, net_1 is not cut anymore and the number of cut nets decreases to 1. Therefore, this move is accepted and $cell_4$ is permanently moved to the natural partition. Then, for $cell_1$, the natural partition, potential cluster $pclu_1$, is formed as shown in Figure 3.8(b) by a dashed line. The score for this natural partition is calculated



(a) Original Circuit (Original natural partition for $cell_1$ is shown with a dashed line)



(b) Potential cluster for $cell_1$ is shown with a dashed line ($score_{clu}(pclu_1) = \frac{1}{2}$)

Figure 3.8: Example for formation of potential clusters and calculation of their scores using NC

as $score_{clu}(pclu_1) = \frac{num_N(pclu_1)}{num_C(pclu_1)-1} \times \frac{1}{\sum_{cell_j \in pclu_1} area(cell_j)} = \frac{1}{2-1} \times \frac{1}{1+1} = \frac{1}{2}$.

Since the NC clustering algorithm is net-based, it is more effective in reducing the number of nets in a circuit. Therefore, the total after placement length of the circuits clustered by NC is typically lower than that of the circuits clustered by cell-based clustering algorithms.

3.5.5 SafeChoice Clustering

In [58], a recent score-based clustering algorithm called SafeChoice Clustering (SC) is proposed. SafeChoice is based on a condition referred to as safe clustering, which guarantees that forming a cluster of some objects does not result in increasing the after placement net lengths. The concept of safe clustering is defined as follows: If a set of cells can be moved to a single position such that the total wire length of the circuit is not increased, that set of cells can be formed as a safe cluster. Since the cells should be moved to one single position, the cells are assumed to have negligible areas to ignore overlaps.

In [58], after defining the concept of safe clustering, a safe condition for pair-wise clustering is developed where it is proved that if any two cells satisfy the condition of safe clustering, forming a cluster of them does not lead to the degradation of the after placement net lengths. Therefore, they can be grouped together as a pair-wise safe cluster. Then, a clustering algorithm based on a priority queue of pair-wise safe clusters is developed. This priority queue is made by ranking the identified potential safe clusters in decreasing order, based on a score, $score_{SC}$. Then, the potential clusters are finalized based on their rankings in the priority queue. In [58], a criterion for stopping the clustering is proposed in order to prevent the algorithm from forming clusters that can degrade the after placement wire lengths.

The score, $score_{SC}$, that covers both the safeness and area of potential clusters, is

defined for two cells $cell_i$ and $cell_j$ as:

$$score_{SC}(cell_i, cell_j) = SF_{ij} + \theta_{SC} \times \frac{area(cell_i) + area(cell_j)}{aveA_{stdCell}},$$

where θ_{SC} is found experimentally and set to be 4 and $aveA_{stdCell}$ is the average area of the standard cells in the circuit. In addition, SF_{ij} is a measure of safeness for a cluster including $cell_i$ and $cell_j$ and calculated based on the mode of operation of SC algorithm. This mode of operation is defined based on the stopping criterion that is used by SC algorithm. There are three modes of operation for SC: safety guarantee mode, clustering ratio mode and smart mode. In the safety guarantee mode, the SC algorithm forms only the completely safe clusters. However, under the clustering ratio mode, the SC algorithm may produce some unsafe clusters as well as the safe ones in order to reach a certain cell clustering ratio. In the smart mode, the algorithm is stopped whenever a typical placer can achieve the best placement results. For each of these modes of operations, different values of SF_{ij} are used.

3.6 Feedback Loop for Clustering Correction

In [2], the effects of different clustering algorithms on the lengths of individual nets are studied. It is shown that by clustering the circuits, the lengths of individual nets are not always decreased. In fact, a significant number of nets experience increases in their lengths.

It is proposed in [2] to apply pre-placement net length estimation techniques to predict the negative effects of clustering algorithms on the individual net lengths. Since the clustering is conducted before placement, there is no detailed information available about individual net lengths. Therefore, pre-placement net length estimation, that can provide information about the net lengths before, during and after clustering, is performed. Once the individual net lengths before and after clustering have been compared,

further clustering steps can be performed to minimize the negative effects. Although in [2], it is proposed to implement a corrective feedback loop on the clustering stage, it fails to identify and implement the structure of this feedback loop. However, in this thesis, the structure of a predictor-corrector framework for clustering is proposed and implemented. This framework is discussed in detail in Chapter 5.

3.7 Summary

In this chapter, a comprehensive background on the existing clustering algorithms is presented. Clustering is widely used before partitioning and placement stages of physical design to handle the large sizes of circuits. The clustering algorithms are divided into two groups. Scoreless clustering algorithms form clusters iteratively while they do not compare them globally. On the other hand, score-based algorithms improve the quality of clusters by comparing them and forming the clusters with the highest quality.

In this chapter several existing scoreless and score-based clustering algorithms are discussed in detail. To better explain the procedure of these algorithms, several examples are used. Finally, a recently proposed idea for a framework for correcting the negative effects of clustering is presented.

Chapter 4

The Proposed Net Length Estimation Model

4.1 Introduction

In an ideal world, a designer can predict which nets in a circuit will have excessive lengths, and hence cause the highest delay, before its placement of the components is performed. Several of the a-priori net length estimation techniques discussed in Chapter 2 try to achieve this goal [1, 2, 7, 15, 17, 18]. Some of these techniques can only predict wire length distributions or the average lengths of groups of nets [7,15,17]. In [1] and [2], two individual net length estimation techniques, which are based on a polynomial model, are given. However, they still require significant improvement to be deemed reliable.

In this research, the pre-placement net length estimation technique presented in [2] is first further studied and tested on the most recent available benchmarks: the ISPD05 placement benchmarks [10]. The estimation models are validated by comparing the estimated net lengths to the after placement net lengths. A number of improvements to the model and the variables are suggested.

Individual net length estimation techniques, such as [1, 2, 8], utilize pre-specified models such as polynomials or exponential models. However, fitting the data using a pre-specified model is not suitable for today's circuits since the input data are highly non-linear.

In this research, Radial Basis Functions (RBFs) are used to estimate the individual lengths. Using RBFs enables the model to capture the details of a set of highly non-linear input data while the complexity of the model is still manageable [9]. In RBF-based estimation techniques, the estimated length of each net is the weighted sum of a set of

Gaussian distribution functions referred to as RBFs. For each RBF, a center and an associated variance need to be determined. The centers and the variances of the RBFs should be selected to represent specific properties of the data. Once the RBFs are defined, ordinary least-square fitting (OLSF), which minimizes the residual squared error to fit the best model [27], is used to calculate the weight of each RBF used in estimating the wire lengths. Furthermore, a new technique is proposed that adjusts the variance of the RBFs used at each center.

The main contributions proposed in this chapter are as follows:

- Proposal of a new variable to consider the effects of different placers in the net length estimation technique.
- Proposal of a new variable to consider the effects of the presence of fixed cells in the net length estimation technique.
- Proposal and implementation of an RBF-based pre-placement individual net length estimation technique.
- Proposal and implementation of a new method to find the proper variance for radial basis functions.

The rest of this chapter is organized as follows: In Section 4.2, several improvements to the model variables are proposed. Then, in Section 4.3, the proposed RBF-based net length estimation technique is presented. Finally, in Section 4.4, a brief summary is presented.

4.2 Modeling Improvements

The net length estimation technique proposed in [2] is the best existing model for mixed-size circuits. However, it still needs improvements to be used as a reliable estimator.

The recent benchmark circuits, ISPD05 placement benchmarks, include some cells which are fixed in specific locations. The presence of these cells, which are usually macro cells, affect the length of nets that are connected to them.

To improve the accuracy of the model in [2], a study is performed to analyze the characteristics of different placers and the impact of fixed cells in placement solutions. The lengths of nets significantly depend on the algorithm that the placer utilizes to place the circuit. Some placers produce higher net lengths compared to others, on average. Therefore, the effects of different placement algorithms should be considered. Then, a new length variable is proposed to incorporate the effects of different placers and fixed cells into the model. In addition, it is shown that using non-quadratic models for variables that show specific characteristics can make them better correlated to the actual lengths. These improvements along with their rationales are discussed in the following Sections.

4.2.1 Placer Effects

To consider the effects of differences between placement algorithms on the net lengths, the most commonly used academic placers are studied. These placers are Capo10.5 (Capo) [59], mPL6 (mPL) [50] and FastPlace3.0 (FastPlace) [60]. Each placer uses a unique method for placing the cells. Capo is a min-cut placer that recursively bisections a net list until each partition is small enough to be placed optimally. It also allocates whitespace to ensure routability of the placement solution. This is in contrast to mPL and FastPlace, which are analytical placers that minimize a total wire length objective function (mPL uses a log-sum-exp wire length model and FastPlace uses a quadratic wire length model). Capo tends to have higher wire length compared to mPL and FastPlace because of its emphasis on routability. However, it tends to do better with respect to the congestion indicators. It is proposed to model these differences in the net length estimation.

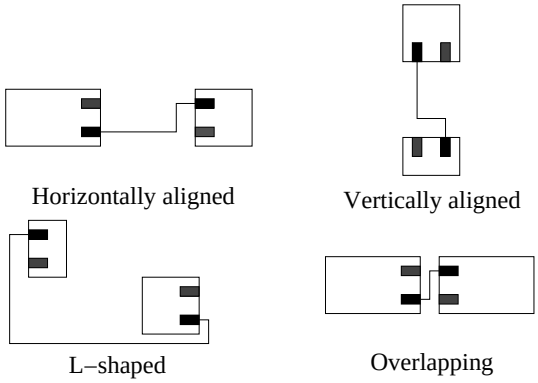


Figure 4.1: Shapes of different configurations of degree-two nets

To quantitatively differentiate between different placers, the occurrences of different configurations of degree-two nets after placement are investigated. Degree-two nets are chosen since they are the major percentage in a typical integrated circuit and are the building blocks for other nets. For example, in ICCAD04 benchmark circuits [28], about 60% of the nets are with degree two [26].

Degree-two nets are divided into 4 categories based on their after placement shape. These categories are: horizontally aligned, vertically aligned, L-shaped and overlapping pins, as shown in Figure 4.1. Considering a degree-two net with two placed pins, $p_1 : (p_{1_x}, p_{1_y})$, $p_2 : (p_{2_x}, p_{2_y})$, these shapes can be written as:

- vertically aligned (V): $p_{1_x} = p_{2_x}, p_{1_y} \neq p_{2_y}$
- horizontally aligned (H): $p_{1_x} \neq p_{2_x}, p_{1_y} = p_{2_y}$
- overlapping (O): $p_{1_x} = p_{2_x}, p_{1_y} = p_{2_y}$
- L-shaped (L): $p_{1_x} \neq p_{2_x}, p_{1_y} \neq p_{2_y}$

Based on this description, in a row-based placement the pins are required to be at the same height along the cell edges for a net to qualify as horizontally aligned even

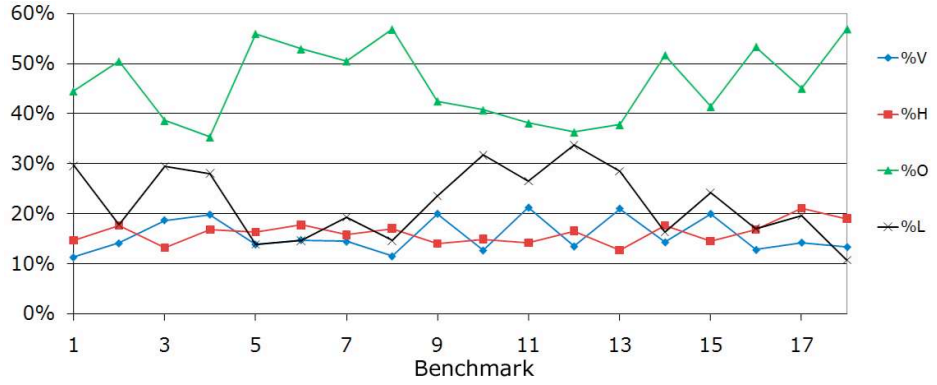
though the cells possessing its pins are in the same row. The same problem exists when considering vertically aligned or overlapping shapes. In this research, it is proposed to consider the pins of a net to be aligned if they are within one standard cell height, h_{std}, of each other. Therefore, relaxed alignment conditions can be explained as:

- vertically aligned (V): $|p_{1_x} - p_{2_x}| \leq h_{std}, |p_{1_y} - p_{2_y}| > h_{std}$
- horizontally aligned (H): $|p_{1_x} - p_{2_x}| > h_{std}, |p_{1_y} - p_{2_y}| \leq h_{std}$
- overlapping (O): $|p_{1_x} - p_{2_x}| \leq h_{std}, |p_{1_y} - p_{2_y}| \leq h_{std}$
- L-shaped (L): $|p_{1_x} - p_{2_x}| > h_{std}, |p_{1_y} - p_{2_y}| > h_{std}$

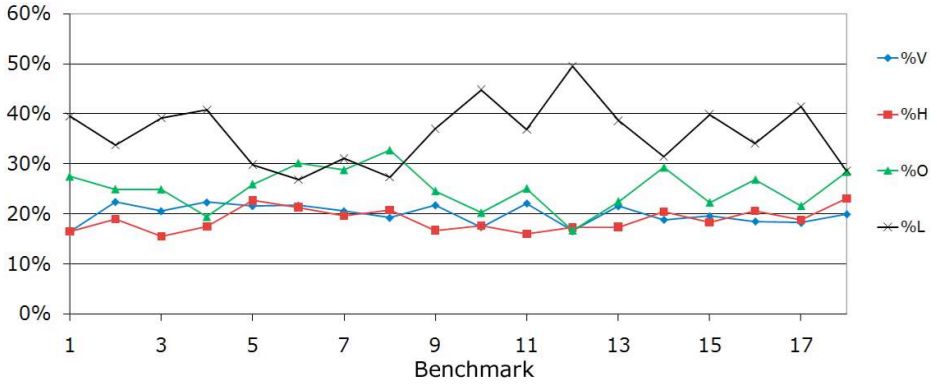
The percentages of nets with each alignment for all of the ICCAD04 benchmarks are calculated for three placers, Capo, FastPlace and mPL, and shown in Figures 4.2 (a), (b) and (c), respectively. In these figures, the abscissa represents the circuit number and the ordinate represents the percentage of nets in a specific configuration. It can be seen that Capo overlaps the largest number of nets, while FastPlace usually has more L-shaped nets. mPL aligns more of the nets horizontally than the other two placers.

