Degree Project in Nanotechnology

Second cycle, 30 credits

Interconnect length estimation for advanced CMOS nodes

AHMED AFTAB DAR BASHIR

Interconnect length estimation for advanced CMOS nodes

AHMED AFTAB DAR BASHIR

Master's Programme, Nanotechnology, 120 credits
Date: April 15, 2025

Supervisors: Francesco Dell Atti, Ji-Yung Lin
Examiner: Per-Erik Hellström
School of Electrical Engineering and Computer Science
Host organization: imec
Swedish title: Sammankopplingslängduppskattning för avancerade CMOS-noder

Abstract

Accurate wire length estimation is critical for early-stage physical design in advanced CMOS nodes. Traditional wire length estimation models, such as those proposed by Donath and Davis, have been widely used to predict interconnect distributions. However, their assumptions—such as uniform placement, rigid grid structures, and simplified interconnect probability functions—do not fully capture the complexities of modern physical designs. This thesis aims to refine these models by incorporating more realistic placement constraints and empirical validation.

The first part of the study focuses on extracting Rent's coefficients using two different methods: the Growing Box Method and the Recursive Splitting Method. Both methods yield nearly identical Rent exponents, differing only at the second decimal place, but the Recursive Splitting Method offers significantly reduced computation time.

Next, the traditional wire length estimation models of Donath and Davis are evaluated. Donath's model shows a significant deviation from actual distributions, likely due to its reliance on hierarchical placement, whereas the project focuses on PnR based placed designs. Davis' model provides better accuracy for longer interconnects but fails to capture short-range interconnect behavior, leading to overestimations by orders of magnitude.

To address these limitations, this thesis proposes a hybrid wire length estimation model. The Davis model is optimized by adjusting its power-law exponent to better fit empirical data for longer interconnects. However, short interconnect behavior is modeled separately using Monte Carlo simulations. This approach accounts for non-uniform cell sizes, pin locations, and realistic placement constraints. The final combined model achieves a strong fit to real design data, with an R^2 value of 0.984.

The study also identifies key challenges, including the sensitivity of wire length estimation to binning choices in PDFs. To mitigate this, CDFs are used instead, improving stability and accuracy. Additionally, the transition point between short and long interconnect models remains an area for potential refinement.

Future work could focus on extending the model to purely *a priori* wire length estimation by starting from a circuit graph and integrating partitioning and placement techniques rather than relying on post-layout data. Further refinements could also involve improving transition continuity between short and long interconnect models, and generalizing the approach to heterogeneous integration scenarios.

Sammanfattning

Noggrann uppskattning av ledningslängder är avgörande för fysisk design i ett tidigt skede vid avancerade CMOS-noder. Traditionella modeller för ledningslängdsuppskattning, såsom de som föreslagits av Donath och Davis, har länge använts för att förutsäga fördelningen av förbindelser. Deras antaganden – såsom jämn placering, styva rutnätsstrukturer och förenklade sannolikhetsfunktioner för förbindelser – fångar dock inte fullt ut komplexiteten i moderna fysiska designer. Denna avhandling syftar till att förfina dessa modeller genom att införliva mer realistiska placeringsbegränsningar och empirisk validering.

Den första delen av studien fokuserar på att extrahera Rents koefficienter med hjälp av två olika metoder: den växande boxmetoden och den rekursiva delningsmetoden. Båda metoderna ger nästan identiska Rent-exponenter, som endast skiljer sig på andra decimalen, men den rekursiva delningsmetoden erbjuder avsevärt minskad beräkningstid.

Därefter utvärderas de traditionella modellerna för ledningslängdsuppskattning av Donath och Davis. Donaths modell visar en tydlig avvikelse från de faktiska fördelningarna, troligen på grund av dess beroende av hierarkisk placering, medan detta projekt fokuserar på PnR-baserade placerade designer. Davis modell ger bättre noggrannhet för längre förbindelser men misslyckas med att fånga beteendet hos korta förbindelser, vilket leder till överskattningar med flera tiopotenser.

För att hantera dessa begränsningar föreslår denna avhandling en hybridmodell för ledningslängdsuppskattning. Davis modell optimeras genom att justera dess potenslagsexponent för att bättre passa empiriska data för längre förbindelser. Kortare förbindelser modelleras dock separat med hjälp av Monte Carlo-simuleringar. Denna metod tar hänsyn till icke-uniforma celldimensioner, kontaktpunkternas positioner och realistiska placeringsbegränsningar. Den slutliga kombinerade modellen uppnår god överensstämmelse med verkliga designdata, med ett R^2-värde på 0.984.

Studien identifierar också viktiga utmaningar, inklusive känsligheten i ledningslängdsuppskattning med avseende på binningval i sannolikhetsfördelningsfunktioner (PDF). För att mildra detta används i stället kumulativa fördelningsfunktioner (CDF), vilket förbättrar stabilitet och noggrannhet. Övergångspunkten mellan modellerna för korta och långa förbindelser återstår dessutom som ett område för potentiell förbättring.

Framtida arbete kan fokusera på att utöka modellen till en ren *a priori* uppskattning av ledningslängd genom att utgå från ett kretsschema och

integrera partitionerings- och placeringsmetoder snarare än att förlita sig på layoutdata. Vidare förbättringar kan också involvera förbättrad kontinuitet i övergången mellan korta och långa förbindelsemodeller, samt generalisering av metoden till heterogena integrationsscenarier.

Acknowledgments

Firstly, I thank Allah for granting me the strength, motivation, and patience to pursue knowledge and research, and for all the countless blessings in my life.

I would like to express my deepest gratitude to my parents for their continuous and relentless support throughout this journey. Their unwavering belief in me has been a constant source of encouragement.

I am sincerely thankful to my supervisors, Francesco Dell'Atti and Ji-Yung Lin, for their invaluable guidance, support, and for helping me find direction in my research. Their insights and encouragement have been instrumental to this work. My heartfelt thanks also go to my examiner, Per-Erik Hellström, for his constructive feedback and support, which have helped improve the quality of this thesis.

I would also like to extend my gratitude to all my friends and family for their love, support, and constant encouragement.

Stockholm, April 2025
Ahmed Aftab Dar Bashir

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Setting

This research was conducted at imec, a Belgium-based research and innovation center specializing in nanoelectronics and digital technology. As part of the Physical Design Research Team, this project was carried out in close collaboration with ARM. Consequently, the designs analyzed in this study are based on ARM architectures.

Accurate wire length distribution models are essential for early-stage physical design estimation, enabling efficient placement, routing, and performance prediction in modern VLSI circuits. Traditional models often assume rigid grid structures with uniform cell sizes. However, with increasing design complexity, variations in cell size, placement density, and pin locations significantly impact interconnect distributions.

The primary objective of this project is to develop a theoretical wire length distribution model that accurately characterizes interconnect behavior in advanced CMOS node-based ARM designs. This model aims to bridge the gap between traditional Rent's rule-based estimations and modern physical design challenges by incorporating non-uniform cell sizes, variable pin locations, and realistic placement constraints.

Given the growing complexity of physical design in advanced technology nodes, the project examines the following question:

- How can a theoretical wire length distribution model be formulated to accurately predict interconnect requirements while maintaining computational efficiency and practical applicability?

1.2 Motivation

As semiconductor technology advances towards smaller feature sizes, the complexity and cost of fabrication continue to rise significantly. With the transition to advanced CMOS nodes, techniques such as Extreme Ultraviolet Lithography (EUV) are required to achieve the necessary resolution, adding substantial manufacturing costs. The Back-End-of-Line (BEOL) process, responsible for interconnect fabrication, is particularly affected by these cost constraints. Therefore, a precise wire length distribution model is essential to optimize BEOL resources efficiently, reducing waste and improving cost-effectiveness.

Furthermore, to obtain accurate interconnect data, conventional methodologies require full Place-and-Route (PnR) execution, which is computationally expensive and time-consuming. This motivates the need for a theoretical model that can predict wire length distributions early in the design cycle, starting directly from the netlist.

Such a model would enable early-stage routing requirement estimation and guide BEOL optimization in terms of metal pitch, the number of metal layers, and overall routing resources. By integrating this approach, it becomes possible to make informed design decisions that improve manufacturability, enhance performance, and reduce the overall design-to-fabrication turnaround time. The development of an accurate wire length distribution model for advanced ARM-based CMOS designs is, therefore, a crucial step toward efficient and cost-effective semiconductor design and manufacturing.

1.3 Related Literature

Wire length estimation methods can be broadly categorized into three approaches [1]:

1. **Empirical Methods:** These methods derive equations for circuit properties through data extraction and curve fitting. They rely on statistical analysis of existing designs to establish predictive models. While they offer reasonable accuracy, they often lack generalization across different technology nodes.

2. **Procedural Methods:** These methods refine wire length estimates by analyzing detailed aspects of the design process, including placement, routing, and interconnection structures. They provide improved accuracy by considering physical design constraints and design flow specifics but

are computationally expensive and heavily dependent on the specific design methodology used.

3. **Theoretical Methods:** These methods generate closed-form equations for wire length distribution based on simplified assumptions about interconnection structures. Among these, Rent's Rule remains one of the most widely used frameworks.

Among the theoretical methods, Rent's Rule stands out as a robust framework for wire length estimation. Originally proposed by Landman and Russo in the 1960s [2], Rent's Rule establishes a relationship between the number of terminals and blocks in a partitioned circuit. This rule has been extensively utilized in hierarchical wire length estimation, forming the basis for numerous subsequent models. In the late 1970s, Donath extended Rent's Rule to estimate wire lengths more systematically, leading to significant advancements in theoretical estimation techniques [3, 4].

Building upon Donath's approach, Davis introduced a stochastic wire length model in the late 1990s [5]. This model incorporated probabilistic principles to overcome the limitations of deterministic methods, enabling a more accurate representation of wire length distributions in VLSI designs. Around the same time, Stoorbandt et al. independently developed a probabilistic method lusing generating polynomials to represent distributions efficiently [6].

In the late 2010s, Prasad et al.[7] further refined the Davis Model by incorporating the effects of variable fanout on wire length distribution, improving the accuracy of interconnect predictions in modern CMOS designs. These advancements demonstrate the evolution of wire length estimation models from deterministic approximations to probabilistic and stochastic approaches that better capture real-world design variability.

1.4 Designs Under Consideration

In this study, four ARM designs were analyzed to develop the wire length distribution model. These include two 64-bit ARM core designs and two M0 ARM designs. The two 64-bit ARM designs feature the same cell technology (N3, Nanosheet) and contain approximately 700,000 cells, with the primary distinction being that the second design operates at a higher frequency. The two M0 ARM designs have a significantly smaller cell count of about 16,000 and differ in terms of cell technology: one utilizes N2, Nanosheet technology,

while the other employs A7, CFET technology. The specifications of these designs are summarized in Table 1.1.

Table 1.1: Design Specifications

Design	64-bit ARM #1	64-bit ARM #2	M0 ARM #1	M0 ARM #2
No. of cells	∼700k	∼700k	∼16k	∼16k
Cell Technology	N3, Nanosheet	N3, Nanosheet	N2, Nanosheet	A7, CFET

The three key cell technologies used in this study—N3, N2, and A7—are described below:

- **N3 (imec's 3nm technology):** This technology employs nanosheet (NSH) transistors with a 5-track (5T) standard cell height, a gate pitch of 42 nm, and a metal pitch of 18 nm. It also features a back-end-of-line (BEOL) metal stack with 17 full-stack (FS) layers and utilizes buried power rails (BPR) for scaling improvements[8].
- **N2 (imec's 2nm technology):** This is an evolution of N3, using nanosheet transistors but with a 6-track (6T) standard cell height and a larger gate pitch of 48 nm. The metal pitch remains at 18 nm, and the BEOL stack is similar to N3 with 17 FS layers[9].
- **A7 (imec's 7Å technology):** The A7 technology introduces complementary field-effect transistors (CFET), allowing further device stacking and power efficiency improvements. It features a 3.5-track (3.5T) standard cell height, a gate pitch of 42 nm, and a slightly relaxed metal pitch of 19 nm. The BEOL stack consists of 13 FS layers and 5 bottom-stack (BS) layers, along with backside power delivery network (BS-PDN) for enhanced power efficiency[10].

Table 1.2 summarizes the key attributes of these technology nodes.

Table 1.2: Cell Technology Specifications

Technology Node	N3	N2	A7
Transistor Structure	NSH	NSH	NSH CFET
Std Cell Height [#tracks]	5T	6T	3.5T
Gate Pitch [nm]	42	48	42
Metal Pitch (MP) [nm]	18	18	19
Scaling Boosters	BPR	-	BS-PDN
BEOL Metal Stack	17 (FS)	17 (FS)	13 (FS) + 5 (BS)

1.5 Delimitations

This study is concerned with developing a theoretical wire length distribution model for advanced CMOS node-based ARM designs. However, certain aspects of wire length modeling and physical design are explicitly beyond the scope of this work.

A complete theoretical wire length distribution model based on Rent's rule generally consists of two major components:

1. **Placement and routing of modules** from the circuit graph onto a Manhattan grid.
2. **Extraction of Rent's coefficients** from the placed circuit and implementing the proposed wire length distribution model.

While the overarching motivation remains the estimation of routing requirements and facilitating early-stage design decisions, both of these components represent extensive research questions on their own. Given the complexity and scale of such an endeavor, this thesis is explicitly limited to the second part—extracting Rent's parameters from post-placement circuits and refining existing wire length models.

Instead of formulating an *a priori* model that predicts wire length distributions before placement and routing, this thesis takes a *posteriori* approach. This means that PnR-run ARM designs are analyzed to optimize current wire length estimation methods and propose improvements tailored for advanced nodes. Consequently, this study does not attempt to:

- Developing or implementing placement and routing techniques from circuit graph.
- Predict wire length distributions before placement using a purely theoretical model.
- Address specific process or manufacturing variations affecting interconnect behavior.
- Extend the model beyond ARM-based designs or to older technology nodes.

1.6 Thesis Outline

The outline of the thesis is as follows: Chapter 2 provides a comprehensive background on the digital design cycle and presents the theoretical model for

digital design. It outlines the stages and processes involved in digital design, establishing the framework for understanding the methodologies discussed in later chapters. Chapter 3 presents the theoretical foundation for Rent's rule and reviews existing wirelength estimation techniques. Chapter 4 introduces the methodology for the extraction of rent coefficients, including detailed procedures and results. Chapter 5 presents the methodology and results for the wire length distribution estimation models. Chapter 6 provides a discussion of the findings. Finally, Chapter 7 concludes the thesis by summarizing the key insights and outlining potential directions for future research, emphasizing the significance of the results and their broader impact on digital design.

Chapter 2

Digital Design Cycle and Models

This chapter presents a basic background on digital design, situating the research presented in this thesis. Using the Y-chart proposed by Gajski and Kuhn, we explore the stages of the design cycle and clarify the specific position of the thesis work in that process. Furthermore, this chapter describes how the main goal of this thesis is the development of more accurate wire length estimation methods. Moreover, a background about the theoretical models of digital design is provided.

2.1 Digital Design Cycle

Realizing a digital design involves multiple steps (Figure 2.1), starting from system specification and progressing through functional, logic, circuit, and physical design, ultimately leading to chip fabrication and testing.

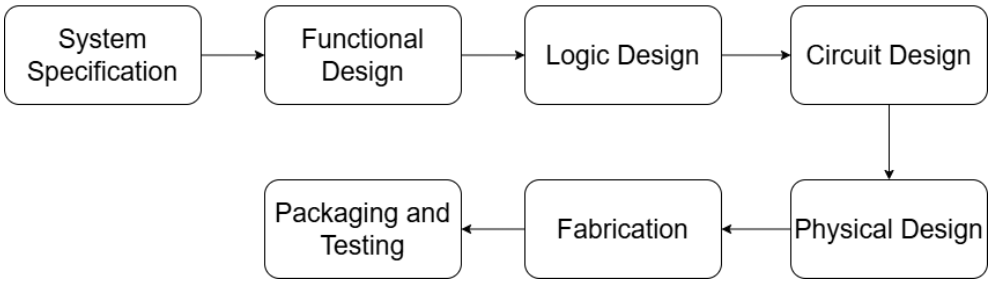


Figure 2.1: A basic digital design cycle [11]

In order to view these steps in broader sense, we illustrate it using a design

space, where a "point" in the space represents the system to be designed and the different design views can be illustrated as the "projections" along one of the directions. These dimensions and views can be represented using Gajski and Kuhn's Y-chart [12], as shown in Figure 2.2.

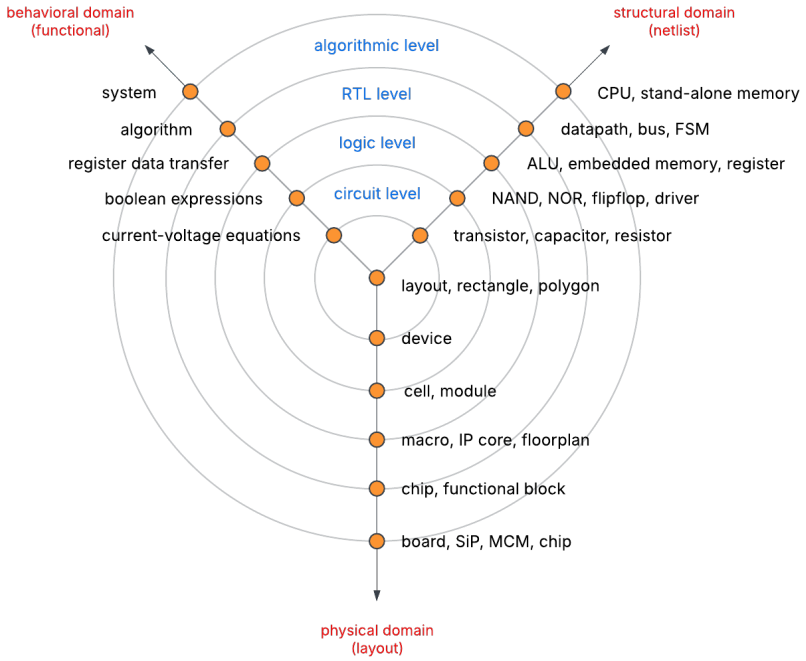


Figure 2.2: Gajski-Kuhn VLSI design abstraction-level chart [13]

Digital systems can be described from three primary perspectives, each offering a unique understanding of their design and functionality. The first perspective is the **behavioral domain**, which focuses on the system's functional and temporal behavior. This involves examining the relationship between the system's inputs and outputs over time. In addition to functional and timing aspects, the behavioral perspective also includes considerations such as power dissipation and the system's resilience to errors, both of which are critical for modern digital systems.

The **logical structural domain**, being the second perspective, describes how a design implements the desired behavior, focusing on the arrangement of basic components. Unlike the behavioral perspective, which defines what the system does, this view details its organization using block diagrams, circuit outlines, and logical schematics.

The third and final perspective is the **physical structural domain**, which

deals with the system's physical realization in a specific technology. This perspective focuses on geometric properties, including length, diameter, area, and volume, which only become apparent in this dimension. The geometric depiction of the digital design is called the layout and the method to determine this representation is known as the layout process. Together, these three perspectives provide a comprehensive framework for understanding and designing digital systems.