Each of the configurations results in a different minimum length, i.e. the minimum length when no other cells are encountered along the path connecting the pins of the net. For example, the minimum length for a vertically aligned net is between zero and two standard height of the cells of the circuit. On the other hand, the minimum length of an L-shaped net is between zero and half of the perimeter of the cells whose pins are connected to the net.

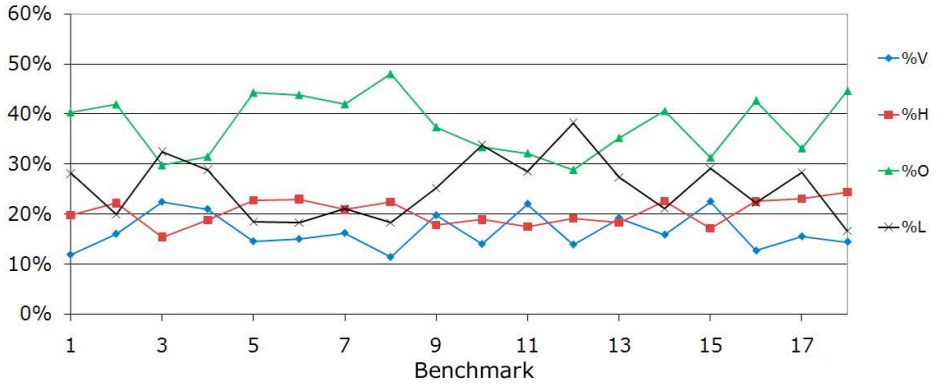
In this thesis, it is proposed to change the macro base length variable, x_2, proposed in the estimation technique of [2] discussed in Chapter 2, to reflect these configurations. In addition, the effects of the presence of macro cells should be included in this variable. Macro cells are considered since if one of the pins of a net belongs to a macro cell, its expected length increases significantly, in relation with the size of the macro cell.



(a) Capo10.5



(b) FastPlace3.0



(c) mPL6

Figure 4.2: Percentage occurrences of different net configurations placed by different placers for all circuits in the ICCAD04 benchmark suite

Table 4.1: Proposed base length values for different configurations of degree-two nets

Configuration	Net Type	
	Standard	Non-Standard
V: $base_V$	h_{std}	$\lambda * \max\{h_1, h_2\}$
H: $base_H$	$(w_1 + w_2)/2$	$\lambda * \max\{w_1, w_2\}$
O: $base_O$	h_{std}	$\lambda * h_{std}$
L: $base_L$	$(w_1 + w_2 + h_1 + h_2)/2$	$\lambda * (\max\{w_1, w_2\} + \max\{h_1, h_2\})$

The nets are then categorized in two groups: nets that connect only pins of standard cells and nets that are connected to pins of one or more non-standard cells. For each category of nets, a new base length calculation is proposed, where the calculations are based on the average expected length of a net if the pins of the net are placed in a given configuration.

The base lengths for different configurations of degree-two nets are given in Table 4.1. In this table, (w_1, h_1) and (w_2, h_2) are the width and height of cells 1 and 2, whose pins are connected to the net, respectively, and h_{std} is the height of standard cells in a circuit. For nets that only connect pins of standard cells, Column 2, the base length calculations are as follows:

If the pins are horizontally or vertically aligned, the expected base length of the net connecting them, $base_H$ or $base_V$, will be the average of their widths, $(w_1 + w_2)/2$ or heights $2(h_{std})/2 = h_{std}$, respectively.

For a net in an L-shape, the half-perimeter length is chosen as the expected base length, $base_L$. For a net with overlapping pins, the average expected base length, $base_O$, is set to be the height of one standard cell.

If a net connects pins of non-standard cells, then the width (horizontal), the height (vertical), the half-perimeter (L-shape) of the macro cell or a standard height (overlapping) times a factor, $\lambda \geq 1$, is used to calculate the base length.

To calculate λ, the ratio of the average actual length of those nets which are connected

Table 4.2: Average probabilities in percentage for each configuration using different placers

Configuration	Probability	Capo	FP	mPL
Vertical	P_V	15.6%	20.0%	16.6%
Horizontal	P_H	16.1%	18.8%	20.3%
Overlap	P_O	46.1%	25.1%	37.8%
L-shape	P_L	22.2%	36.1%	25.3%

to pins of one or more non-standard cells over the average actual length of the nets which are only connected to pins of standard cells for each circuit, is calculated. This value ranges from 5 to 20, depending on the circuit. In order to find a single λ for all the circuits, a line search optimization using different values of λ, $5 \leq \lambda \leq 20$, is performed. For each value, the estimation is performed and the correlation coefficient of estimated lengths to the actual lengths is calculated. Based on the comparison between the recorded correlation coefficients, the best overall λ is around 10. Therefore, in the following experiments, λ is set to 10.

Once the base lengths for each configuration have been calculated, for each placer, the new base length variable, x_2', is set to be equal to a weighted sum of these base lengths. The weights are the average probabilities of each configuration occurring over all the benchmarks for a specific placer. So the new base length can be calculated as follows:

$$\text{new base length} : x_2' = P_H base_H + P_V base_V + P_L base_L + P_O base_O, \qquad (4.1)$$

where, $base_H$, $base_V$, $base_L$ and $base_O$ represent the appropriate base length for each configuration as given in Table 4.1, and P_H, P_V, P_L and P_O represent the average probabilities of specific configurations occurring for the placer. These average probabilities are calculated and given in Table 4.2.

The effectiveness of x_2' is evaluated by replacing the macro base length variable, x_2, with x_2' and the model correlation coefficients are calculated for all degree-two nets. The

Table 4.3: Comparison of the model correlation coefficients using macro base length and the new base length. The actual lengths, which are used in the model, are produced by Capo10.5 (Capo), FastPlace3.0 (FP) and mPL6 (mPL) placers.

Circuit	Macro Base			New Base			Improvement		
	Capo	FP	mPL	Capo	FP	mPL	Capo	FP	mPL
IBM01	0.62	0.65	0.58	0.70	0.72	0.68	11.7%	10.0%	16.7%
IBM02	0.72	0.77	0.73	0.74	0.81	0.76	2.4%	4.4%	4.0%
IBM03	0.70	0.71	0.65	0.72	0.75	0.69	3.4%	5.5%	6.2%
IBM04	0.62	0.70	0.60	0.65	0.75	0.65	5.4%	6.1%	8.3%
IBM05	0.77	0.75	0.74	0.77	0.75	0.74	-0.1%	0.0%	-0.4%
IBM06	0.83	0.83	0.82	0.85	0.84	0.84	1.9%	0.6%	2.5%
IBM07	0.58	0.61	0.55	0.61	0.64	0.59	4.6%	5.9%	6.7%
IBM08	0.72	0.83	0.71	0.73	0.83	0.72	1.6%	0.0%	1.3%
IBM09	0.60	0.63	0.56	0.63	0.66	0.59	4.1%	4.9%	5.3%
IBM10	0.46	0.56	0.46	0.50	0.61	0.51	8.0%	8.6%	9.7%
IBM11	0.56	0.59	0.50	0.58	0.62	0.53	3.6%	5.5%	7.7%
IBM12	0.52	0.62	0.50	0.53	0.64	0.51	2.8%	2.2%	2.8%
IBM13	0.59	0.61	0.58	0.61	0.64	0.60	3.7%	5.1%	3.4%
IBM14	0.53	0.54	0.51	0.55	0.56	0.54	4.6%	4.5%	5.0%
IBM15	0.57	0.58	0.56	0.58	0.61	0.59	3.1%	5.6%	4.2%
IBM16	0.56	0.59	0.58	0.58	0.61	0.60	3.0%	3.3%	3.4%
IBM17	0.47	0.49	0.43	0.48	0.51	0.46	3.9%	4.1%	7.7%
IBM18	0.64	0.61	0.63	0.66	0.64	0.65	2.4%	4.6%	3.4%
Average	-	-	-	-	-	-	3.9%	4.5%	5.4%

results of this experiment using the ICCAD04 benchmark suite are tabulated in Table 4.3. Using the new base length improves the model correlation by 3.9% to 5.4% on average for each placer and always improves the correlation except for IBM05, which has a decrease of as much as 0.4%. This can be because IBM05 has a special structure and does not have any macro cells.

4.2.2 Fixed Cells

A main difference between the ICCAD04 and ISPD05 benchmarks is the fact that the ISPD05 benchmarks have fixed cells. The existence of fixed cells can affect the lengths of nets that are immediately connected to, or are in the neighborhood of such a cell. Placement data confirming this increase in length are shown in Figure 4.3, where the

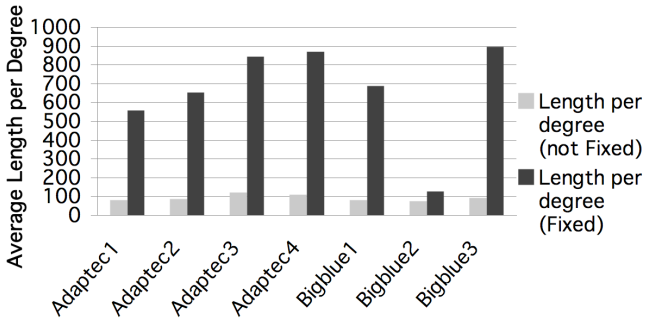


Figure 4.3: Comparison between the average length per degree of nets connected to pins of one or more fixed cells and nets not connected to pins of fixed cells

average length per degree of nets in the neighborhood of fixed cells is compared to the average length per degree of those nets which are only connected to the pins of movable cells.

From Figure 4.3, it can be seen that the average length per degree for the nets which are connected to the pins of one or more fixed cells is about 5 times longer, on average, than nets which are not connected to the pins of the fixed cells in almost all circuits. In the circuit Bigblue2, the average length per degree for nets connected to the pins of one or more fixed cells is about twice that of nets not connected to the pins of fixed cells. This can be because even though this circuit has the biggest number of fixed cells, all these cells are relatively small and not much bigger than standard cells. Therefore, in this case the lengths of the nets that are connected to pins of one or more fixed cells have not increased dramatically. It can be concluded that, on average, the length of nets of a certain degree, in the neighborhood of fixed cells is significantly longer than that of the other nets of the same degree.

It is proposed to take into account the effects of fixed cells on the wire length by multiplying the new base length of nets connected to pins of at least one fixed cell, by a constant. It should be mentioned that the new base length for nets with degree two,

Table 4.4: Comparison between correlation coefficients of the estimated lengths to the actual lengths for ISPD05 circuits with $\alpha_0 = 1$ (no fixed cell impact) and $\alpha_0 = 5$ using Capo10.5

Circuit	Corr. $\alpha_0 = 1$	Corr. $\alpha_0 = 5$	Improvement
Adaptec1	0.597	0.613	2.68%
Adaptec2	0.615	0.627	1.95%
Adaptec3	0.506	0.519	2.57%
Adaptec4	0.525	0.540	2.86%
Bigblue1	0.644	0.648	0.62%
Bigblue2	0.601	0.602	0.17%
Bigblue3	0.487	0.491	0.82%
Bigblue4	0.580	0.588	1.38%
Average	0.569	0.579	1.63%

which are connected to non-standard cells, is already multiplied by a constant. Since all the fixed cells are non-standard cells, degree-two nets are not included in this process.

A new variable, $x'_{2\alpha}$, is proposed to account for the impact of fixed cells on net lengths and is calculated as

$$x'_{2\alpha} = \alpha x'_2, \quad \text{where } \alpha = \begin{cases} 1 & \text{if the net is not connected to any fixed cells,} \\ \alpha_0 & \text{otherwise,} \end{cases}$$

where, α_0 is a number greater than one, which is used to increase the base length of a net connected to the pins of one or more fixed cells.

In this work, it is proposed to set the value of α_0 to 5 which is calculated by finding the ratio of the average length of the nets used in the training set that are connected to the pins of one or more fixed cells, to the average length of all the training data set of the nets that are not connected to the pins of fixed cells. The calculated value of α_0 can be used for length estimation for any circuit containing fixed cells. To evaluate the effectiveness of this technique, the correlation coefficients for the ISPD05 benchmarks are computed using the calculated value of $\alpha_0 = 5$. The results show that a modest improvement of 1.63% in the correlation coefficient is achieved, on average. Bigblue2

exhibits the smallest improvement when compared to the other circuits. This can be explained by examining Figure 4.3, which shows that a value of α_0 less than 5 is more appropriate for this circuit. It should be mentioned that this improvement is on top of the other improvement and made after using the new base length variable.

4.2.3 Fitting the Best Model

The model presented in (2.5) is quadratic in all of the variables. A quadratic model can fit some variables well. However, not all variables show linear or second order characteristics. In Figure 4.4, the average after placement net lengths versus the average values of the variable Nettint$_{nc}$, for circuit adaptec1, are shown with a solid red line. Because of the sheer number of nets in the circuit, scatter plots can become confusing. To produce this graph, values of Nettint$_{nc}$ are put in 20 equally-sized bins in increasing order. Then, the average actual net length is calculated for each bin and a plot of the average length value versus the average variable value is drawn, i.e. the plot is made of 20 points each showing the average length versus the average variable value for 5% of the data. In the same figure, quadratic and logarithmic models are fitted for the average Nettint$_{nc}$ values with a green dashed line and a black dash-dotted line, respectively. It can be seen that the logarithmic curve in Figure 4.4 can better fit the high values of Nettint$_{nc}$.