2.1.1 Design Trajectory

Returning to the original description of the design cycle in Figure 2.1, it can be expanded to realize how each step traverses the Y-chart as follows:

1. **System Specification:** The system specification, being a high-level and frequently informal representation of the system, is where the design process starts. Important elements including performance, functionality, size, speed, and power requirements are described in this specification. However, it is typically incomplete and imprecise, utilizing language, diagrams, and plots to formulate the problem. Due to its informal nature, the system specification is unsuitable for automated digital design tools, necessitating a formal description of system behavior. This formalization marks the first stage of Gajski's Y-chart and is addressed in the functional design phase.

2. **Functional Design:** In this phase, the system's behavior is formally described, providing the foundation for subsequent design steps. This involves defining the relationship between inputs and outputs, with an emphasis on functional and temporal behavior. The formal description ensures compatibility with automated tools, allowing for simulation and verification of the system's intended operation.

3. **Logic Design:** The arithmetic and logic operations required to implement the functional behavior are determined and tested during the logic design process. Usually, hardware description languages like VHDL or Verilog are used to describe these operations. The logic is represented using boolean expressions, which are optimized to produce a simple design while maintaining the required functionality. This process, which is frequently called refining, guarantees that the design is precise and efficient.

4. **Design Synthesis:** This phase transforms the behavioral description into a structural representation. Often called "technology mapping",this step maps functions to components from a predefined library associated with the chosen

technology. At this stage, the design remains abstract, without geometric details, focusing instead on logical interconnections.

5. **Physical Design:** The logical framework is translated into a tangible representation in a particular technology during the physical design process. Interconnections become actualized as physical wires, and the system takes on its final shape, such as a chip, board, or cabinet. Design guidelines based on material qualities and limitations of the production process dictate the layout.

6. **Fabrication and Testing:** Once the design is complete, it undergoes fabrication and testing. Fabrication transfers the layout to a wafer, while testing ensures the system functions correctly. Although these steps are not part of the design trajectory, they are crucial for producing a functional end product.

2.1.2 Applicability of this Thesis in the Design Cycle

This thesis's work falls under the design cycle's physical design phase, specifically addressing the layout generation step, which determines a circuit's physical structure based on its logical representation. Wire-length estimation plays a critical role in this process, as it directly impacts the quality and feasibility of the layout. Accurate wire-length predictions ensure that circuits meet system specifications while adhering to the constraints of the physical architecture, including minimal component pitches and wire widths defined by advanced CMOS nodes.

2.2 Theoretical Models of Digital Design

This thesis centers on *a posteriori* wire length estimation methods. Such estimations are applied when the placement of components is predetermined, aiming to predict the wire length that will result after routing. This technique proves particularly useful in scenarios where routing is considerably more computationally expensive than either placement or wire length prediction. The estimations are grounded in three theoretical frameworks: *the circuit model*, *the physical architecture model*, and *the layout generation model*.

This chapter provides a background about these three models with some definitions and illustrations relevant to these topics.

2.2.1 The Circuit Model

A circuit can be defined as a design that is viewed through the logical structural perspective. On the Gajski-Kuhn's Y-chart it lies on the structural domain branch (Figure 2.2). A circuit can be represented as a set of interconnected *blocks*, which can signify transistors, gates, or entire subcircuits. These interconnections are referred to as *nets*, and nets connected to more than two blocks are termed *multi-terminal nets*. Some nets also connect to the external environment of the circuit, known as *external nets*, as opposed to *internal nets*, which connect only to blocks within the circuit. To effectively model these external nets, a special type of block, called a *pin*, is introduced. A pin represents the external terminal for a net, and every external net is connected to exactly one pin. Consequently, the number of pins in a circuit corresponds to the number of external nets, while the other internal blocks are referred to as logic blocks. A basic circuit model is shown in Figure 2.3.

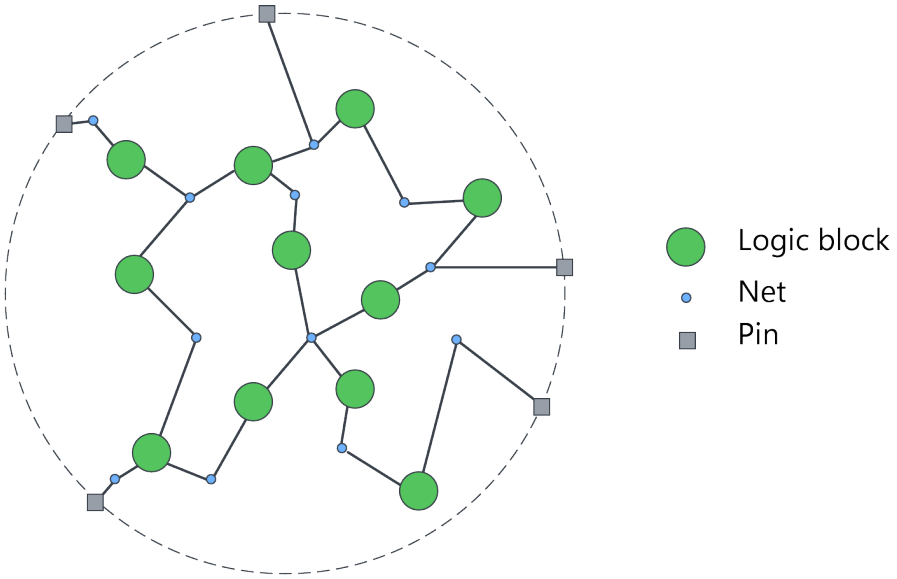


Figure 2.3: A basic circuit model [14]

2.2.2 The Physical Architectural Model

The significance of a circuit's interconnection length emerges only once the circuit has been mapped onto a physical architecture. According to the structural levels outlined in the Y-chart, this physical architecture might correspond to a substrate used for placing logic cells or a computer board

on which chips are mounted. Across all abstraction levels, the architecture generally exhibits a regular structure, enabling the definition of unit distances and imposing constraints on where components can be placed.

In the physical architecture model, logic blocks correspond to *cells*, which are designated positions for placement, while *pads* are positions for pins. Each cell or pad is assumed to accommodate only one logic block or pin, respectively. Nets are represented by *channels* and connection points within this framework, creating a structured mapping of interconnections.

One widely adopted representation of the physical architecture is the Manhattan grid, illustrated in Figure 2.4. In this model, adjacent cells are positioned on a uniform lattice, with interconnecting channels linking them. Distances within the grid are computed using the Manhattan metric, where the distance d between two points located at (X_1, Y_1) and (X_2, Y_2) is given by:

$$d = |X_2 - X_1| + |Y_2 - Y_1| \tag{2.1}$$

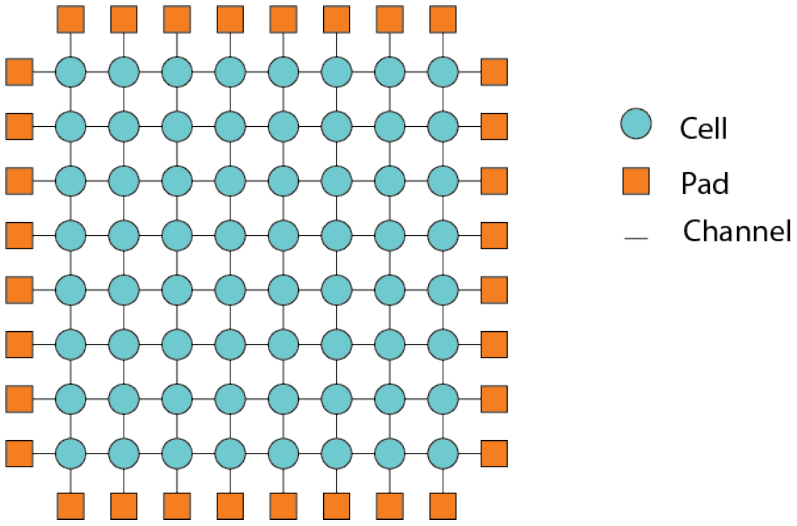


Figure 2.4: The Manhattan Grid

2.2.3 The Layout Generation Model

In its abstract form, a circuit consists of interconnected logic blocks, with properties such as wire length remaining undefined until the physical implementation stage. As described by Gajski and Kuhn in their Y-chart (Figure 2.2), this transition happens during the layout generation phase, where the logical design is translated into a physical layout, making it possible to

assess wire lengths and interconnection efficiency. This section introduces the models for *placement* and *routing* within this context.

2.2.3.1 The Placement Model

Circuit placement involves assigning logic blocks and pins to designated cells and pads on a grid (Figure 2.5) according to optimization goals, mainly minimizing wire length. The wire length between blocks is estimated as the shortest distance between their corresponding cells, with the overall wire length being the sum of these individual distances. By minimizing this total wire length, the placement becomes more compact, thereby reducing interconnect delays and enhancing overall efficiency.

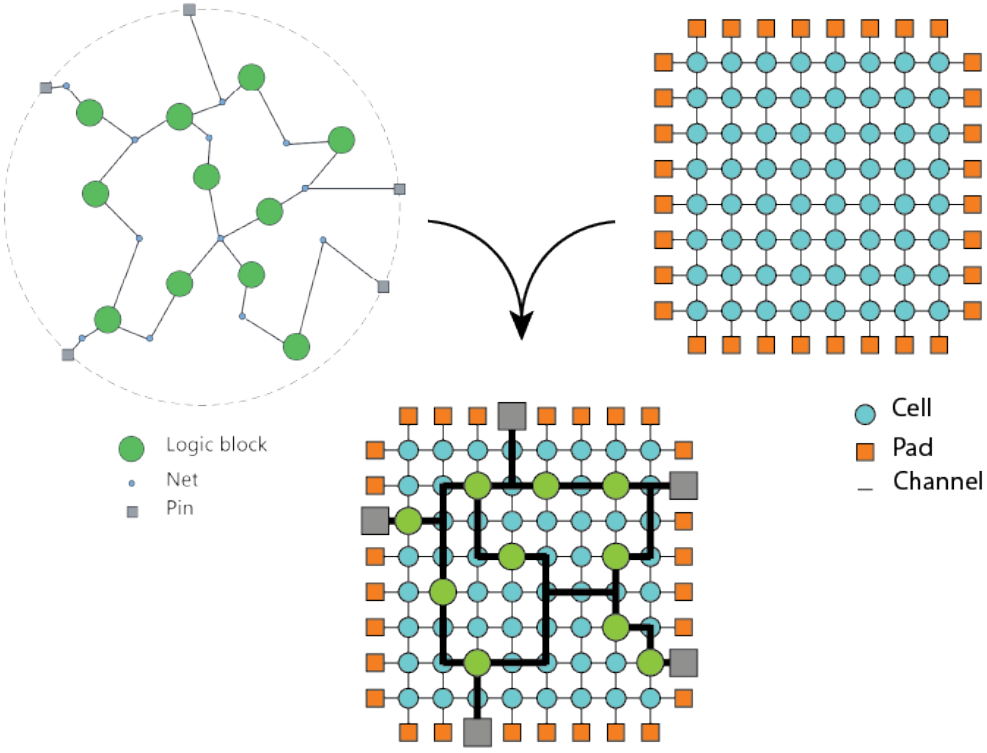


Figure 2.5: Placement of a circuit

2.2.3.2 The Routing Model

The interconnection length is determined once the wire's routing path through the physical architecture is defined. Routing allocates paths within channels, focusing on minimizing the path length—i.e., the number of channel segments

the wire passes through. Although channel occupation could be considered as an additional factor, it is not explicitly modeled due to its complexity and the limitations of estimation methods. Nonetheless, this simplification is not critical, as standard cell designs typically adjust channel widths or layers to accommodate routing requirements. This assumption is supported by the modeling of short interconnects. Figure 2.6 shows the routing of nets within the architectural model.

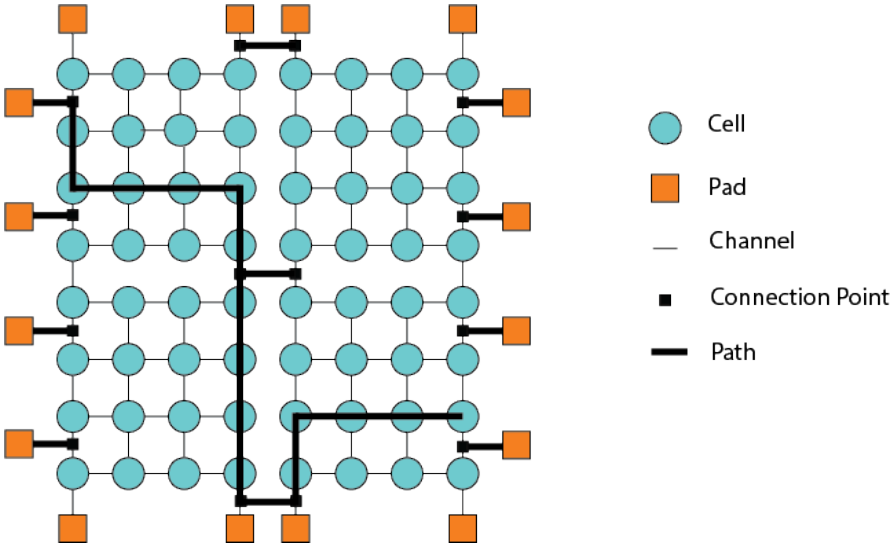


Figure 2.6: Routing a net through the shortest path

Chapter 3

Rent's Rule and Wirelength Estimation techniques

As mentioned previously in Chapter 1, wire length estimation methods are categorized into empirical, procedural, and theoretical approaches. Each has its strengths, but not all are equally suitable for every situation.

Among the theoretical methods, Rent's Rule stands out as a robust framework for wire length estimation. Proposed in the 1960s by Landman and Russo [2] and further developed over decades, it correlates the number of terminals and blocks in a partitioned circuit using a simple mathematical relationship that will be explained in detail in section **??**. Unlike random-placement models, Rent's Rule captures the inherent structure and complexity of circuit designs, making it more accurate and versatile for various architectures.

The work of Donath [3, 4] in the 1970s extended Rent's Rule to estimate wire lengths hierarchically, forming the foundation of modern theoretical estimation techniques. Despite its simplicity, Rent's Rule has proven to be remarkably effective in predicting interconnection requirements across diverse design scales, including 3D architectures and complex multi-terminal nets.

Given the limitations of empirical and procedural methods for early design-stage estimations, Rent's Rule provides an optimal balance of simplicity, accuracy, and adaptability. It forms the basis of this thesis and serves as the starting point for exploring advanced wire length estimation techniques. The following sections are dedicated to explain the implication of Rent's Rule, it's basic understanding and coffecients involved in it.

3.1 Definition and Interpretation

To theoretically validate Rent's rule, consider a circuit placed within a physical architecture, as shown in Figure 3.1. Assume the architecture is sufficiently large such that its boundaries do not influence the analysis. A bounded region within this architecture is defined, containing a statistically homogeneous functional circuit block. Homogeneity implies that parameters such as the average wire length per logic block and the average number of terminals per logic block are independent of the block's position within the region or its surroundings.

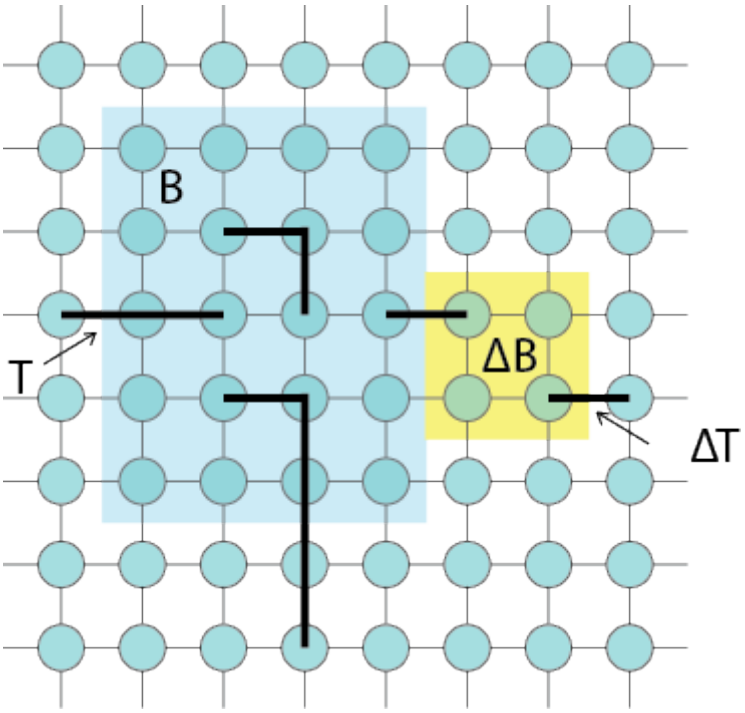


Figure 3.1: Change in bounding box size

Within this bounded region, logic blocks communicate with the external system through T terminals. By slightly expanding the boundary to include an additional ΔB logic blocks, it is reasonable to assume these new blocks will require the same level of communication as the original B blocks. Therefore, the incremental number of terminals required, ΔT, is given by:

$$\Delta T = \frac{T}{B}\Delta B \tag{3.1}$$

This analysis can be refined by considering placement optimization,

represented by the Rent exponent p. The modified expression introduces a proportionality factor, leading to:

$$\Delta T = p\frac{T}{B}\Delta B \tag{3.2}$$

To simplify further, we replace the finite changes ΔB and ΔT with differentials dB and dT, respectively:

$$\frac{dT}{T} = p\frac{dB}{B} \tag{3.3}$$

Solving this differential equation yields:

$$T = tB^p \tag{3.4}$$

Here, t is the constant of integration and represents the average number of terminals per logic block. This final expression corresponds to Rent's rule, with the Rent exponent p reflecting the level of placement optimization in a statistically homogeneous circuit with a given interconnection topology.

3.2 Understanding Rent Exponent

The Rent exponent (p) in Rent's rule offers valuable insights into the placement optimization and interconnection complexity of a circuit. When $p = 1$, there is no placement optimization, and the circuit behaves as if its logic blocks are arranged randomly. This value also represents the upper bound for p, as the maximum number of terminals for any region containing B logic blocks in a homogeneous system is given by $T = tB$.

The lower bound of p is determined by the circuit's interconnection topology, as it is generally impossible to arrange all connected logic blocks adjacently, even with optimal placement. This lower bound, denoted as p^*, is known as the *intrinsic Rent exponent*, a concept introduced in [15]. For circuits with optimal placement, the Rent exponent is equal to p^* and is solely defined by the circuit's topological properties. Therefore, the intrinsic Rent exponent serves as a measure of the complexity of the circuit's interconnection topology [16].

For instance, consider two extremes: a simple chain of logic blocks connected in sequence and a circuit with a highly nested network of loops. Intuitively, the latter has a more complex interconnection structure, and this difference in topological complexity is captured by the intrinsic Rent exponent. Higher values of p^* correspond to greater interconnection complexity. For

connected graphs, p^* typically ranges from 0 to 1. In practice, the intrinsic Rent exponent for typical circuits falls between 0.3 for regular architectures (e.g., RAM) and 0.75 for more complex designs (e.g., high-performance VLSI circuits).

3.3 Rent Characteristics

As previously mentioned, Rent's rule describes the relationship between the number of terminals (T) and the number of blocks (B) in a partitioned circuit. When plotted on a log-log scale, the data points typically align along a straight line, confirming Rent's prediction. The slope of this line represents the Rent exponent (p), which is a key metric for circuit complexity. Figure 3.2 illustrates this behavior for a 64-bit arm core with about 700k cells, where the estimated Rent exponent is $p = 0.424$.