A reason for why Nettint$_{nc}$ can be better modeled using a logarithmic relationship can be explained with a simple example, as follows: Consider a degree-two net connecting cells with zero area, for simplicity. By increasing the number of external connections, the value of Nettint$_{nc}$ will increase in direct proportion, by definition. If there are initially zero external connections, the two cells will be placed at the same location resulting in a net of length zero. If each cell now has an external connection, the cells are not expected to be at the same location since they are being "pulled apart" by the external connections. As the number of external connections increases, the length is expected to

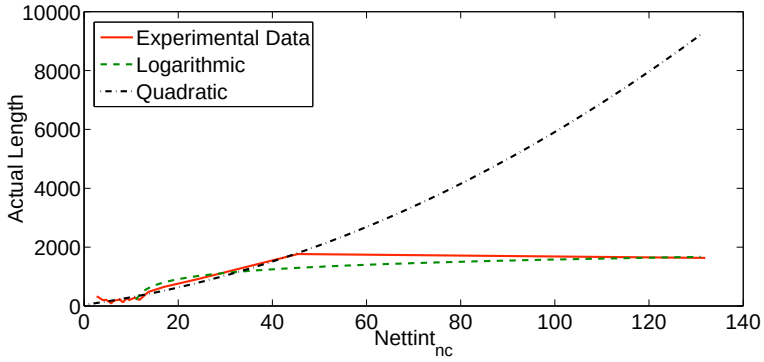
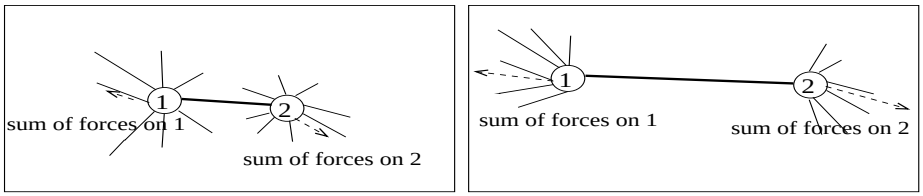


Figure 4.4: Demonstration of actual length versus Nettint$_{nc}$ variable for adaptec1 and quadratic and logarithmic curve fits



(a) Scattered external connections (b) Aligned external connections

Figure 4.5: Illustration of the effects of forces generated by external connections in different directions on net length

increase as well. However, the rate of increase is expected to decay, since the force added by each additional external net is a lower portion of the total force. This tapering of the increase in the length is due to the fact that the location of the connected cells can be all around the circuit, as shown in Figure 4.5 (a), and not in two opposite directions, Figure 4.5 (b). Hence, some forces cancel out.

Since the initial expected length with no external connections is zero, the expected length of the net can be calculated as the sum of the expected increases as each external connection is added. This summation can be formulated as:

$$\ell_{act} \propto \sum_{l=1}^{num_{ext}} f(l),$$

where, ℓ_{act} is the actual length of the net, num_{ext} is the number of external connections, and $f(\cdot)$ is a function which approximates the expected increase in length as each external connection is added. An upper bound for $f(\cdot)$ is $\frac{1}{l}$ since any sequence which decays as or more slowly than $\frac{1}{l}$ would diverge, implying an infinite length, which cannot occur. Therefore, the expected value of L as the number of cells increase can be proportional to:

$$\ell_{act} \propto \sum_{l=1}^{num_{ext}} \frac{1}{l} \approx \int_{1}^{num_{ext}+1} \frac{1}{l} dl = \log\left(num_{ext} + 1\right).$$

The above equation shows that the expected length of a net is better modeled using a logarithmic function of the number of its external connections than a quadratic or linear function of the number itself. Since, Nettint_{nc} is a measure of the number of external connections of a net, its effect on net length is better modeled by a logarithmic curve.

To improve the accuracy of the length estimates, the net length estimation technique presented in (2.5) is modified and the variable x_6 is replaced with variable x'_6, where $x'_6 = \log(x_6)$. The correlation coefficients of the variables x_6 and x'_6 to the actual lengths are calculated and compared. These correlation coefficients are presented in Table 4.5. It can be seen that the correlation between the values of this model variable and the actual after placement lengths improved by up to 56% for all but IBM06 which had a slightly lower correlation with the logarithmic variable. On average for ICCAD04 benchmarks there is 19% improvement and for ISPD05 benchmarks there is 25% improvement in the correlation coefficients of the variable values to the actual lengths.

4.2.4 Model Performance

The discussed enhancements are implemented and the new model variables are tabulated in Table 4.6. To evaluate the performance of the model several experiments are performed.

The first evaluation is to consider the correlation coefficients of each variable to the

Table 4.5: Correlation coefficients of Nettint$_{nc}$, x_6, and log(Nettint$_{nc}$), x'_6, to the actual lengths for ICCAD04 and ISPD05 circuits using Capo10.5

Circuit	Corr. Nettint$_{nc}$	Corr. log (Nettint$_{nc}$)	Improvement
IBM01	0.36	0.45	25%
IBM02	0.35	0.55	56%
IBM03	0.42	0.50	20%
IBM04	0.41	0.45	9%
IBM05	0.55	0.63	15%
IBM06	0.59	0.57	-3%
IBM07	0.37	0.48	29%
IBM08	0.49	0.62	27%
IBM09	0.40	0.45	13%
IBM10	0.25	0.32	25%
IBM11	0.38	0.39	3%
IBM12	0.33	0.39	19%
IBM13	0.43	0.44	4%
IBM14	0.33	0.43	30%
IBM15	0.37	0.41	3%
IBM16	0.41	0.46	11%
IBM17	0.29	0.41	40%
IBM18	0.43	0.53	24%
Average	0.40	0.47	19%
Adaptec1	0.36	0.42	18%
Adaptec2	0.44	0.45	1%
Adaptec3	0.29	0.36	26%
Adaptec4	0.29	0.34	18%
Bigblue1	0.30	0.44	47%
Bigblue2	0.31	0.39	28%
Bigblue3	0.22	0.34	55%
Bigblue4	0.30	0.38	28%
Average	0.31	0.39	25%

Table 4.6: The proposed variables for model improvement

New Variable	Name
x'_2	New Base Length
$x'_{2\alpha}$	Fixed Base Length
x'_6	$\log(\text{Nettint}_{nc})$

actual after placement lengths. Since different placers can have different impacts, the actual lengths obtained by three placers: mPL, FastPlace (FP), and Capo are used and the results are presented in Table 4.7. Each row of this table shows the average over all circuits of the correlation coefficient of each variable to the actual net lengths. The first group of columns, 2, 3 and 4, represent the results obtained by mPL, FastPlace and Capo for the ICCAD04 benchmarks, respectively. In the second group of columns, 5, 6 and 7, the averages for the ISPD05 benchmarks for each placer are shown.

Table 4.7: Average correlation coefficients of variables of the model to after placement lengths

Variable	ICCAD04			ISPD05		
	mPL Corr.	FP Corr.	Capo Corr.	mPL Corr.	FP Corr.	Capo Corr.
x_1	0.39	0.34	**0.42**	0.30	0.31	**0.35**
$x'_{2\alpha}$	0.47	**0.52**	0.48	0.35	0.39	**0.41**
x_3	0.43	**0.47**	0.44	0.29	0.27	**0.30**
x_4	0.40	**0.46**	0.41	0.29	0.28	**0.31**
x_5	0.38	0.38	**0.40**	0.29	0.29	**0.31**
x'_6	0.41	**0.46**	0.42	0.35	0.37	**0.39**
x_7	0.40	**0.44**	0.41	0.29	0.27	**0.30**
x_8	0.40	**0.44**	0.41	**0.29**	0.26	**0.29**
x_9	0.39	**0.43**	0.41	0.29	0.27	**0.30**
Average	0.41	**0.44**	0.42	0.30	0.30	**0.33**

The average correlation coefficient of all variables are very close to each other, which means that most of the variables are of the same importance in estimating the net lengths. The correlation coefficients of the variables to the after placement lengths presented in Table 4.7, on their own, are not high. Part of the reason for the low correlations is

that each variable used in the length estimation model covers only a certain portion of characteristics of a net. For example, the net degree variable covers the effects of the number of cells connected to a net but it does not include the effects of the second level neighbors of a net or the composition of its neighbors.

According to Table 4.7, the correlation coefficients of each variable to the actual after placement lengths for the ISPD05 circuits are lower compared to those from the ICCAD04 circuits. This is due to significant differences between these two sets of circuits. In Table 4.8, a statistical comparison of ICCAD04 and ISPD05 Circuits is performed and some of their differences are highlighted. The total numbers of nets and cells in each circuit are presented in Columns 2 and 3 of Table 4.8, respectively. In Columns 4 and 5 of this table, the number of fixed cells (unmovable) and the percentage of the total chip area that fixed cells occupy are given. In the ISPD05 benchmarks, a large number of fixed cells exist. The ICCAD04 circuits do not contain any fixed cells, except for the I/O pads. In addition, the fixed cells cover a large portion of the placement area since they are all macro cells. These fixed cells can make significant changes in the lengths of the nets in their neighborhood, where nets might be forced to go over or around the fixed cells in order to connect two cells. In Columns 6 and 7, the average and maximum net degrees of the circuits are given. Even though the average net degrees for the two sets of benchmarks are close, the ISPD05 circuits contain some nets that have very high degrees compared to those in the ICCAD04 benchmarks, as shown in Column 7. These high degree nets make it harder to predict the net lengths than when the model deals exclusively with lower degree nets.

Even though individual variables do not show high correlation to the actual lengths, the estimated net lengths found by the proposed model after applying all of the suggested improvements have high correlation to the actual after placement wire lengths. This can be seen in Table 4.9 where the correlation coefficients of the estimated lengths to

Table 4.8: Statistical comparison of ICCAD04 and ISPD05 circuits

Circuit	# Nets	# Cells	# Fixed Cells	Fixed Area	Average Net Degree	Maximum Net Degree
IBM01	14111	12752	0	0.00%	3.58	42
IBM02	19584	19601	0	0.00%	4.15	134
IBM03	27401	23136	0	0.00%	3.41	55
IBM04	31970	27507	0	0.00%	3.31	46
IBM05	28446	29347	0	0.00%	4.44	17
IBM06	34826	32498	0	0.00%	3.68	35
IBM07	48117	45926	0	0.00%	3.65	25
IBM08	50513	51309	0	0.00%	4.06	75
IBM09	60902	53395	0	0.00%	3.65	39
IBM10	75196	69429	0	0.00%	3.96	41
IBM11	81454	70558	0	0.00%	3.45	24
IBM12	77240	71076	0	0.00%	4.11	28
IBM13	99666	84199	0	0.00%	3.58	24
IBM14	152772	147605	0	0.00%	3.58	33
IBM15	186608	161570	0	0.00%	3.84	36
IBM16	190048	183484	0	0.00%	4.10	40
IBM17	189581	185495	0	0.00%	4.54	36
IBM18	201920	210613	0	0.00%	4.06	66
Average	87242	82194	0	0.00%	3.84	44
Adaptec1	221142	211447	543	63.17%	4.27	2271
Adaptec2	266009	255023	566	81.36%	4.02	1935
Adaptec3	466758	451650	723	82.63%	4.02	3713
Adaptec4	515951	496045	1329	77.18%	3.71	3974
Bigblue1	284479	278164	560	45.99%	4.02	2621
Bigblue2	577235	557866	23084	62.22%	3.68	11869
Bigblue3	1123170	1096812	1293	78.08%	3.41	7623
Bigblue4	2229886	2177353	8170	57.65%	3.99	20766
Average	710579	690545	4533.5	68.53%	3.89	6847

Table 4.9: Correlation coefficients of the estimated net lengths to the actual lengths produced by different placers and comparison to previous net length estimation techniques after applying all the suggested improvements to the proposed model

Circuit	Proposed Model			Improvement Over Given Model			
				Fathi [2]			Bodapati [1]
	mPL Corr.	FP Corr.	Capo Corr.	mPL	FP	Capo	Capo
IBM01	0.68	0.72	0.70	16.6%	10.0%	18.1%	42.2%
IBM02	0.76	0.81	0.74	4.0%	4.5%	5.4%	34.2%
IBM03	0.69	0.75	0.72	6.2%	5.5%	4.3%	24.1%
IBM04	0.65	0.75	0.65	8.3%	6.0%	7.0%	20.9%
IBM05	0.74	0.76	0.77	1.2%	1.8%	1.1%	23.9%
IBM06	0.84	0.84	0.85	2.4%	0.7%	2.2%	34.6%
IBM07	0.59	0.64	0.61	6.7%	6.0%	6.3%	26.3%
IBM08	0.72	0.83	0.73	1.3%	0.1%	1.5%	16.0%
IBM09	0.59	0.66	0.63	5.3%	4.9%	6.3%	20.6%
IBM10	0.51	0.61	0.50	9.7%	8.7%	11.6%	22.4%
IBM11	0.54	0.62	0.59	7.8%	5.5%	4.5%	33.0%
IBM12	0.51	0.64	0.54	2.8%	2.3%	4.9%	33.8%
IBM13	0.60	0.64	0.61	3.5%	5.2%	3.1%	32.2%
IBM14	0.54	0.56	0.55	4.9%	6.4%	1.9%	27.9%
IBM15	0.59	0.61	0.58	4.1%	6.3%	6.2%	24.3%
IBM16	0.60	0.61	0.58	3.4%	3.3%	3.6%	20.8%
IBM17	0.46	0.51	0.48	7.7%	5.6%	5.2%	21.0%
IBM18	0.65	0.64	0.66	3.3%	5.6%	2.7%	13.3%
Average	**0.63**	**0.68**	**0.64**	**5.5%**	**4.9%**	**5.3%**	**26.2%**
Adaptec1	0.55	0.55	0.62	7.6%	7.1%	7.7%	-
Adaptec2	0.54	0.54	0.63	3.4%	2.0%	4.6%	-
Adaptec3	0.48	0.51	0.53	7.6%	9.2%	6.3%	-
Adaptec4	0.57	0.51	0.55	2.9%	5.4%	4.0%	-
Bigblue1	0.59	0.56	0.65	5.7%	3.2%	4.1%	-
Bigblue2	0.55	0.60	0.60	4.6%	3.8%	3.5%	-
Bigblue3	0.48	0.46	0.49	4.4%	2.7%	5.5%	-
Bigblue4	0.49	0.45	0.59	4.9%	3.5%	3.9%	-
Average	**0.53**	**0.52**	**0.58**	**5.1%**	**4.6%**	**4.9%**	-

the actual net lengths found by the mPL, FastPlace and Capo placers are provided for both sets of benchmarks, in Columns 2, 3 and 4, respectively. These results have been compared to two of the most recent and detailed models in the literature, proposed by Fathi [2] and Bodapati [1]. In Columns 5, 6 and 7, the percentage improvements over the model in [2] are reported for each of the placers, which show a positive improvement for every circuit, with the averages around 5%. The percentage improvement over the model in [1] is presented in Column 8. The results show a 26.2% improvement on average and up to 42.2% improvement for circuit IBM01.

4.3 Radial Basis Function-Based Net Length Estimation

In this research, a new net length estimation technique which is based on Radial Basis Functions (RBFs) is developed. As the data available for length estimation are highly non-linear, fitting a specific model such as the polynomials used in [1, 2] does not always yield acceptable results. An example of this problem is shown in Section 4.2.3 for the model variable $Nettint_{NC}$ where it is shown that a polynomial model is not adequate to model this variable. In this section, it is proposed to employ RBFs for net length estimation to overcome the inadequacies of polynomial fitting. In addition, a new technique is developed to tune the shapes of the Gaussian RBFs by finding the proper variance parameter.