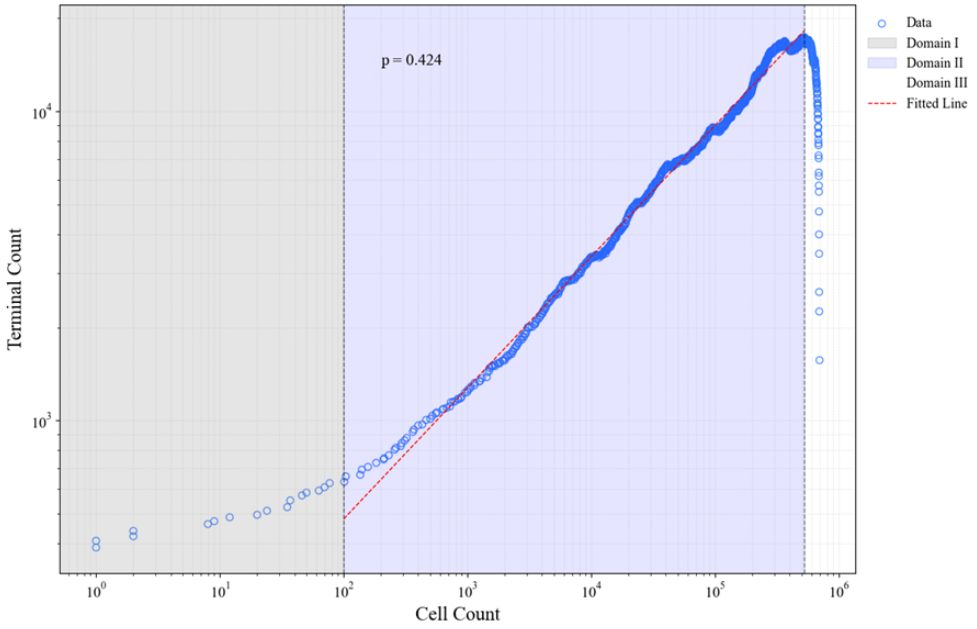


Figure 3.2: Cell count vs terminal count for a of 64-bit arm core with ~700k instances

However, deviations from this behavior occur in specific regions, referred to as Rent regions. These deviations, discussed below, reveal additional insights into circuit design constraints and interconnection complexity.

3.3.1 Rent Region I

In Region I, Rent's rule is most consistent, with a clear linear relationship between the number of terminals and blocks. This region typically represents the majority of the circuit's behavior, where the Rent exponent provides an accurate measure of interconnection complexity. It is here that the intrinsic properties of the circuit's topology dominate.

3.3.2 Rent Region II

Region II corresponds to the largest module sizes, where the number of terminals is significantly lower than predicted by Rent's rule. This deviation occurs due to physical and design constraints, such as the limitation of circuit pins. Since the number of pins grows with the square root of the number of blocks (proportional to the chip's boundary), it often increases slower than required. Designers address this by reducing interconnection complexity through techniques such as technology mapping, parallel-to-serial conversion, encoding, and block duplication. These methods minimize pin usage while maintaining functionality, thereby altering Rent's rule at higher levels of hierarchy.

3.3.3 Rent Region III

Region III is observed for small module sizes, where the number of terminals per block exceeds the predictions of Rent's rule. This deviation arises because the complexity of interconnections at low hierarchy levels is often constrained by the implementation technology rather than circuit topology. For instance, logic blocks in most circuits typically have one output terminal but multiple input terminals. A mismatch between available and required interconnections results in a relatively higher terminal count. This effect diminishes as module sizes increase, allowing Rent's rule to reassert itself for larger partitions.

Figure 3.2 captures these three distinct regions, providing a comprehensive view of Rent's behavior across various module sizes. Different methods to extract this *rent plot* will be explained in Chapter 4.

3.4 Wire Length Estimation Techniques

Wire length estimation plays a fundamental role in the design and optimization of VLSI circuits. Accurate estimations are crucial for predicting circuit

performance, power consumption, and manufacturability during the early design stages. This chapter presents two prominent approaches to wire length estimation: Donath's hierarchical placement model and Davis' stochastic wire length distribution. The theoretical underpinnings of these methods are explored here, while their practical implementation will be detailed in the next chapter.

3.4.1 Donath's Wire Length Estimation

Donath's method, introduced in the late 1970s [3, 4], was one of the first systematic approaches for wire length estimation. It is based on a hierarchical placement model, which uses recursive partitioning of the circuit and its physical architecture into smaller subregions. The model assumes a Manhattan grid as the underlying architecture and minimizes wire lengths by using Rent's rule to maintain placement optimization.

Empirical studies conducted on real computer systems have shown that the distribution of interconnection lengths can be approximated by the following function:

$$H(l) \approx \begin{cases} Bl^{-\gamma}, & \text{for } 1 \leq l \leq l_c \\ 0, & \text{otherwise,} \end{cases} \tag{3.5}$$

where l represents the interconnection length measured in circuit pitches, B is a constant, γ is a parameter that characterizes the logic, and l_c denotes a cut-off length. Furthermore, a straightforward relationship has been observed between the exponents of the Rent equation and the length distribution function, expressed as $\gamma = 3 - 2p$.

3.4.1.1 Hierarchical Placement Model

The hierarchical placement model begins by partitioning the circuit into four equal subcircuits, which are then mapped onto four corresponding subregions of a Manhattan grid. This recursive partitioning continues until each subregion contains a single logic block. At each level of hierarchy, the number of terminals and the interconnections between subregions are determined using Rent's rule. This rule ensures that the interconnection complexity is proportional to the size of the subcircuits.

Figure 3.3 illustrates the hierarchical partitioning process. At every step, the model aims to place densely connected logic blocks close to each other to minimize interconnection lengths. The final placement reflects an optimized

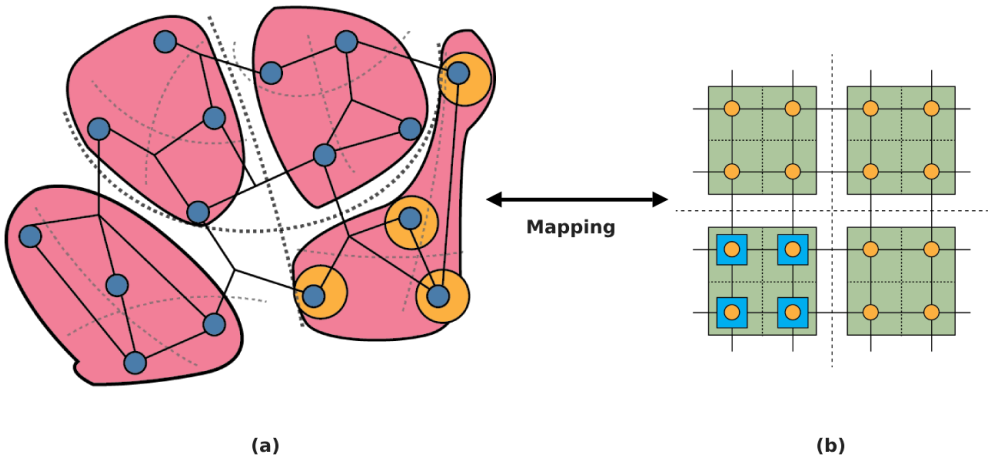


Figure 3.3: Donath's placement model: (a) recursive partitioning of a circuit (b) Manhattan grid mapping of the same circuit [17]

spatial distribution of logic blocks, resulting in a wire length distribution that approximates real-world layouts.

3.4.1.2 Donath's Wire Length Distribution Function

P. Christie *et.al* calculated the wirelength distribution at each hierarchical level [18]. The total number of interconnections at a given level k can be expressed through a probability density function $P_k(l)$, where l denotes the interconnection length measured in circuit pitches. The function $P_k(l)$ represents the fraction of interconnections with length l at a specific level k. This can be formulated as:

$$P_k(l) = \frac{\text{Number of wires of length } l \text{ within a fully interconnected array}}{\text{Total number of wires}} \tag{3.6}$$

The total number of wires at level k, denoted as $h_k(l)$, is then given by the product of $P_k(l)$ and the number of interconnections n_k:

$$h_k(l) = n_k P_k(l) \tag{3.7}$$

where the total number of interconnections at the k-th hierarchical level is given by:

$$n_k = AC \cdot p \cdot (1 - 4^{p-1}) \cdot 4^{k(p-1)}, \tag{3.8}$$

where:

- A: A proportionality constant dependent on the circuit structure and Rent's rule,
- C: Total number of gates in the circuit, given as $C = 4^L$,
- k: The current hierarchical level, with $0 \leq k < L$,
- p: Rent's exponent, describing the relationship between the number of gates and interconnections.

To compute the overall length distribution $H(l)$ across all levels, the contributions from each hierarchical level are summed:

$$H(l) = \sum_{k=0}^{L-1} h_k(l) \tag{3.9}$$

where L is the number of hierarchical levels.

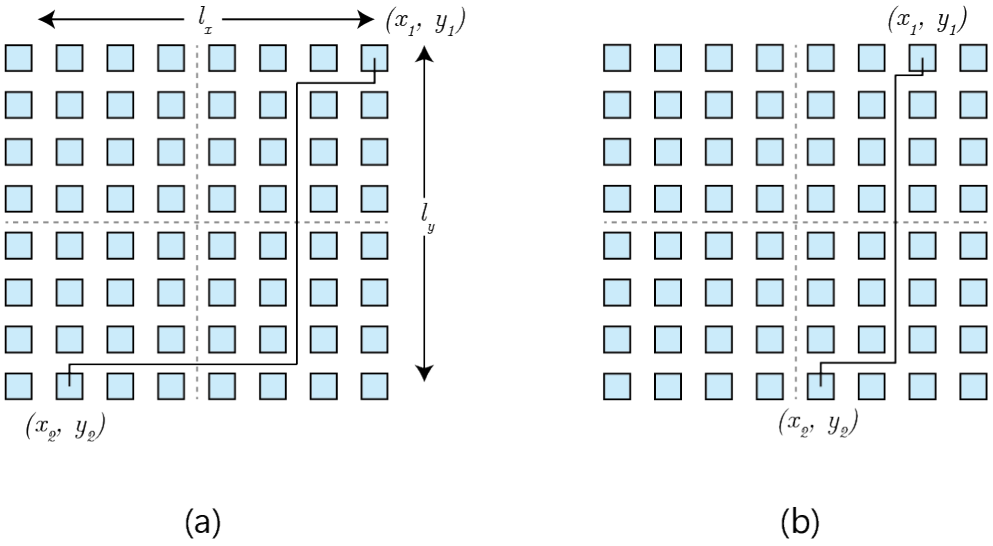


Figure 3.4: (a) Diagonal interconnections (b) Adjacent interconnections [18]

To derive $P_k(l)$, the interconnections are classified into diagonal and adjacent interconnections, as shown in Fig. 3.4. The diagonal interconnections, as indicated in Fig. 3.4(a), occur between gates at positions (x_1, y_1) and (x_2, y_2). The length l of such interconnections is determined as:

$$l = |x_1 - x_2| + |y_1 - y_2| \tag{3.10}$$

Assuming uniform gate distribution and independent placement, the probability distributions P_{x_d} and P_{y_d} for the x- and y-axis contributions are identical. The diagonal distribution $P_k(l)$ is derived as:

$$P_{k_d} = P_{x_d} * P_{y_d} \tag{3.11}$$

For the adjacent interconnections, as depicted in Fig. 3.4(b), the length l is calculated similarly. Combining these diagonal and adjacent probabilities yields the total probability density function for interconnections at level k:

$$P_k(l) = \frac{2P_{k_d}(l) + 4P_{k_a}(l)}{6} \tag{3.12}$$

Where, the function $P_k(l)$ is defined as:

$$P_k(l) = \begin{cases} \frac{-l^3+4al^2+l}{6a^4}, & \text{for } 0 \leq l \leq a \\ \frac{5l^3-36al^2+(72a^2-5)l-32a^3+8a}{18a^4}, & \text{for } a \leq l \leq 2a \\ \frac{-l^3+12al^2-(48a^2-1)l+64a^3-4a}{18a^4}, & \text{for } 2a \leq l \leq 4a \\ 0, & \text{otherwise.} \end{cases} \tag{3.13}$$

where $a = \sqrt{N}$ and N is the total number of cells. The overall system length distribution function $H(l)$ is computed by summing the level-wise distributions. For example, using Rent's parameters for hierarchy depth L and gate block sizes, the distribution $H(l)$ can be visualized as in Fig. 3.5, where $H(l)$ rapidly decreases as l increases.

Finally, the expectation of the interconnection length $\langle l \rangle$ is evaluated to validate the distribution:

$$\langle l \rangle = \sum_{l=0}^{L-1} l \cdot H(l) \tag{3.14}$$

This derivation method confirms the adherence to Donath's hierarchical framework, ensuring that the interconnection length distribution aligns with Rent's partitioning model.

3.4.2 Davis' Wire Length Estimation

Davis' stochastic wire length model [5] builds upon Donath's approach by incorporating probabilistic principles to address the shortcomings of deterministic methods. This model derives a complete wire length distribution

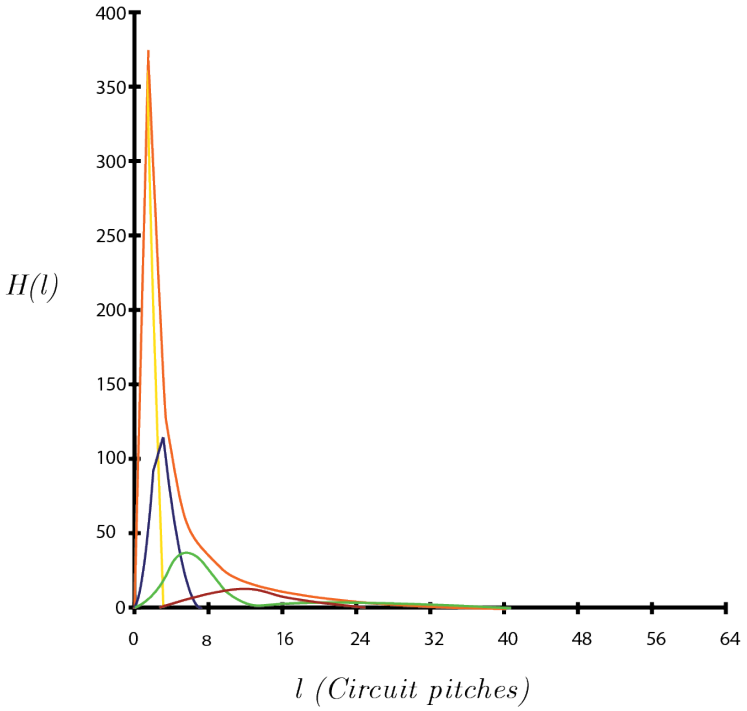


Figure 3.5: System length distribution function [18]

using Rent's rule as its foundation and introduces stochastic variables to capture the variability in interconnection lengths. This model provides a comprehensive framework for estimating local, semi-global, and global interconnect requirements, which are essential for the design of future integrated circuits.

3.4.2.1 Site Density Function / Structural Distribution

The site density function, also referred to as the structural distribution, describes the spatial arrangement of interconnects within the logic network. It quantifies the number of gate pairs separated by a specific distance ℓ in a square array of N gates. The number of gate pairs $M(\ell)$ separated by a Manhattan distance ℓ is given by:

$$M(\ell) = \begin{cases} \frac{\ell^3}{3} - 2\ell^2\sqrt{N} + 2\ell N, & \text{for } 1 \leq \ell < \sqrt{N} \\ \frac{1}{3}(2\sqrt{N} - \ell)^3, & \text{for } \sqrt{N} \leq \ell < 2\sqrt{N} \end{cases} \tag{3.15}$$

This function captures the structural distribution of interconnects, which

is essential for understanding the spatial density of wires in the network.

3.4.2.2 Occupational Probability / Probability Distribution

The occupational probability, or the probability distribution, describes the likelihood of an interconnect having a specific length ℓ. The occupational probability is derived from *law of conservation of terminals* [5], and is given by:

$$I_{\text{exp}}(\ell) = \frac{\alpha k}{2\ell} \left[(1 + \ell(\ell - 1))^p - (\ell(\ell - 1))^p + (\ell(\ell + 1))^p - (1 + \ell(\ell + 1))^p\right] \tag{3.16}$$

Here, α is the fraction of on-chip terminals that are sink terminals.

3.4.2.3 Interconnect Density Function

The interconnect density function (i.d.f.) quantifies the number of interconnects per unit length and is derived by combining the site density function and the occupational probability. A visual representation of these two distributions is shown in Figure 3.6. The i.d.f. is defined as:

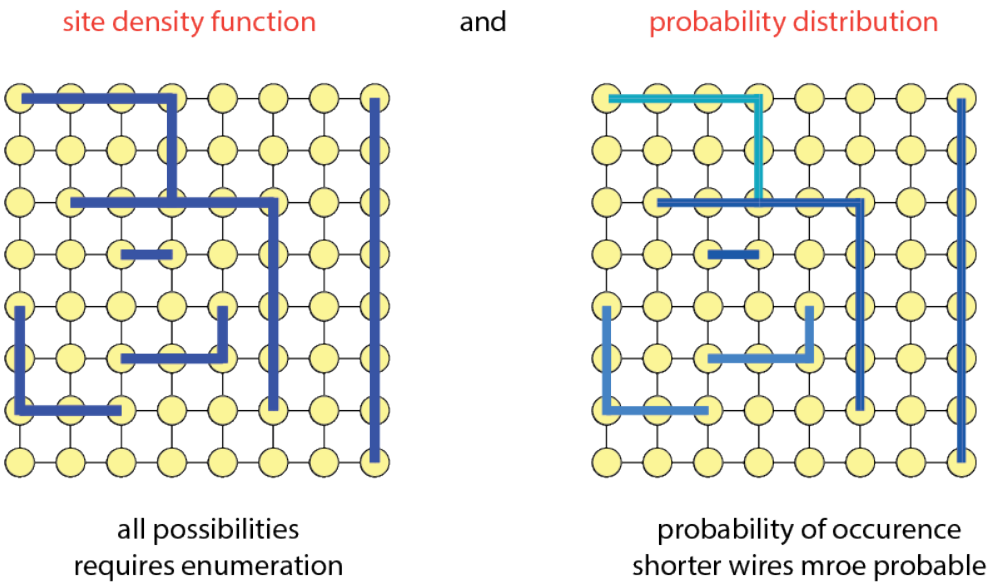


Figure 3.6: Site density function and probability distribution visualized [19]

$$i(\ell) = M(\ell) \cdot I_{\text{exp}}(\ell) \tag{3.17}$$

Substituting the expressions for $M(\ell)$ and $I_{\text{exp}}(\ell)$, the i.d.f. becomes:

$$i(\ell) = \begin{cases} \frac{\alpha k}{2}\left(\frac{\ell^3}{3} - 2\sqrt{N}\ell^2 + 2N\ell\right)\ell^{2p-4}, & \text{for } 1 \leq \ell < \sqrt{N} \\ \frac{\alpha k}{6}(2\sqrt{N} - \ell)^3\ell^{2p-4}, & \text{for } \sqrt{N} \leq \ell < 2\sqrt{N} \end{cases} \tag{3.18}$$

where α is the fraction of on-chip terminals that are sink terminals, and Γ is the normalization factor. The i.d.f. provides a continuous description of the interconnect distribution, enabling the calculation of the total number of interconnects within a specific length range:

$$I(a < \ell < b) = \int_a^b i(\ell)\, d\ell \tag{3.19}$$

Davis' wire-length estimation model offers a robust framework for predicting interconnect requirements in GSI systems. By integrating the concepts of structural distribution, occupational probability, and interconnect density, this model provides a comprehensive tool for analyzing and optimizing the wiring requirements of integrated circuits. The closed-form analytical expressions derived from this model facilitate efficient computation and simulations.