4.3.1 Algorithm Overview

There are two main components in an estimation technique. One is specifying the model structure, where one decides which model variables should be used and how these variables can be modeled. Several model variables that affect the length of a net and which can be calculated before placement are used as the parameters in this research. These variables include the seven variables used in [2] along with two improved variables pre-

sented in Section 4.2. In Table 4.10, these variables are shown.

Table 4.10: The variables used in the proposed net length estimation model

Variable	Name	Modeling Role
x_1	net degree	number of cells of a net
$x'_{2\alpha}$	fixed base length	minimum half-perimeter net length considering placer and fixed cells effect
x_3	second level effect	sizes of 2^{nd} level neighbors
x_4	N_{2oth}	effect of degree-two nets
x_5	inv. mutual contraction	connectivity between cells of a net
x'_6	$\log(\text{Nettint}_{nc})$	common and uncommon nets between cells of a net (logarithmic scale)
x_7	degree two to four	effects of other nets in the
x_8 , x_9	congestion metrics	neighborhood of a net

The second main component of an estimation technique is to determine the parameters of the model used in the estimation. In a polynomial model, these parameters are the polynomial coefficients. In an RBF-based estimation model, the parameters comprise the center locations, $\boldsymbol{c_i} \in \boldsymbol{C}$, the variance parameter of the distribution functions, σ^2, and the weights of the RBFs, $w_i \in \boldsymbol{w}$. In this research, Gaussian RBFs are applied for net length estimation and a variance selection method suited for mixed-size ICs is proposed. The locations of centers are calculated using Algorithm Center that is discussed in detail in Section 2.2.5.

The proposed estimation technique uses Algorithm Estimation that is shown in Figure 4.6 and is explained in the following.

Algorithm Estimation: RBF-based net length estimation

Inputs: Subset of actual lengths, $\boldsymbol{\ell_{act}}$, and
Model variables, $\boldsymbol{X}$
Output: Estimated lengths, $\boldsymbol{\ell_{est}}$

1. Find center locations $\boldsymbol{c_i} \in \boldsymbol{C}$ using Algorithm Center
2. Calculate variance, σ^2, of the RBFs
3. Find the estimated lengths, $\boldsymbol{\ell_{est}}$, using Algorithm RBF

Figure 4.6: High-level algorithm for RBF-based net length estimation

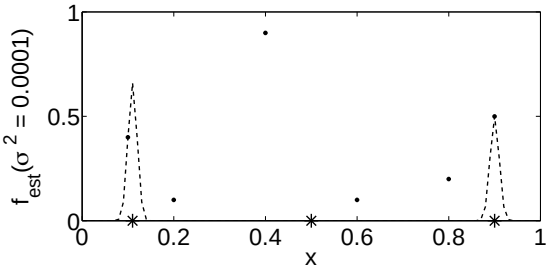
The input to the algorithm is a set of training data which includes a subset of the actual net lengths and the values of all of the model variables. Estimated lengths of individual nets, $\ell_{est}(net_j)$, are the outputs of the algorithm.

In Step 1 of Algorithm Estimation, the proper locations of the centers are selected using the center placement method, Algorithm Center, discussed in Section 2.2.5. In Step 2, an appropriate value for the variance parameter of the RBFs is chosen. The proposed method for variance selection is discussed in Section 4.3.2. Finally, in Step 3, the weights of RBFs are estimated and the estimated net lengths, $\boldsymbol{\ell_{est}}$, are obtained using Algorithm RBF, discussed in Section 2.2.4, with Gaussian distribution functions.

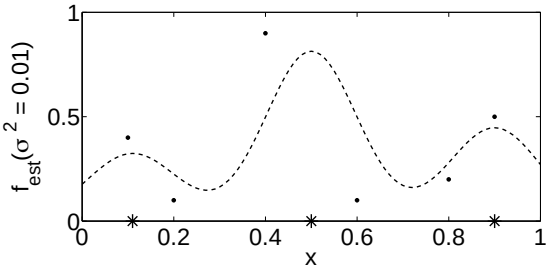
4.3.2 Variance Selection

Choosing the right variance parameter for the Gaussian distribution functions used in the proposed RBF-based length estimation model is very important. To better show this importance, an example is given in Figure 4.7. Three separate estimates of six given data points using three centers are presented for three different values of the variance parameter of the Gaussian distributions. In this figure, the given data points are shown by $\bullet$, the center locations are represented by $*$ and the dotted line is the function estimate. The variances are set to be 0.0001, 0.01 and 0.1 to illustrate how the variance parameter can affect the estimation results. In Figure 4.7(a), it is shown that using a variance of 0.0001 results in undesirable sharp changes in the estimated function. Hence, data points located between two neighbor centers cannot be properly evaluated. On the other hand, in Figure 4.7(c), using 0.1 as the variance, the estimated function is too smooth and unable to follow the changes in the given data trends. The function estimate resulting from using a variance of 0.01 is shown in Figure 4.7(b). In this figure, the function estimate can better capture details as well as the trends.

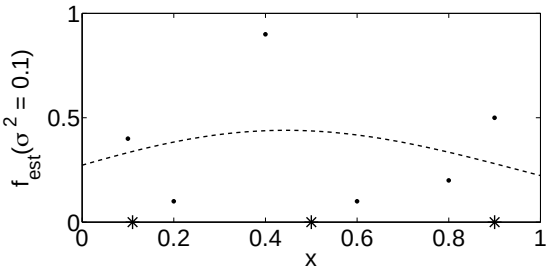
In this research, a method for finding the variance, σ^2, is proposed. In Section 2.2.5,



(a) Variance = 0.0001



(b) Variance = 0.01



(c) Variance = 0.1

Figure 4.7: Comparison of estimation using RBFs with different variances. Given data points are shown by $\bullet$, the center locations are represented by $*$ and the dotted line is the function estimate.

the radius r used in the constructive selective center placement method is set to be $\frac{\sqrt{n_{var}}}{4}d_c$. This implies that the maximum distance between each data point and its nearest center is r. Considering that in a Gaussian distribution, one standard deviation around the center contains about 68% of the distribution, if a data point is within one standard deviation of a center, the estimated value of that point is significantly determined by the distribution function related to that center. Therefore, $\frac{\sqrt{n_{var}}}{4}d_c$ is chosen for the standard deviation. In addition, selecting $\frac{\sqrt{n_{var}}}{4}d_c$ as the standard deviation makes the radial basis functions of neighboring centers overlap significantly while they do not considerably overlap with functions of non-adjacent centers. This prevents the estimated set of data to be overly-smooth while it is still able to capture some of the details along with the trends in the data. Considering that in the proposed RBF-based length estimation model nine variables are used, $n_{var} = 9$, the standard deviation, σ, is calculated as follows:

$$\sigma = \frac{\sqrt{9}}{4}\ d_c = \frac{3}{4}\ d_c.$$

The experimental results, which are provided in Section 4.3.3, support the selected standard deviation. The variance can then be calculated by squaring the standard deviation:

$$\sigma^2 = \frac{9}{16}\ {d_c}^2. \tag{4.2}$$

4.3.3 Experimental Results

In this section, several experiments are presented that validate the variance selection method and RBF-based length estimation technique, both of which are proposed in this research. Benchmark circuits from the ICCAD04 and ISPD05 benchmarks are used in the experiments. The number of center grids is determined by performing a linear search from 2 to 20 on the benchmark circuits. Using 10 center grids represents a good trade-off between the number of final centers and the correlation coefficients obtained over all the benchmarks. Therefore, the number of center grids is set to 10 in the variance

selection and model performance experiments. All experiments are performed using a $2.93GHz$ Intel® Xeon® X7350 server. The algorithm development environment is 64-bit $MATLAB$ 7.8.0.

Variance Selection Results

The proposed variance selection procedure is discussed in detail in Section 4.3.2. An experiment is performed in which the RBF-based estimation model is constructed using 10 center grids on each dimension. According to Section 4.3.2, the standard deviation is calculated to be $\sigma = \left(\frac{\sqrt{9}}{4} \times \frac{1}{10-1}\right) = \frac{1}{12}$. A linear search is performed around the calculated value to validate the theoretically-obtained standard deviation. The correlation coefficient of the estimated lengths to the actual lengths is used as a metric to evaluate the estimation model performance. In Figure 4.8, this correlation coefficient versus standard deviation is plotted. It can be seen that the experiment supports the theoretically-calculated value for standard deviation from which the variance parameter for the RBFs is calculated to be $\sigma^2 = \frac{1}{144}$.

Model Results

To show the effectiveness of the proposed RBF-based net length estimation technique, the estimation results are compared with the results of the best model for mixed-size circuits available in the literature [2]. At first, to have a fair comparison, the same percentage of data, 50%, is used for training while the other 50% are used to validate the estimation model. In addition, to better show the capabilities of the proposed model, another experiment is performed with only 10% of data for training and the other 90% for validation. This additional experiment is carried out to better show the effectiveness, sensitivity and generality of the proposed model. Correlation coefficients of the estimated lengths to the actual after placement lengths are used to compare the outputs of the estimation models. The actual after placement lengths are produced by the Capo10.5

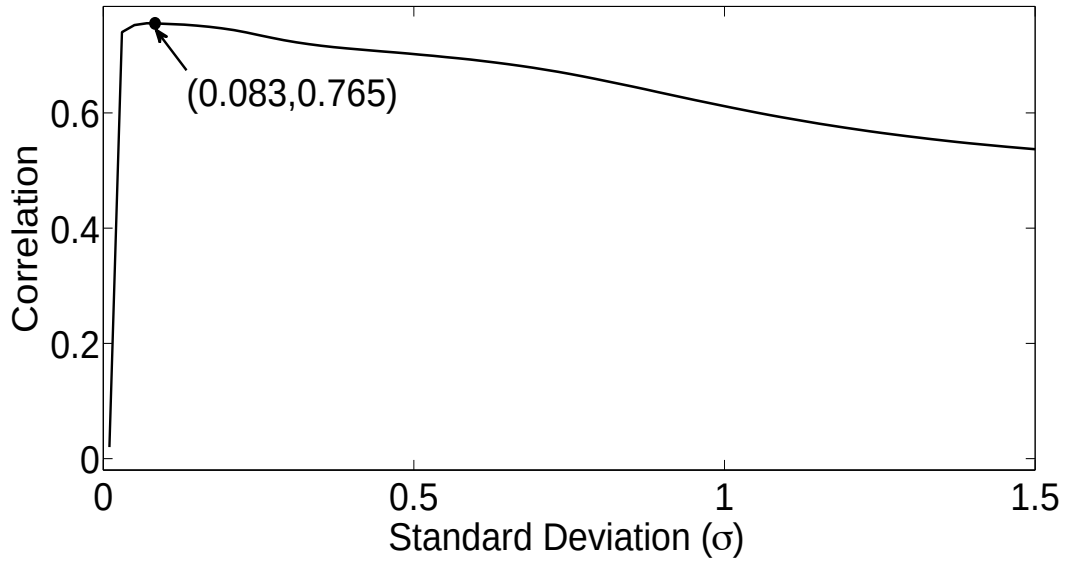


Figure 4.8: Linear search to verify the calculated standard deviation for RBFs

placer [59], which is a partitioning-based placer, and the mPL6 placer [50], an analytical placer. The experiments are performed on the ICCAD04 and ISPD05 benchmarks. The variance parameter for RBFs is set to $\frac{1}{144}$, the number of center grids on each dimension is 10, and the radius, r, for the constructive selective uniform center placement method is $r = \left(\frac{\sqrt{9}}{4} \times \frac{1}{10-1}\right) = \frac{1}{12}$.

The results are tabulated in Tables 4.11 and 4.12 where the actual lengths used to measure the quality of estimated lengths, are produced by the Capo10.5 and mPL6 placers, respectively. In each table, columns 2-4 include the correlation coefficients of the estimated lengths to the actual lengths for the model in [2] and the proposed model with 50% and 10% training data, respectively. In Columns 5 and 6, the percentage improvement of the proposed RBF-based model over the model of [2] is shown. With 50% training data improvement is achieved for all the benchmarks. For the ICCAD04 circuits and for both placers, an average improvement of around 16% is found and the maximum improvement exceeds 30%.

Since the ISPD05 circuits have variables and net lengths with wide and sparse ranges, estimating their lengths is more challenging. However, for these circuits, the proposed RBF-based estimation model results are improved by 6.17% on average, compared to the other model when the actual lengths are produced by Capo10.5 placer. In addition, a maximum improvement of around 10% is observed for the circuit bigblue1. When the mPL6 placer is used for the placement and hence to compute the actual lengths, an average improvement of 8.36% and a maximum improvement of 14% are seen over the model in [2] for the ISPD05 circuits.

The experiment with 10% training data verifies that the proposed model can compete with the best estimation model for mixed-size circuits existing in the literature with only one-fifth of the training data. It can be seen that for the majority of the benchmarks, improvement is achieved and the average improvement is still significant, 5% for ICCAD04 circuits and around 2% for ISPD05 circuits.

4.4 Summary

In this chapter, an RBF-based net length estimation technique is proposed. This technique is an individual a-priori length estimation technique. The effects of different placer algorithms on the net lengths along with the effects of the existence of fixed cells in ICs on net lengths are considered in the development of this technique. An algorithm is proposed which estimates lengths of individual nets using RBFs. In addition, a new method is proposed to calculate the proper value for the variance parameter of radial basis functions. To justify all of these improvements and proposed methods, experiments are provided that illustrate the effectiveness of these proposals.