Chapter 4

Rent Coefficient Extraction: Methods and Results

This chapter delves into the methodology for determining the Rent exponent (p) and average terminals per cell (t) from physical architectures, based on Rent's Rule. Rent's Rule, a power law that relates the number of cells to the number of terminals in a design, forms a straight line when plotted on a log-log scale. Accurate extraction of Rent coefficients requires extensive data collection for various partitions of the architecture, ensuring a comprehensive coverage of the physical implementation.

The extraction process begins with accessing physical architectures using imec's NoMachine (remote host) and Cadence Innovus. Through custom TCL scripts, key architectural data such as the number of cells, number of nets, and their respective locations are retrieved. This information is partitioned either through TCL scripts or Python algorithms, enabling the identification of the number of cells and corresponding terminals within defined boundaries. Here, the term "terminals" does not refer to the total physical terminals of the architecture but rather to the nets emanating from virtual boundaries around groups of cells. The data obtained is stored in CSV format for further processing.

The log-log Rent plot is generated using Python libraries such as Pandas and Matplotlib, providing a visual representation of the relationship between cells and terminals. The Rent coefficients are extracted by calculating the slope and intercept of the linear region (Region 2) of the Rent plot. The slope corresponds to the Rent exponent (p), while the intercept provides the average terminals per cell (t).

In this chapter, we describe two approaches for partitioning the design

and extracting Rent coefficients: the *Growing Box Method* and the *Recursive Splitting Method*. Each method is detailed in subsequent sections, outlining their respective advantages, implementation steps, and relevance to Rent analysis. This systematic approach provides a robust framework for Rent coefficient extraction, ensuring accuracy and scalability for complex architectures.

4.1 Growing Box Method

The Growing Box Method is a systematic approach for extracting Rent coefficients by iteratively growing a virtual box from the center of the physical architecture to the chip periphery. This method provides a straightforward way to analyze the relationship between the number of cells contained within the box and the unique nets crossing its boundaries, which is critical for generating data for Rent's Rule analysis.

The methodology is implemented entirely in Tcl within Cadence Innovus, where the physical architecture is loaded for computation. The process begins by retrieving the boundary dimensions of the design and calculating the coordinates of its center. A virtual box is initialized at the center, with an initial side length that determines the number of cells contained in the smallest data point.

At each iteration, the following steps are performed:

- The corner coordinates of the virtual box are stored, along with the coordinates of all cells and nets in the design.
- The number of cells inside the virtual box is determined by comparing the coordinates of the box to the coordinates of the cells.
- The number of unique nets crossing the boundaries of the box is calculated by analyzing the net coordinates relative to the boundary of the box:
  - If a net crosses only one side of the box, it is counted as one.
  - If a net crosses two sides of the box, it is counted as two.
- The size of the box is incremented linearly at each step, controlling the resolution and the number of data points.

The iteration continues until the number of unique nets crossing the boundary starts to decrease, signaling the end of Region 2 in the Rent plot

and the beginning of Region 3. The extracted data—representing the number of cells and corresponding terminals for each virtual box size—is stored in a CSV file for subsequent analysis.

In Python, the data is plotted on a log-log scale using Pandas and Matplotlib libraries. The slope of the curve in Region 2 provides the Rent exponent (p), while the intercept indicates the average number of terminals per cell (t). For the purpose of our consequent implementation of wire length distribution modeling, we will just focus on the rent exponent. An example Rent plot for a 64-bit ARM core design with approximately 700,000 cells demonstrates the effectiveness of the method, where the rent exponent was 0.424 (Figure 4.1).

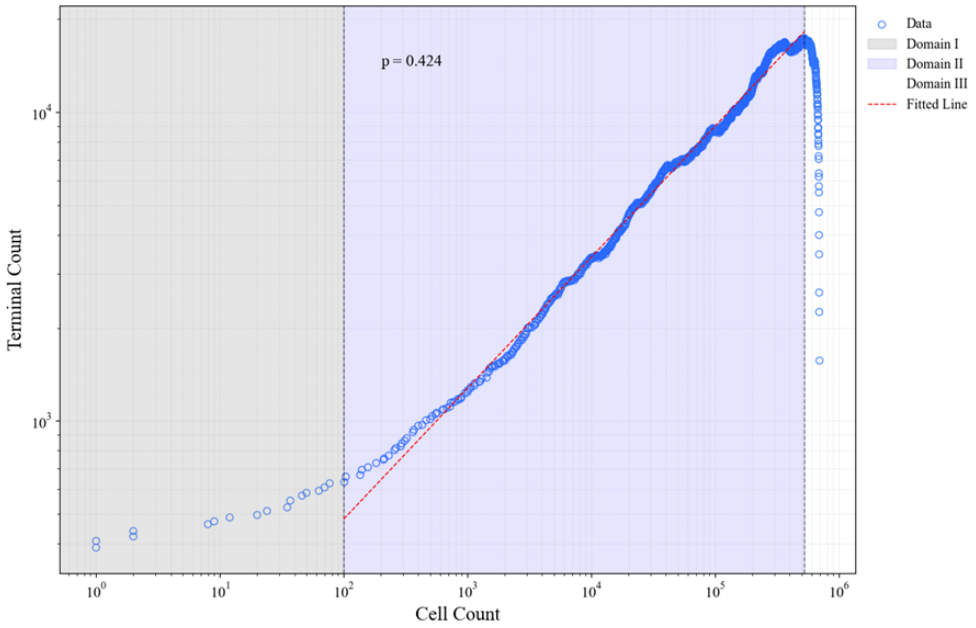


Figure 4.1: Rent plot- Cell count vs Terminal count for a of 64-bit arm core with about 700k instances

While the computation in Tcl provides accuracy, it is relatively slow for architectures with a very large number of cells and nets due to the iterative nature of the algorithm. For example, processing a design with 700,000 cells required approximately 180 minutes. A summary of computation times for various designs is provided in Table 4.1.

The flow chart showing the procedure and the Rent plot is shown in Figure 4.2. This method, while computationally intensive for larger designs, provides a reliable framework for Rent coefficient extraction.

Design	Number of Cells	Computation Time (min)
64-bit ARM Core Design 1	∼700k	180
64-bit ARM Core Design 2	∼700k	180
ARM M0 Design 1	∼16k	20
ARM M0 Design 2	∼16k	20

Table 4.1: Computation Times for Growing Box Method

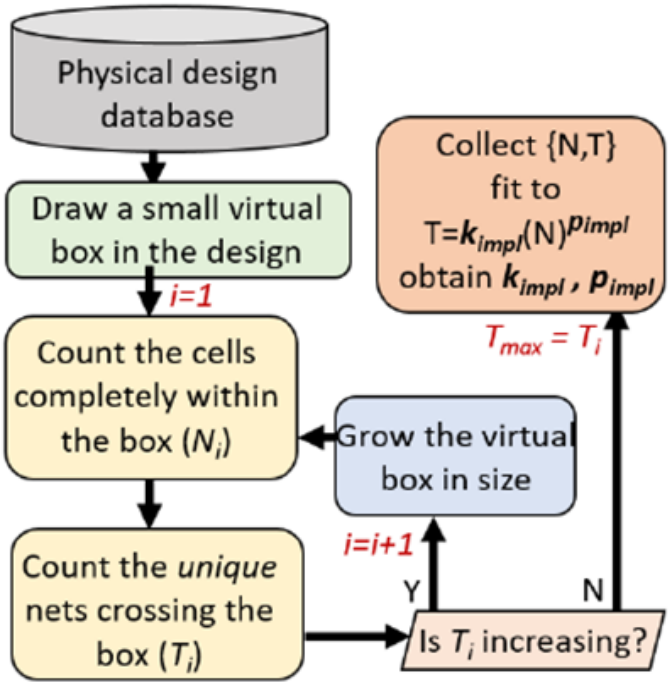


Figure 4.2: Growing box method flow chart [7]

4.2 Recursive Splitting Method

The Recursive Splitting Method is an iterative approach for extracting Rent coefficients, inspired by the recursive partitioning of circuit graphs in Donath's hierarchical placement model (3.4.1.1). However, in this thesis, the method is applied directly to the physical architecture, as the focus is on a posteriori estimation rather than a priori modeling. A visual representation of the recursive splitting process is shown in Figure 4.3.

The implementation begins with storing design data in a CSV file using TCL code in Cadence Innovus. This CSV file contains the names of all cells and nets, their respective x and y coordinates, and the chip's boundary

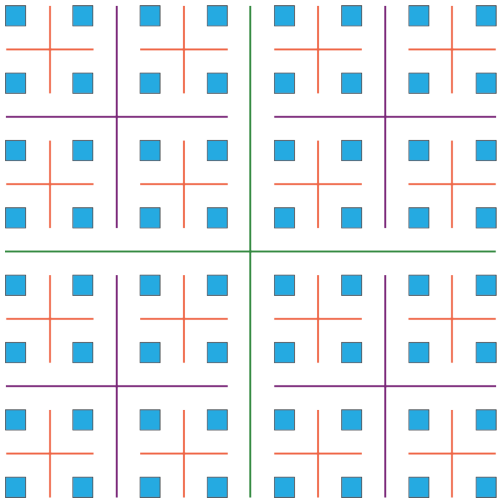


Figure 4.3: Recursive partitioning of a grid in 4 sub modules

coordinates. Due to the complexity of the code compared to the Growing Box Method, the subsequent computations and visualizations were performed entirely in Python.