Table 4.11: Comparison of the correlation coefficients of estimated net lengths to the actual lengths produced by Capo10.5 placer using the proposed RBF-based model with 50% and 10% of data as the training data set and the model presented in [2]

Circuit	Capo10.5				
	Model in [2]	Proposed Model		Percentage Improvement	
		50%	10%	50%	10%
IBM01	0.59	0.77	0.65	30.51%	10.17%
IBM02	0.70	0.75	0.71	7.14%	1.43%
IBM03	0.69	0.77	0.72	11.59%	4.35%
IBM04	0.61	0.66	0.62	8.20%	1.64%
IBM05	0.76	0.85	0.74	11.84%	-2.63%
IBM06	0.83	0.84	0.81	1.20%	-2.41%
IBM07	0.57	0.71	0.64	24.56%	12.28%
IBM08	0.72	0.77	0.75	6.94%	4.17%
IBM09	0.59	0.73	0.67	23.73%	13.56%
IBM10	0.45	0.57	0.50	26.67%	11.11%
IBM11	0.56	0.69	0.60	23.21%	7.14%
IBM12	0.51	0.58	0.53	13.73%	3.92%
IBM13	0.59	0.67	0.58	13.56%	-1.69%
IBM14	0.54	0.67	0.62	24.07%	14.81%
IBM15	0.55	0.62	0.57	12.73%	3.64%
IBM16	0.56	0.62	0.61	10.71%	8.93%
IBM17	0.46	0.55	0.50	19.57%	8.70%
IBM18	0.64	0.69	0.59	7.81%	-7.81%
Average	0.61	0.69	0.63	15.43%	5.07%
adaptec1	0.57	0.61	0.59	7.02%	3.51%
adaptec2	0.60	0.64	0.61	6.67%	1.67%
adaptec3	0.50	0.54	0.53	8.00%	6.00%
adaptec4	0.51	0.52	0.51	1.96%	0.02%
bigblue1	0.62	0.68	0.65	9.68%	4.84%
bigblue2	0.57	0.59	0.55	3.51%	-2.40%
bigblue3	0.47	0.50	0.47	6.38%	0.57%
Average	0.55	0.58	0.56	6.17%	2.03%

Table 4.12: Comparison of the correlation coefficients of estimated net lengths to the actual lengths produced by mPL6 placer using the proposed RBF-based model with 50% and 10% of data as the training data set and the model presented in [2]

Circuit	mPL6				
	Model in [2]	Proposed Model		Percentage Improvement	
		50%	10%	50%	10%
IBM01	0.58	0.74	0.62	27.59%	6.90%
IBM02	0.73	0.78	0.78	6.85%	6.85%
IBM03	0.65	0.74	0.67	13.85%	3.08%
IBM04	0.60	0.64	0.61	6.67%	1.67%
IBM05	0.73	0.84	0.72	15.07%	-1.37%
IBM06	0.82	0.82	0.80	0.00%	-2.44%
IBM07	0.55	0.72	0.62	30.91%	12.73%
IBM08	0.71	0.76	0.74	7.04%	4.23%
IBM09	0.56	0.70	0.63	25.00%	12.50%
IBM10	0.46	0.55	0.50	19.57%	8.70%
IBM11	0.50	0.65	0.55	30.00%	10.00%
IBM12	0.50	0.54	0.51	8.00%	2.00%
IBM13	0.58	0.67	0.57	15.52%	-1.72%
IBM14	0.51	0.67	0.60	31.37%	17.65%
IBM15	0.56	0.61	0.57	8.93%	1.79%
IBM16	0.58	0.63	0.61	8.62%	5.17%
IBM17	0.43	0.54	0.48	25.58%	11.63%
IBM18	0.63	0.68	0.59	7.94%	-6.35%
Average	0.59	0.68	0.61	16.03%	5.17%
adaptec1	0.51	0.54	0.52	5.88%	2.01%
adaptec2	0.53	0.58	0.55	9.43%	4.49%
adaptec3	0.45	0.49	0.46	8.89%	2.60%
adaptec4	0.53	0.54	0.53	1.89%	-0.74%
bigblue1	0.56	0.64	0.58	14.29%	3.57%
bigblue2	0.53	0.58	0.51	9.43%	-3.11%
bigblue3	0.46	0.50	0.47	8.70%	3.06%
Average	0.51	0.55	0.52	8.36%	1.70%

Chapter 5

Proposed Net Length Estimation Model Application

5.1 Introduction

The proposed individual length estimation model can have several applications to empower circuit designers to improve the quality of their designs. Among numerous applications associated with net length estimation, physical-driven synthesis, synthetic benchmark generation, field-programmable gate array routing estimation and technology extrapolation can benefit the most from a-priori length estimation of interconnects [8]. Accurate prediction of interconnection length can help estimate the actual layout area and evaluate the fit of a logic design to a fabrication technology. Also, with the knowledge of predicted wire length, a quick estimate of the necessary wiring space and routing difficulty can be performed in the early design planning stage [61].

The proposed application of the net length estimation technique in this research is to use it as a tool to better understand the effects of pre-placement clustering on individual nets and hence to improve the overall clustering quality. Clustering algorithms used before placement try to reduce the overall length of wires by grouping cells that are closely related. However, forming one cluster can negatively impact the length of the nets that are in its neighborhood. In this research, it is proposed to use the a-priori individual length estimation technique developed in Chapter 4 in a predictor-corrector framework, in order to improve the results obtained by different clustering algorithms.

To increase the efficiency of a clustering algorithm, it would be useful to know which nets are likely to increase in length as a result of clustering and then try to cluster the cells of these nets in order to reduce their length increase. However, the lengths of nets

are not known before placement. In the predictor-corrector framework, it is proposed to use the a-priori length estimation technique described in this thesis to find which properties in a net can cause an increase in the chances of that net getting stretched during clustering. The predictor-corrector, which is done after clustering, is divided into two steps:

- In the first step, referred to as the predictor step, by finding characteristics of nets that can cause increases in the net length, the nets which are expected to have large length increases during the clustering stage are identified.
- In the second step, the corrector, clustering with a specific objective is applied to improve the quality.

In this application, the effects of clustering on nets are studied. Variables that can contribute to a net being stretched during clustering are identified, and a predictor-corrector framework that can reduce the number of nets being stretched is proposed.

The rest of this chapter is organized as follows: In Section 5.2, a study of the effects of clustering techniques on net length is performed and the negative side effects of clustering algorithms on the lengths of individual nets are explained. A predictor-corrector framework is proposed in Section 5.3 to handle those negative effects. Finally, in Section 5.4, the chapter is summarized.

5.2 Effects of Clustering on Individual Wire Lengths

Pre-placement clustering algorithms are designed to reduce the sizes of integrated circuits in order to simplify the placement stage. Using these algorithms is shown to be effective in decreasing the total wire length required for routing a circuit [11]. However, several individual nets experience significant increases in their lengths. Therefore, the impacts of clustering on individual net lengths should be studied.

In order to better understand the effects of clustering, after placement individual net lengths, with and without clustering, are compared. First, placement is performed on the whole circuit and individual net lengths are computed. Then, one level of clustering is performed, where each circuit is clustered to 70% to 60% of its original size, i.e. cell clustering ratio (CCR) is between 70% to 60%. This CCR is chosen since the normal circuit size reductions are between 70% to 60% of the original size at each level. Two different clustering algorithms, Best-Choice clustering (BC) [47] and Net Cluster clustering (NC) [11] are used. These algorithms are described in Sections 3.5.3 and 3.5.4, respectively. BC and NC are chosen since they are score-based algorithms that have shown some of the best clustering results [62].

Placement is performed on the clustered circuit, and the lengths of individual nets are calculated and compared with the lengths obtained when no clustering had been performed. To have a closer look at the impact of clustering on net length, the corresponding actual lengths of each net net_j in the after clustered, $\ell_{\mathbf{act_{AC}}}$, and pre-clustered sets, $\ell_{\mathbf{act_{PC}}}$, are compared. A variable called $compL_{clu}(net_j)$, for each net net_j, is introduced as,

$$compL_{clu}(net_j) = \frac{\ell_{act_{AC}}(net_j)}{\ell_{act_{PC}}(net_j) + \ell_{act_{AC}}(net_j)}. \tag{5.1}$$

Based on this equation, if the value of $compL_{clu}(net_j)$ is less than 0.5, it means that the length of net_j decreased as a result of clustering, which is desired. Similarly, if the value of $compL_{clu}(net_j)$ is greater than 0.5, it means the length of net_j increased as a result of clustering, which is undesirable. If $compL_{clu}(net_j)$ is equal to 0.5, it means that the length of net_j did not change during clustering.

In Tables 5.1 and 5.2, the statistics of the variable $compL_{clu}$ are presented for the ICCAD04 benchmarks that are clustered using BC and NC, respectively. In all of these experiments, the individual net lengths are found using the Capo placer. In each table, the maximum, minimum and mean values of the variable $compL_{clu}$ are shown in Columns

2 to 4, respectively. In Column 5, the standard deviation of $compL_{clu}$, $\sigma_{compL_{clu}}$, is presented to show the variation of the variable values.

From Tables 5.1 and 5.2, it can be seen that even though the lengths of nets in most of the circuits have decreased, on average, this reduction is not substantial and for some circuits and some nets there is a large increase. In this thesis, it is proposed to improve the quality of clustering results by applying a corrective clustering.

To better understand the effects of clustering on individual net lengths, the percentage of nets that experience increases in their lengths during clustering are compared to the percentage of nets that experience length decreases. Thus, a variable is defined for each circuit as follows:

$$ratio_{inc-dec} = \frac{per_{inc}}{per_{dec}},$$

where for each circuit, $ratio_{inc-dec}$ is the ratio of percentage of nets that experience increases in their lengths during clustering, per_{inc}, and the percentage of nets that experience length decreases, per_{dec}.

In Table 5.3, the percentages of the nets with increased length after clustering are shown in Columns 2 and 3 for each clustering algorithm. Similarly, the percentages of the nets that experience decreases in their length during clustering are given in Columns 4 and 5 for each clustering algorithm. The average values of variable $ratio_{inc-dec}$ are presented for the ICCAD04 benchmarks when they are clustered using BC and NC in Columns 6 and 7, respectively. In these experiments, Capo is used as the placer.

From this table, it can be seen that even though, on average, around 49% of the nets experienced length decreases, a significant percentage, 44%, of the nets experienced length increases. Most of the length increases are not substantial, but if the nets with the largest increases in length can be identified, they can be targeted by corrective steps to improve the overall quality of the clustering.

From the data presented in Tables 5.1, 5.2 and 5.3, it might seem that clustering

Table 5.1: Comparison of after and before clustering net lengths. These results are obtained by Best-Choice clustering, using Capo for placement.

Circuit	Maximum $compL_{clu}$	Minimum $compL_{clu}$	Average $compL_{clu}$	$\sigma_{compL_{clu}}$
IBM01	0.999	0.001	**0.500**	0.211
IBM02	1.000	0.000	0.440	0.229
IBM03	1.000	0.000	0.461	0.232
IBM04	1.000	0.001	0.481	0.230
IBM05	0.999	0.001	0.492	0.224
IBM06	1.000	0.000	**0.506**	0.229
IBM07	1.000	0.000	**0.502**	0.228
IBM08	1.000	0.000	0.496	0.224
IBM09	1.000	0.000	0.476	0.215
IBM10	1.000	0.000	0.486	0.224
IBM11	1.000	0.000	0.485	0.220
IBM12	1.000	0.000	0.490	0.222
IBM13	1.000	0.000	0.475	0.222
IBM14	1.000	0.000	0.484	0.222
IBM15	1.000	0.000	0.483	0.218
IBM16	1.000	0.000	0.486	0.214
IBM17	1.000	0.000	0.488	0.216
IBM18	0.999	0.000	0.485	0.208
Average	1.000	0.000	0.484	0.221

Table 5.2: Comparison of after and before clustering net lengths. These results are obtained by Net Cluster clustering, using Capo for placement.

Circuit	Maximum $compL_{clu}$	Minimum $compL_{clu}$	Average $compL_{clu}$	$\sigma_{compL_{clu}}$
IBM01	0.999	0.001	0.498	0.210
IBM02	1.000	0.000	0.439	0.231
IBM03	1.000	0.000	**0.550**	0.265
IBM04	1.000	0.001	0.483	0.228
IBM05	0.999	0.001	0.482	0.222
IBM06	1.000	0.000	0.484	0.216
IBM07	1.000	0.000	0.486	0.214
IBM08	1.000	0.001	0.497	0.222
IBM09	1.000	0.000	0.480	0.212
IBM10	1.000	0.000	0.494	0.222
IBM11	1.000	0.000	0.486	0.217
IBM12	1.000	0.000	0.485	0.216
IBM13	1.000	0.000	0.474	0.223
IBM14	1.000	0.000	0.483	0.218
IBM15	1.000	0.000	0.480	0.217
IBM16	1.000	0.000	0.485	0.214
IBM17	1.000	0.000	0.486	0.215
IBM18	0.999	0.000	0.486	0.205
Average	1.000	0.000	0.487	0.220

Table 5.3: The average values of the ratio, $ratio_{inc-dec}$, of the percentage of nets that experience increases in their lengths during clustering, per_{inc}, and the percentage of nets that experience length decreases, per_{dec}

Circuit	$per_{inc}(\%)$		$per_{dec}(\%)$		Average $ratio_{inc-dec}$	
	BC	NC	BC	NC	BC	NC
IBM01	47	45	45	47	**1.04**	0.96
IBM02	37	37	57	57	0.65	0.66
IBM03	41	54	53	40	0.78	**1.37**
IBM04	44	45	50	50	0.87	0.90
IBM05	44	42	46	47	0.95	0.90
IBM06	49	44	44	49	**1.10**	0.90
IBM07	47	44	45	48	**1.03**	0.91
IBM08	46	46	46	46	0.99	1.00
IBM09	42	43	51	50	0.84	0.85
IBM10	44	46	47	46	0.93	1.00
IBM11	45	45	50	50	0.90	0.90
IBM12	46	45	48	49	0.96	0.92
IBM13	43	43	51	52	0.84	0.83
IBM14	44	44	48	49	0.91	0.89
IBM15	44	44	50	50	0.88	0.87
IBM16	44	44	47	48	0.93	0.91
IBM17	45	44	48	49	0.94	0.90
IBM18	44	44	48	48	0.91	0.91
Average	44	44	49	49	0.91	0.92

is not always successful in reducing wire length. An explanation for this is that using clustering at least can reduce the runtime of the global placer and enable the designer to spend more time on the detailed placement stage and refinements, and hence obtain better overall results. The main goal of this research is to propose techniques that can be used to improve the efficiency of clustering, by focusing on reducing its negative effects.

5.3 A Predictor-Corrector Framework for Clustering

In this section a predictor-corrector framework is proposed to improve the quality of clustering. Before applying the predictor-corrector, a clustering algorithm, such as one of those introduced in Chapter 3, is applied to the circuit to reduce its size. The designer can decide which clustering algorithm is best suited to use and based on the placement algorithm used, how much the circuit size should be reduced. Normal circuit size reductions are between 70% to 60% of the original size at each level.

The predictor-corrector framework is divided into two steps: the predictor and the corrector. Each step is described in the following sections.

5.3.1 The Proposed Predictor Step

The main purpose of this step is to further improve the quality of clustering by finding which global or local properties of nets can cause their lengths to increase. To be able to perform prediction, the estimation model variables shown in Table 4.10 and discussed in Chapter 4, are studied for all nets. Common variables of those nets that have the highest length increases in all benchmarks are identified as variables that can predict net stretching after clustering. These variables can then be used to choose or design a clustering algorithm for the corrector step.

In order to determine which nets have the largest increases in their after clustering lengths, a set of estimation-based experiments are conducted. In these experiments,

the ratio of after clustering lengths to pre-clustering lengths for all nets that are not completely absorbed by clustering are calculated. This ratio is referred to as $nld_{ratio}(net_j)$ and can be calculated for each net net_j as:

$$nld_{ratio}(net_j) = \frac{\ell_{act_{AC}}(net_j)}{\ell_{act_{PC}}(net_j)}. \tag{5.2}$$

Based on this equation, if the value of nld_{ratio} is less than 1, it means that the length of the net decreased as a result of clustering, which is desired. In a similar way, if the value of nld_{ratio} is greater than 1, it means the length increased as a result of clustering, which is undesirable. If nld_{ratio} is equal to 1, it means that the net length did not change during clustering.

It is desired to understand the relation between values of nld_{ratio} for all nets and each variable, $x_1, ..., x_9$, of the estimation model. In addition, it is desired to find out if there exists any variable values for which nld_{ratio} goes above one, i.e. the after clustering length becomes higher than pre-clustering length. For example, a general hypothesis is that the higher a net degree is, the greater the chances are for that net to have a length increase. This hypothesis can be validated if we take the nets that have high values of the variable associated with net degree, x_1, and for these nets calculate the average nld_{ratio}. If this ratio is significantly above one then the hypothesis is likely to be correct.

To be able to find relationships between variable values and increases in length, the graphs of nld_{ratio} for all nets versus the values of each variable are drawn using the same methodology as used for Figure 4.4. In these figures, the nets are sorted in 100 equally-sized bins in an increasing order based on value of the variable which is being considered. For the nets belonging to each bin, the geometric mean of the values of nld_{ratio} are calculated. The geometric mean is used instead of the arithmetic mean since nld_{ratio} expresses a multiplicative change in length and not an additive change.

Examples of these graphs for variables x_1: net degree, $x'_{2\alpha}$: fixed base length, and x_7:

2PinCong, are shown in Figures 5.1 and 5.2 for circuits IBM04 and IBM06, respectively. In these graphs, the abscissa shows the value of the variable on a log-scale and the ordinate is the average value of nld_{ratio} for the bin. From these graphs it can be seen that for small values of each variable, there exist large fluctuations in the values of nld_{ratio}. However, as the values of the variables increase, the magnitude of the fluctuations reduces.

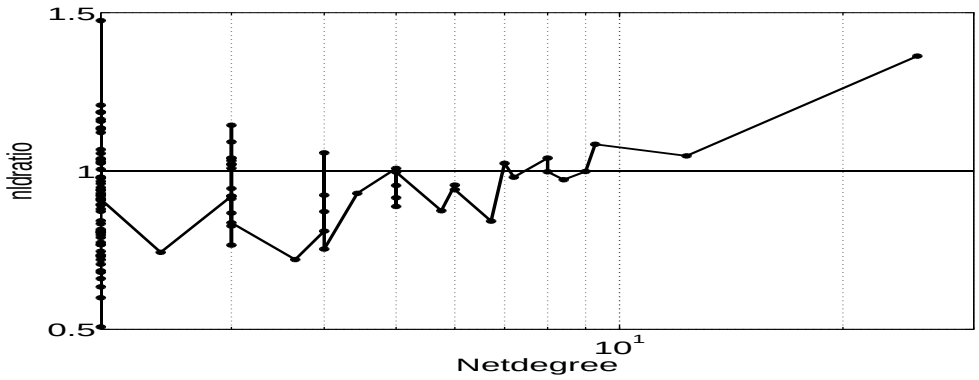
In order to be able to predict which nets need corrective action after each clustering level, the point at which the fluctuation of the nld_{ratio} dies and there is a constant decrease or increase in the lengths is determined. In Table 5.4, the highest percentages for all of the variables from all the circuits are given. Each entry in this table is the number of consecutive bins containing the largest values of the variables where the average nld_{ratio} is consistently higher than one. For example, for IBM01, for each of the top three bins of values of x_1, an average increase in length is seen. Since each bin represents 1% of the total nets, it can be said that the 3% of the nets that have the highest values of x_1 experienced length increases. Zero means that the bin containing the largest values of the variable does not become higher than one. It is proposed to use all variables belonging to the columns that do not have a zero entry, x_3, x_5, x_7, x_8 and x_9, in the corrector step.

5.3.2 The Proposed Corrector Step

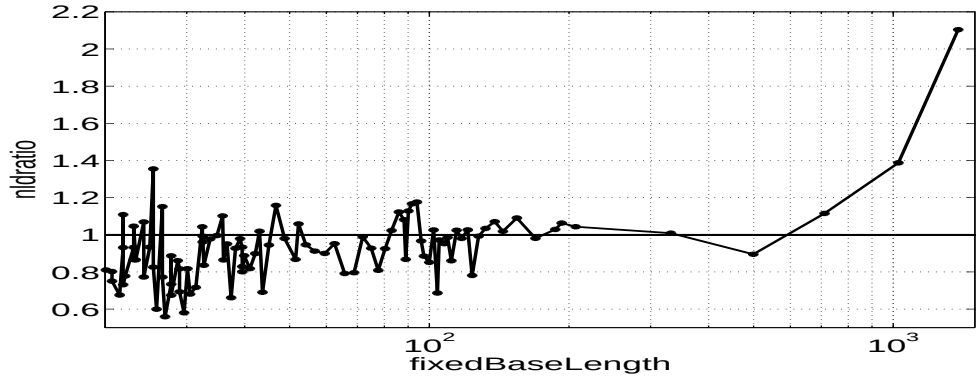
In this step, it is proposed that after each predictor step, a small corrector clustering step is performed where those nets which are expected to have large length increases because they have high values of the identified variables are clustered. These nets are determined using the variables that are identified in the predictor step because they have no zero entries. These variables, referred to as *corrector variables*, are:

x_3: second level effect

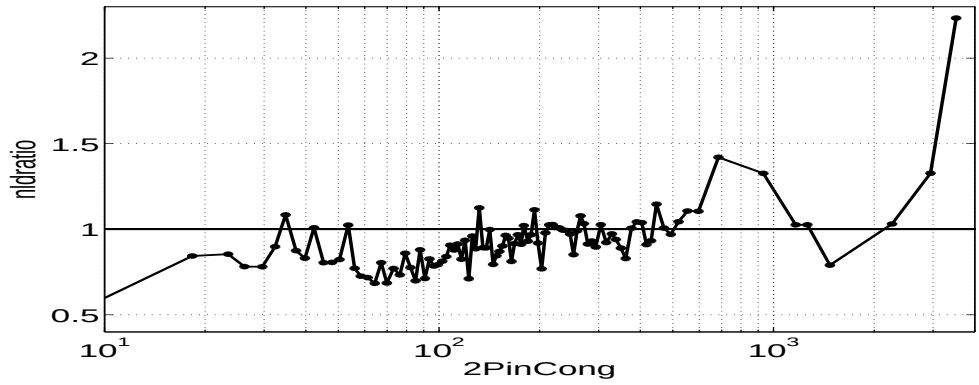
x_5: Inverse mutual contraction



(a) nld_{ratio} versus Net Degree (x_1)

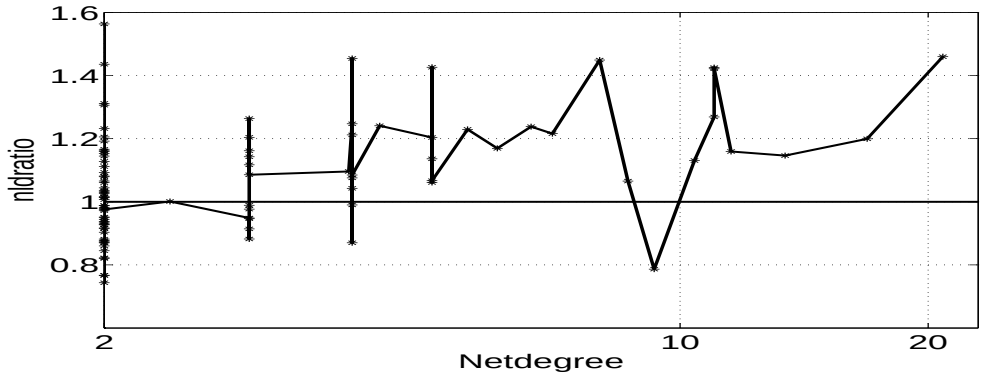


(b) nld_{ratio} versus Fixed Base Length ($x'_{2\alpha}$)

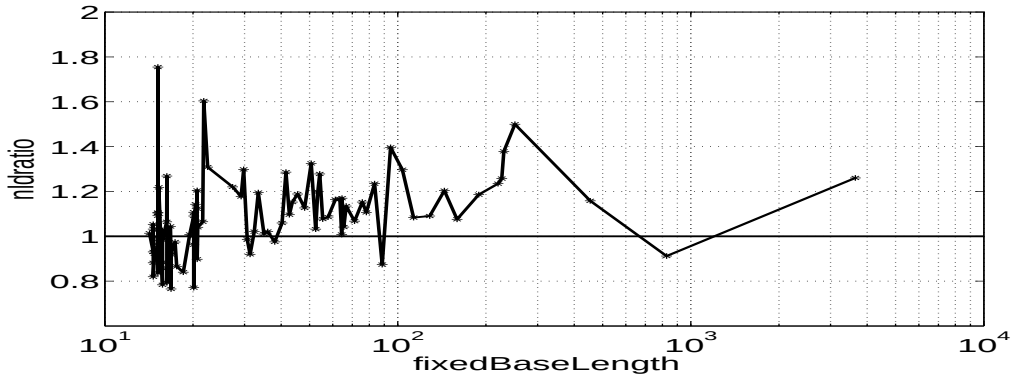


(c) nld_{ratio} versus 2PinCong (x_7)

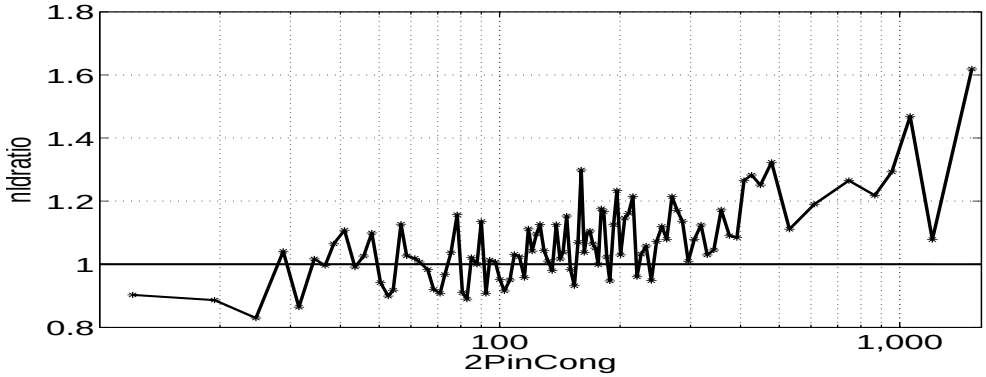
Figure 5.1: Demonstration of the relation between after and before clustering lengths and several estimation variables for IBM04 using log-scale for abscissa



(a) nld_{ratio} versus Net Degree (x_1)



(b) nld_{ratio} versus Fixed Base Length ($x'_{2\alpha}$)



(c) nld_{ratio} versus 2PinCong (x_7)

Figure 5.2: Demonstration of the relation between after and before clustering lengths and several estimation variables for IBM06 using log-scale for abscissa

Table 5.4: Number of consecutive bins containing the largest values of variables with average nld_{ratio} greater than one

Circuit	Variable								
	x_1	$x'_{2\alpha}$	x_3	x_4	x_5	x'_6	x_7	x_8	x_9
IBM01	3	3	2	2	2	1	3	3	6
IBM02	1	3	3	2	1	3	6	7	3
IBM03	3	1	1	9	5	1	1	1	1
IBM04	3	3	2	3	1	2	3	2	2
IBM05	1	1	2	3	1	1	2	3	2
IBM06	7	1	33	15	8	20	26	24	20
IBM07	1	3	2	1	1	1	2	2	4
IBM08	7	1	1	2	3	1	1	1	1
IBM09	1	2	2	1	1	3	3	1	2
IBM10	**0**	**0**	1	**0**	1	**0**	2	2	1
IBM11	1	1	1	2	1	1	1	1	1
IBM12	1	1	1	3	4	3	2	2	2
IBM13	1	6	1	1	1	1	1	1	1
IBM14	5	1	1	1	1	**0**	1	1	1
IBM15	1	1	1	2	1	1	1	1	1
IBM16	1	2	2	5	1	4	1	1	1
IBM17	1	3	10	9	14	7	3	5	3
IBM18	1	10	2	17	8	20	2	13	18
Min	0	0	1	0	1	0	1	1	1
Average	2	2	**3.8**	4	**3.1**	4	**3.4**	**3.9**	**3.9**

x_7: 2PinCong

x_8: 3PinCong

x_9: 4PinCong

Note that most of these variables represent the congestion effects due to other nets in the neighborhood of the net being considered, and do not directly measure the connectivity between cells.

To show the benefits of applying a corrector step, a simple clustering scheme is implemented. In this corrector step, two-pin nets with values of the corrector variables, x_3, x_5, x_7, x_8 and x_9, are in the top one percent are considered for clustering. If these nets have not already been clustered, their cells are put together to form a new cluster.

The improvements in the after placement solutions for the ICCAD04 benchmarks when the corrector step is used after clustering using both NC and BC are shown in Columns 4 and 7 of Table 5.5, respectively. These results show that, on average, there are improvements of 3.4% and 3.7% in the total actual wire length using the NC and BC clustering algorithms, respectively, while maximums of 33.5% and 22.5% are achieved. Since the quality of clustering and placement results is already high, the average improvement in total after placement net length is significant. Also, it should be noted that these improvement are significant since they are 3.4% and 3.7% of the total actual wire length of the whole circuit. In addition, getting improvement of over 20% only with a small clustering ratio for some special cases shows that by studying these cases and applying this information to the other circuits, the placement solutions can be highly improved.

To further validate the proposed predictor-corrector framework, the corrector is applied to the PEKO Benchmark Suite3 [63] after clustering with Net Cluster. These circuits are tested since their optimal wire lengths are known. The PEKO Benchmark

Table 5.5: Comparison of total wire length, produced by Capo, for circuits clustered using Net Cluster and best-choice before and after corrector (Corr.)

Circuit	Net Cluster			Best-Choice		
	Before Corr. ($\times 10^6$)	After Corr. ($\times 10^6$)	Improv.	Before Corr. ($\times 10^6$)	After Corr. ($\times 10^6$)	Improv.
IBM01	2.54	2.49	**2.0%**	2.57	2.57	**0.3%**
IBM02	5.06	5.12	-1.2%	5.27	5.20	**1.4%**
IBM03	7.81	7.29	**7.1%**	7.83	7.76	**0.9%**
IBM04	8.63	8.26	**4.5%**	9.08	8.55	**6.2%**
IBM05	9.96	9.92	**0.4%**	10.09	10.16	-0.7%
IBM06	9.08	6.80	**33.5%**	8.29	7.81	**6.2%**
IBM07	11.07	10.86	**1.9%**	12.09	12.01	**0.7%**
IBM08	13.25	13.10	**1.1%**	13.57	13.45	**0.8%**
IBM09	14.01	14.10	-0.6%	15.06	14.44	**4.3%**
IBM10	33.63	31.29	**7.5%**	39.04	31.88	**22.5%**
IBM11	20.06	20.17	-0.5%	21.29	21.13	**0.8%**
IBM12	35.78	35.46	**0.9%**	41.53	37.44	**10.9%**
IBM13	25.83	25.38	**1.8%**	29.02	26.72	**8.6%**
IBM14	38.34	38.09	**0.7%**	38.51	39.66	-2.9%
IBM15	50.62	51.06	-0.9%	53.19	52.49	**1.3%**
IBM16	59.87	59.31	**0.9%**	65.29	64.07	**1.9%**
IBM17	72.20	70.76	**2.0%**	73.35	72.35	**1.4%**
IBM18	45.50	45.48	0.0%	47.33	46.55	**1.7%**
Average	-	-	**3.4%**	-	-	**3.7%**

Suite3 is used because its circuits have the same number of cells and nets as the ICCAD04 benchmarks, but also include terminal connections. The improvement of the placement results relative to the optimal wire lengths obtained using Net Cluster before and after the corrector are compared and reported in Column 5 of Table 5.6. The results show that an average of 4.0%, and up to 13.8%, improvement relative to the optimal wire lengths is achieved. These results are significant especially after considering that the PEKO circuits do not contain any marco blocks and furthermore all of their cells are the same size while one of the strengths of the proposed estimation model is how it handles macro blocks and cells with different sizes.

5.4 Summary

In this chapter, the negative effects of clustering algorithms are considered. Several experiments are performed to clarify the significance of these negative effects. Then, a predictor-corrector framework for clustering is proposed.

The predictor-corrector framework includes two main steps. In the first step, the prediction step, the nets that are expected to have large increases in their lengths during clustering are identified. To perform the prediction, the estimation model variables are studied and several variables are selected to be utilized in the predictor step.

In the second step, the correction step, the identified nets are targeted by a corrective clustering algorithm. The experimental results support the effectiveness of the proposed predictor-corrector framework as it is seen that the placement results are significantly improved by applying the proposed framework.

Table 5.6: Comparison of total wire length, produced by Capo, for circuits clustered using Net Cluster before and after corrector for PEKO benchmarks

Circuit	Optimal Wire Lengths ($\times 10^6$)	Before Corrector ($\times 10^6$)	After Corrector ($\times 10^6$)	Improvement Relative to Optimal
PEKO01	0.81	1.65	1.63	**2.2%**
PEKO02	1.26	2.22	2.20	**2.6%**
PEKO03	1.50	2.80	2.72	**6.2%**
PEKO04	1.75	3.42	3.39	**1.8%**
PEKO05	1.91	4.31	4.24	**3.1%**
PEKO06	2.06	3.54	3.48	**3.7%**
PEKO07	2.88	5.47	5.16	**13.8%**
PEKO08	3.14	5.94	5.85	**3.1%**
PEKO09	3.64	6.85	6.73	**4.1%**
PEKO10	4.73	8.85	8.65	**5.3%**
PEKO11	4.71	9.25	9.05	**4.8%**
PEKO12	5.00	11.43	11.33	**1.6%**
PEKO13	5.87	11.70	11.62	**1.3%**
PEKO14	9.01	16.04	15.86	**2.5%**
PEKO15	11.50	20.96	20.84	**1.3%**
PEKO16	12.50	22.56	22.36	**2.0%**
PEKO17	13.40	27.04	26.84	**1.5%**
PEKO18	13.20	22.40	21.46	**11.3%**
Average	-	-	-	**4.0%**

Chapter 6

Conclusion and Future Work

6.1 Summary and Contributions

In this thesis, a new a-priori individual net length estimation technique that estimates the lengths of nets before the placement stage, is proposed. In this thesis, it is proposed to use RBFs for length estimation to better capture the non-linearity of the given data sets. As macro and fixed cells are appearing more and more in today's integrated circuits, a new model variable is proposed to consider the effects of the presence of these cells in the length estimation technique. In addition, the proposed technique considers the effects of different placers on the individual net lengths. The quality of the lengths estimated using the proposed RBF-based technique proves its effectiveness on modern mixed-size benchmarks. The performance of the proposed technique is improved by around 15%, on average, for the ICCAD04 benchmarks compared to the best existing technique in the literature for such mixed-size circuits [2].

The other main contribution proposed in this thesis is to design and implement a predictor-corrector framework for clustering. First, the negative side effects of clustering algorithms on individual net lengths are studied. Then, a predictor-corrector framework is proposed to correct these effects. The variables used in the proposed RBF-based net length estimation technique are applied in the predictor step. Then, nets that are highly expected to experience significant length increases during clustering are identified. In the corrector step, the cells belonging to these nets are clustered to avoid net stretching. By applying the predictor-corrector framework, the placement results are improved by around 3.5%, on average. This improvement is quite significant since the quality of the

existing clustering and placement algorithms is already high.

The main contributions of the thesis are listed below:

- A new RBF-based a-priori individual net length estimation technique is proposed and implemented. This is the first time that RBFs have been applied in net length estimation and it is shown that using RBFs is very effective in enabling the estimation technique to better deal with the highly non-linear data encountered in physical design. The quality of length estimates is improved significantly when RBFs are applied.

- A new method for finding the variance parameter for RBFs that is based on the characteristics of the net length estimation data, is proposed. This method makes the RBF-based technique more suitable for capturing the highly non-linear data of integrated circuits.

- A new model variable is proposed to consider the effects of different placers in the net length estimation technique. Since the proposed technique tries to estimate the after placement lengths of nets before the circuit has been placed, the performance of the placement algorithms affects the quality of the length estimates significantly. Therefore, in this thesis, it is proposed to cover these effects by defining a new variable in the estimation technique.

- A new model variable is proposed to consider the effects of the presence of fixed cells in the net length estimation technique. In modern ICs, there exist several fixed cells whose locations are set and cannot be changed during placement. This affects the lengths of the nets that are connected to these cells. Therefore, in the proposed RBF-based technique, the effects of the presence of fixed cells are considered.

- A novel predictor-corrector framework for clustering is proposed and implemented.

Clustering algorithms are used to improve the quality of placement results. However, it is shown that clustering can affect some individual nets negatively. These nets are identified in the predictor step of the framework. In the corrector step, these nets are targeted to improve the performance of the clustering algorithm.

6.2 Future Work

The future work can be classified into two categories based on the main contributions of this thesis:

- Experience with the proposed RBF-based net length estimation technique suggests several future research directions. In this technique, nine different model variables are used. Thus, the technique estimates the net lengths using a set of given data with nine dimensions. Even though a new center placement method is used and a new variance selection method is developed in this thesis, determining the center locations and variances of RBFs are still complicated and time consuming due to the large number of dimensions. By reducing the number of dimensions, the complexity of the proposed RBF-based estimation technique will decrease and the quality of length estimates will be improved. Therefore, future work may consider reducing the number of variables and dimensions. This work could be performed by defining new variables instead of using those nine variables used in this thesis that are mostly taken from the literature. These new variables may combine several properties of the cells and nets of a circuit which results in fewer variables and dimensions. This needs a comprehensive variable analysis over many integrated circuits.

- The predictor-corrector framework also has possibilities for future improvement. The predictor step can be enhanced by making specific conditions for specific nets

and specific circuits. In addition, a new clustering algorithm specifically for the corrector step can be designed. This algorithm can utilize different existing clustering algorithms simultaneously in order to cover all of their advantages.

Bibliography

[1] S. Bodapati and F. Najm. Prelayout estimation of individual wire lengths. *IEEE Trans. on VLSI*, 9(6):943–958, 2001.

[2] B. Fathi, L. Behjat, and L. Rakai. A pre-placement net length estimation technique for mixed-size circuits. In *Proc. of SLIP*, pages 45–52, 2009.

[3] N. Kurd, S. Bhamidipati, C. Mozak, J. Miller, T. Wilson, M. Nemani, and M. Chowdhury. Westmere: A family of 32nm ia processors. In *Proc. of ISSCC*, pages 96 –97, 2010.

[4] N. Sherwani. *Algorithm for VLSI Physical Design Automation, Third Edition.* Kluwer Academic Publishers, Massachusetts, USA, 1999.

[5] S. Sait and H. Youssef. *VLSI Physical Design Automation, Theory and Practice.* IEEE Press, Piscataway, USA, 1995.

[6] W. Donath. Placement and average interconnection lengths of computer logic. *IEEE Trans. on CAS*, 26(4):272–277, 1979.

[7] W. Donath. Wire length distribution for placements of computer logic. *IBM Journal of RD*, 25(3):152–155, 1981.

[8] A. Kahng and S. Reda. Intrinsic shortest path length: a new, accurate a priori wirelength estimator. In *Proc. of ICCAD*, volume 2005, pages 173–180, 2005.

[9] O. Nelles. *Nonlinear system identification.* Springer, Berlin, Germany, 2001.

[10] G. Nam, C. Alpert, P. Villarrubia, B. Winter, and M. Yildiz. The ISPD2005 placement contest and benchmark suite. In *Proc. of ISPD*, pages 216–219, 2005.

[11] J. Li, L. Behjat, and A. Kennings. Net Cluster: A net-reduction-based clustering preprocessing algorithm for partitioning and placement. *IEEE Trans. on CAD*, 26 (4):669–679, 2007.

[12] M. Feuer. Connectivity of random logic. *IEEE Trans. on Computers*, C-31(1):29–33, 1982.

[13] J. Cotter and P. Christie. The analytical form of the length distribution function for computer interconnection. *IEEE Trans. on CAS*, 38(3):317–320, 1991.

[14] A. Caldwell, A. Kahng, St. Mantik, I. Markov, and A. Zelikovsky. On wirelength estimations for row-based placement. *IEEE Trans. on CAD*, 18(9):1265–1278, 1999.

[15] M. Pedram and B. Preas. Interconnection length estimation for optimized standard cell layouts. In *Proc. of ICCAD*, pages 390–393, 1989.

[16] T. Hamada, C. Cheng, and P. Chau. A wire length estimation technique utilizing neighborhood density equations. *IEEE Trans. on CAD*, 15(8):912–922, 1996.

[17] H. Heineken and W. Maly. Standard cell interconnect length prediction from structural circuit attributes. In *Proc. of IEEE CICC*, pages 167–170, 1996.

[18] B. Hu and M. Marek-Sadowska. Wire length prediction based clustering and its application in placement. In *Proc. of DAC*, pages 800–805, 2003.

[19] Q. Liu, B Hu, and M. Marek-Sadowska. Wire length prediction in constraint driven placement. In *Proc. of SLIP*, pages 99–105, 2003.

[20] J. Dambre, D. Stroobandt, and J. Van Campenhout. Toward the accurate prediction of placement wire length distributions in vlsi circuits. *IEEE Trans. on VLSI*, 12(4): 339–348, 2004.

[21] T. Yan and H. Murata. Fast wire length estimation by net bundling for block placement. In *Proc. of ICCAD*, pages 172–178, 2006.

[22] T. Yan, S. Li, Y. Takashima, and H. Murata. A theoretical study on wire length estimation algorithms for placement with opaque blocks. In *Proc. of ASP-DAC*, pages 268–273, 2007.

[23] T. Taghavi, F. Dabiri, A. Nahapetian, and M. Sarrafzadeh. Tutorial on congestion prediction. In *Proc. of SLIP*, pages 15–24, 2007.

[24] A. Farshidi, L. Behjat, L. Rakai, and B. Fathi. A pre-placement individual net length estimation model and an application for modern circuits. *Integration*, 44(2): 111–122, 2011.

[25] Q. Liu and M. Marek-Sadowska. Pre-layout wire length and congestion estimation. In *Proc. of DAC*, pages 582–587, 2004.

[26] L. Rakai, J. Li, L. Behjat, and J. Huang. A structural study and hyperedge clustering technique for large scale circuits. In *Proc. of IEEE Workshop on Signal Processing Systems Design and Implementation*, pages 66–70, 2006.

[27] M. Buhmann. *Radial basis functions : theory and implementations*. Cambridge University Press, New York, USA, 2003.

[28] S. Adya, S. Chaturvedi, J. Roy, D. Papa, and I. Markov. Unification of partitioning, floorplanning and placement. In *Proc. of ICCAD*, pages 550–557, 2004.

[29] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996.

[30] J. Rodgers and W. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[31] D. Ranaweera, N. Hubele, and A. Papalexopoulos. Application of radial basis function neural network model for short-term load forecasting. *IEE Proc. of GTD*, 142: 45–50, 1995.

[32] Z. Yun, Z. Quan, S. Caixin, L. Shaolan, L. Yuming, and S. Yang. Rbf neural network and anfis-based short-term load forecasting approach in real-time price environment. *IEEE Trans. on Power Systems*, 23(3):853–858, aug. 2008.

[33] S. Seshagiri and H. Khalil. Output feedback control of nonlinear systems using rbf neural networks. *IEEE Trans. on Neural Network*, 11(1):69–79, Jan 2000.

[34] S. Chen and S. Billings. Neural networks for nonlinear dynamic system modelling and identification. *International Journal of Control*, 56(2):319–346, 1992.

[35] M. Pottmann and D. Seborg. A nonlinear predictive control strategy based on radial basis function models. *Computer and Chemical Engineering*, 21(9):965–980, 1997.

[36] M. Er, S. Wu, J. Lu, and H. Toh. Face recognition with radial basis function (rbf) neural networks. *IEEE Trans. on Neural Networks*, 13(3):697–710, may 2002.

[37] X. Wang, J. Li, and Y. Niu. Face recognition with radial basis function (rbf) neural networks. In J. Wang, X. Liao, and J. Wang, editors, *Advances in Neural Networks ISNN 2005*, volume 3497 of *Lecture Notes in Computer Science*, pages 171–176. Springer Berlin / Heidelberg, 2005.

[38] C. Lindquist. Radial basis function transforms and their use in signal processing. In *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 315–319, 1-3 1993.

[39] J. Carr, W. Fright, and R. Beatson. Surface interpolation with radial basis functions for medical imaging. *IEEE Trans. on Medical Imaging*, 16(1):96–107, feb. 1997.

[40] T. Ishikawa, Y. Tsukui, and M. Matsunami. A combined method for the global optimization using radial basis function and deterministic approach. *IEEE Trans. on Magnetics*, 35(3):1730–1733, 1999.

[41] N. Sundararajan, P. Saratchandran, and Y. Lu. *Radial basis function neural networks with sequential learning : MRAN and its applications*. World Scientific, Singapore, 1999.

[42] International Technology Roadmap for Semiconducotors. ITRS 2009 report. Technical report, http://www.itrs.net/Links/2009ITRS/Home2009.htm, 2009.

[43] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. *IEEE Trans. on VLSI*, 7(1):69–79, 1999.

[44] G. Nam. ISPD 2006 placement contest: Benchmark suite and results. In *Proc. of ISPD*, page 167, 2006.

[45] A. Caldwell, A. Kahng, and I. Markov. Improved algorithms for hypergraph bipartitioning. In *Proc. of ASP-DAC*, pages 661–666, 2000.

[46] C. Alpert, D. Huang, and A. Kahng. Multilevel circuit partitioning. *IEEE Trans. on CAD*, 17(8):655–667, 1998.

[47] C. Alpert, A. Kahng, G. Nam, S. Reda, and P. Villarrubia. A semi-persistent clustering technique for VLSI circuit placement. In *Proc. of ISPD*, pages 200–207, 2005.

[48] J. Cong and S. Lim. Edge separability-based circuit clustering with application to multilevel circuit partitioning. *IEEE Trans. on CAD*, 23(3):346–357, 2004.

[49] B. Hu and M. Marek-Sadowska. Fine granularity clustering for large scale placement problems. In *Proc. of ISPD*, pages 67–74, 2003.

[50] T. Chan, J. Cong, M. Romesis, J. Shinnerl, K. Sze, and M. Xie. mPL6: Enhanced multilevel mixed-size placement. In *Proc. of ISPD*, pages 212–214, 2006.

[51] A. Kahng, S. Reda, and Q. Wang. APlace: A general analytical placement framework. In *Proc. of ISPD*, pages 233–235, 2005.

[52] G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. In *Proc. of DAC*, pages 343–348, 1999.

[53] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. In *Proc. of DAC*, pages 526–529, 1997.

[54] J. Cong and S. Lim. Edge separability based circuit clustering with application to circuit partitioning. In *Proc. of ASP-DAC*, pages 429–434, 2000.

[55] C. Fiduccia and R. Mattheyses. A linear time heuristic for improving network partitions. In *Proc. of DAC*, pages 175–181, 1982.

[56] G. Nam, S. Reda, C. Alpert, P. Villarrubia, and A. Kahng. A fast hierarchical quadratic placement algorithm. *IEEE Trans. on CAD*, 25(4):678–691, 2006.

[57] J. Li and L. Behjat. Net Cluster: a net-reduction based clustering preprocessing algorithm. In *Proc. of ISPD*, volume 26, pages 200–205, 2006.

[58] J. Yan, C. Chu, and W. Mak. Safechoice: a novel clustering algorithm for wirelength-driven placement. In *Proc. of ISPD*, pages 185–192, 2010.

[59] J. Roy and I. Markov. *Partitioning-driven Techniques for VLSI Placement*. Handbook of Algorithms for VLSI Physical Design Automation. CRC Press, 2008.

[60] N. Viswanathan, M. Pan, and C. Chu. Fastplace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control. In *Proc. of ASP-DAC*, pages 135–140, 2007.

[61] J. Cho. Wiring space and length estimation in two-dimensional arrays. *IEEE Trans. on CAD*, pages 612–615, 2000.

[62] J. Li, L. Behjat, and J. Huang. An effective clustering algorithm for mixed-size placement. In *Proc. of ISPD*, pages 111–118. ACM, New York, NY, USA, 2007.

[63] C. Chang, J. Cong, and M. Xie. Optimality and scalability study of existing placement algorithms. In *Proc. of ASP-DAC*, pages 621–627. ACM, New York, NY, USA, 2003.

[64] A. Caldwell, A. Kahng, and I. Markov. Design and implementation of move-based heuristics for vlsi hypergraph partitioning. In *ACM JEA*, volume 5, 2000.

[65] A. Chatterjee and R. Hartley. A new simultaneous circuit partitioning and chip placement approach based on simulated annealing. In *Proc. of DAC*, pages 36–39, 1990.

[66] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(2):291–307, 1970.

[67] C. Cheng. Risa: Accurate and efficient placement routability modeling. In *Proc. of ICCAD*, pages 690–695, 1994.

[68] A. Kahng and Q. Wang. Implementation and extensibility of an analytic placer. In *Proc. of ISPD*, pages 18–25, 2004.

[69] A. Agnihotri, S. Ono, and P. Madden. Recursive bisection placement: Feng shui 5.0 implementation details. In *Proc. of ISPD*, pages 230–232, 2005.

[70] C. Chang, J. Cong, Z. Pan, and X. Yuan. Multilevel global placement with congestion control. *IEEE Trans. on CAD*, 22(4):395–409, 2004.

[71] J. Roy, D. Papa, S. Adya, H. Chan, A. Ng, F. Lu, and I. Markov. Capo: Robust and scalable open-source min-cut floorplacer. In *Proc. of ISPD*, pages 224–226, 2005.

[72] S. Areibi, M. Xie, and A. Vannelli. An efficient steiner tree algorithm for vlsi global routing. In *Proc. of CCECE*, volume 2, pages 1067–1072, 2001.

[73] L. Behjat, D. Kucar, and A. Vannelli. A novel eigenvector technique fot large scale combinatorial problems in VLSI layout. *Journal of Combinatorial Optimization*, 6 (3):271–286, 2002.

[74] R. Hadsell and P. Madden. Improved global routing through congestion estimation. In *Proc. of DAC*, pages 28–31, 2003.

Appendix A

Physical Design

The flowchart of the physical design procedure is shown in Figure A.1. The input of this procedure is a circuit schematic. Circuit schematics are usually represented in netlist format. A netlist includes a list of nets, interconnections between circuit components, and a list of circuit components, i.e. cells. As seen in Figure A.1, the physical design process outputs the circuit layout. All of the physical properties of the circuit are determined in the layout [5].

The procedure of physical design contains three main stages that are referred to as partitioning, placement and routing. These stages are shown in Figure A.1. Each stage is further described in the following sections.

A.1 Partitioning

The objective of partitioning is to divide a circuit represented by a netlist, into several partitions that are relatively independent, i.e. as few wires as possible connect them. The partitions are also desired to be roughly from the same sizes. In today's ICs, there exist millions of transistors that should be integrated [10, 44]. Sometimes, the number of transistors may be very large such that it is needed to partition the circuit into smaller sub-circuits. These sub-ciruits should be as independent as possible because the wires that connect two different sub-circuits are much more costly than those that are confined to a single sub-circuit. The procedure of dividing the circuit into smaller sub-circuits is called partitioning [4, 5].

In [53, 55, 64–66], several examples of existing partitioning algorithms are presented.

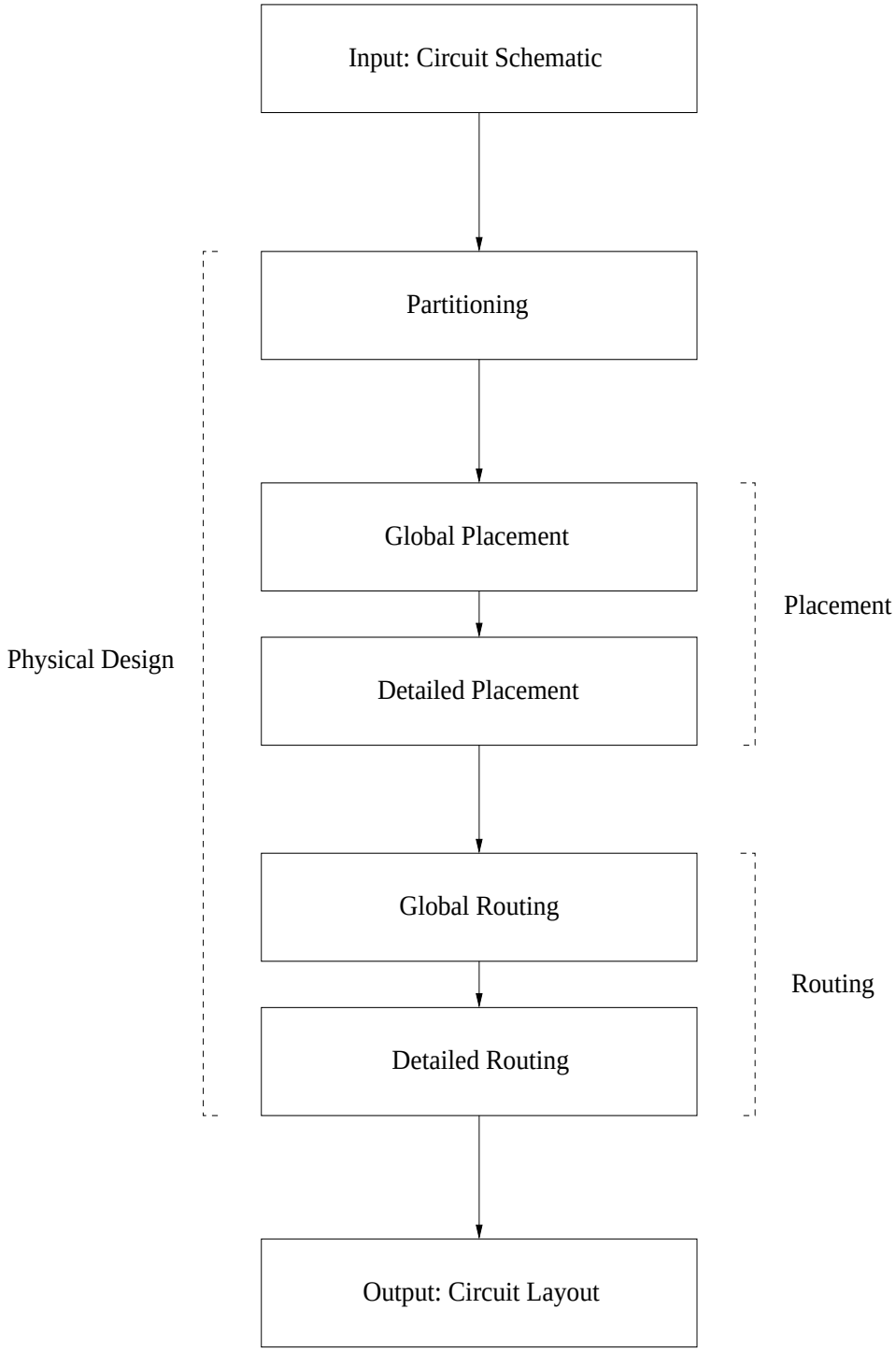


Figure A.1: Physical design procedure

The partitioning algorithms are also used in solving other physical design problems [53].

A.2 Placement

In the second main stage of physical design, the exact locations of the circuit components, such as logical gates and terminals, are determined [4,5]. This stage is referred to as the placement stage. A constraint on the placement stage is that the locations of the cells must be unique so that two distinct circuit components cannot be placed on the same location. The general objective of the placement stage is to minimize the total length of nets that connect the circuit components. Minimization of the total wire length can result in reduction of delays and power consumption of the circuit [60].

There are some other objectives that are also considered by a placement algorithm. One is that the algorithm ensures that the circuit is routable. This means that the algorithm does not place the cells such that highly congested areas in which the routing of nets is not possible, are formed [67].

As shown in Figure A.1, placement algorithms usually include two main steps: global placement and detailed placement [5]. In the first step, global placement is performed in which a rough placement solution is produced. This solution is not a finalized placement solution, since it may contain overlaps between cells. In the second step, detailed placement is performed on this solution. In detailed placement, legalization techniques are used to remove the overlaps between cells. Also, local refinement techniques are applied to improve the quality of the placement results. Detailed placement outputs a legal placement solution which does not include any overlaps between cells.

The existing placement algorithms are typically classified into two groups: analytical placement algorithms and partitioning-based placement algorithms. Placement is treated as an optimization problem in analytical algorithms. In [50, 60, 68], some examples of

these algorithms are presented. On the other hand, in partitioning-based algorithms, the circuit is first partitioned. Then, each partition is separately placed on the layout area. Examples of partitioning-based algorithms are proposed in [69–71].

A.3 Routing

In the third main stage of physical design, routing is performed where the paths for all of the nets of the circuit are determined [4, 5]. The objective of routing is to route all the wires in the circuit while minimizing the total wire length. If an excessive number of nets must pass through a certain area, the circuit is not routable. Therefore, in the routing stage, it is essential to find a routable solution.

As shown in Figure A.1, the routing is usually performed in two main steps: global routing and detailed routing. In global routing, an approximate routing solution is produced. In this solution, approximate paths of the nets are determined. Then, the detailed routing is performed in the second step. In this step, the routing solution is further refined and the exact paths of the nets are determined. Several examples of existing routing algorithms are presented in [72–74].

A.4 Other Approaches to Improve Physical Design

In order to improve the quality of the physical design and reduce the runtime, several approaches are used. One approach is clustering that is used to handle the large sizes of today's integrated circuits [11]. Clustering is used before partitioning and placement to reduce the sizes of the circuits. This decrease in size results in reduced runtime and improvements in the quality of partitioning and placement [11, 43]. A comprehensive background on clustering is presented in Chapter 3.

Another approach that can be used to improve the quality of physical design is wire

length estimation, which is the main focus of this thesis. The length of wires used for routing the nets of circuits, is usually used to measure the quality of different stages of physical design. Therefore, wire length estimation becomes important. Wire length estimation techniques are usually used to estimate the wire length before, during and after the placement stage to measure the quality of placement and routing stages [6]. In Chapter 2, a background on the existing wire length estimation techniques is presented.