In Python, the CSV file was loaded using Pandas, and the data was organized into NumPy arrays for efficient computation. From the boundary coordinates of the chip, the midpoint of the design was calculated, and the chip was partitioned into four quadrants (or sub-regions). For each sub-region:

- The number of cells within the virtual box was counted by comparing their coordinates with the sub-region boundaries.
- The number of unique nets crossing the boundaries of the virtual box was calculated:
  - If a net crosses only one side of the box, the terminal count is incremented by one.
  - If a net crosses two sides of the box, the terminal count is incremented by two.

At the first iteration, the four quadrants provide four data points for the Rent plot. In subsequent iterations, each of these sub-regions is recursively partitioned into four smaller sub-regions, and the number of cells and terminals are counted again. At each iteration level, the number of data points equals 4^L, where L is the iteration level, starting from 1 (Level 0, corresponding to the entire chip, is excluded as it lies outside Region 2 and provides only a single data point).

For the example of a 64-bit ARM core design with approximately 700,000 cells, five levels of iteration were performed, resulting in data points across Region 2 of the Rent plot. The Rent plot for this design is shown in Figure 4.4, where the Rent exponent p was calculated as 0.43. This value is consistent with the p value obtained using the Growing Box Method.

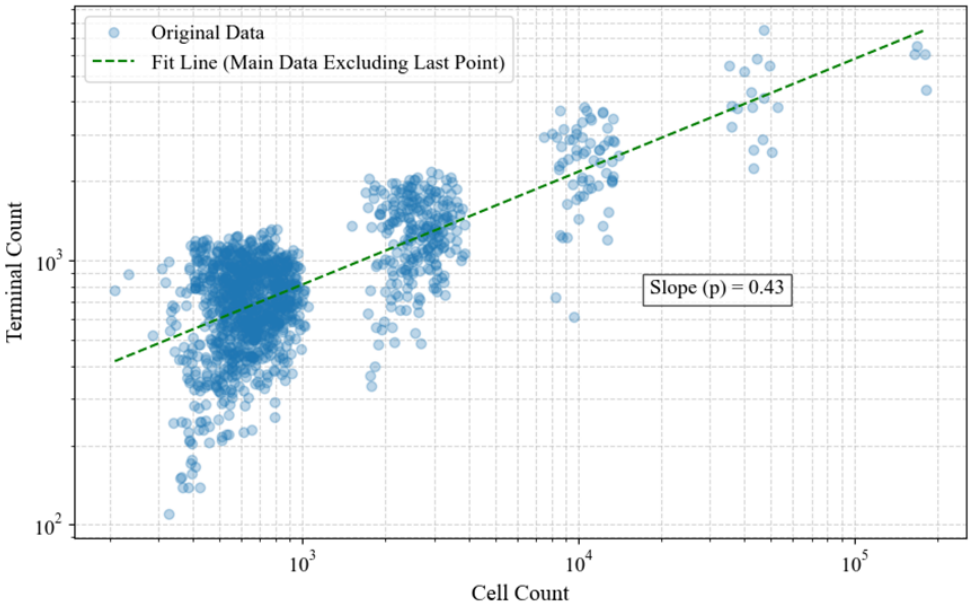


Figure 4.4: Rent plot- Cell count vs Terminal count for a 64-bit arm core with about 700k instances

Design	**Number of Cells**	**Computation Time (min)**
64-bit ARM Core Design 1	∼700k	15
64-bit ARM Core Design 2	∼700k	15
ARM M0 Design 1	∼16k	5
ARM M0 Design 2	∼16k	5

Table 4.2: Computation Times for Recursive Splitting Method

The Recursive Splitting Method proved to be computationally efficient compared to the Growing Box Method. While the latter required approximately 180 minutes to process a design with 700,000 cells, the Recursive Splitting Method completed the computation in just 15 minutes for the same design. A summary of the computation times for various designs is provided in Table 4.2.

Chapter 5

Wire length Distribution Models: Methods and Results

In the previous chapters, the concept of the Rent exponent, its extraction methods, and its utilization in wire length distribution models such as those proposed by Donath and Davis was explored. Building on this foundation, this chapter focuses on the implementation of these models in Python and evaluates the reasonableness of their outputs.

The chapter begins by detailing the procedure to extract actual wire length data from Cadence Innovus, emphasizing the steps required to process and validate the data. Followed by it, Donath's and Davis' models are implemented. Additionally, challenges associated with binning data for probability density functions (PDFs) are discussed, and the rationale for using cumulative distribution functions (CDFs) as a more robust alternative is presented.

Improvements to the existing models are explored, particularly through adjustments to the exponent of the power-law distribution relation. These modifications address discrepancies observed in the models' predictions when compared to empirical data. Furthermore, enhancements through the *variable cell size model* are introduced, with a particular focus on capturing the behavior of very short interconnects that are not well-represented in traditional approaches.

Through Python-based implementation and analysis, this chapter demonstrates how theoretical models can be adapted and refined to better align with real-world data, providing valuable insights into wire length distribution in modern physical architectures. The results and findings presented in this chapter set the stage for more accurate and scalable interconnect prediction

methods in future work.

5.1 Actual Wire Length Distribution Extraction

To analyze the actual wire length distribution, the designs were opened in Cadence Innovus, and the net data was extracted using TCL scripts. For each design, the number of nets, their respective lengths, and net names were retrieved and stored in a CSV file. This data served as the foundation for plotting and analyzing the wire length distribution.

The CSV file was processed in Python using the Pandas library, and the net lengths were stored in a NumPy array for efficient computation. To bin the data for the wire length distribution, the number of bins, bin counts, and bin edges were defined. The binning process was carefully executed to ensure meaningful distribution data.

To address issues associated with non-physical or zero-length nets and ensure proper representation on log-log plots, a filter was applied to exclude net lengths equal to or less than zero. This filtering step was crucial for eliminating noise and ensuring the data accurately reflected the physical characteristics of the design. The resulting distributions were plotted using Python libraries such as Matplotlib, focusing on the designs already mentioned in earlier sections (Figure 5.1).

The following table summarizes the relevant information for the analyzed designs, including the number of cells, number of nets, and Rent exponent. These values provide context for interpreting the wire length distributions and assessing the agreement with the models discussed in subsequent sections.

Table 5.1: Summary of design characteristics used for wire length distribution extraction.

Design	Number of Cells	Number of Nets	Rent Exponent	Slope of the Distribution
64-bit arm core design 1	∼700k	631217	0.43	-1.58
64-bit arm core design 2	∼700k	545912	0.413	-1.52
arm M0 design 1	∼16k	16257	0.378	-1.49
arm M0 design 2	∼16k	15668	0.35	-1.45

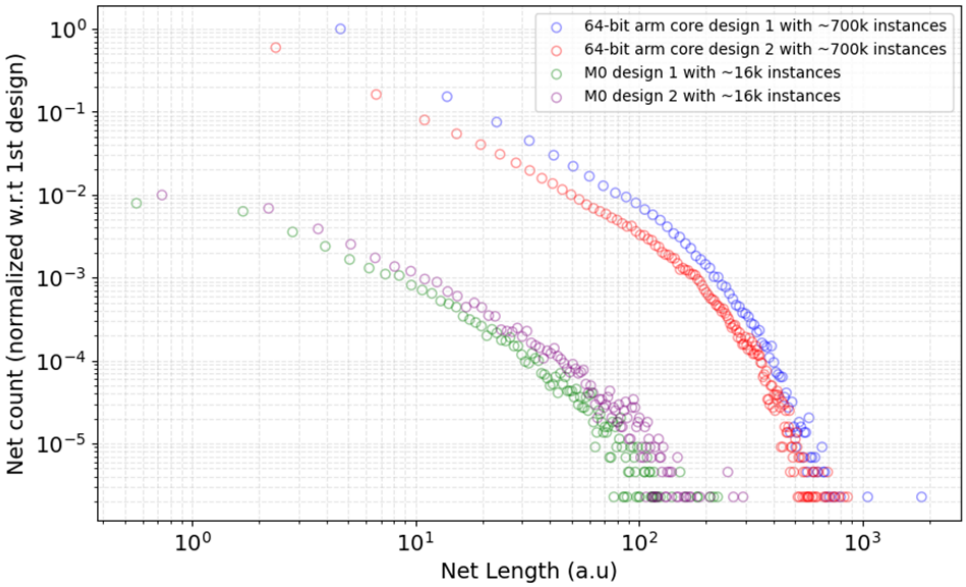


Figure 5.1: Actual wire length distribution of different designs (log-log)

5.2 Implementing Donath's Wire Length Distribution Model

The derivation of Donath's wire length distribution model was previously explained in Chapter 3. While Donath's original approach began with the netlist or circuit graph, this thesis deviates by utilizing the physical implementation of the design. For this purpose, the recursive splitting of the architecture, as described in Section 4.2, serves as the foundational data source for implementing the model.

The sorted data of hierarchical levels and corresponding cell counts, obtained through the recursive splitting method, is loaded into Python. Although the raw data is expressed in microns, computations are simplified by using gate pitches as the unit of measurement. A gate pitch is defined as the minimum distance between two adjacent cells and can be expressed as $\sqrt{\text{chip area}/\text{total cells}}$. Assuming an equidistant arrangement of N cells, the side length of the design can be approximated as $\sqrt{N}$. This approximation enables efficient computation of the wire length distribution.

In Python, a function was created to compute the number of nets at the k-th hierarchical level based on Equation 3.8. Another function was implemented to compute the probability density function p_k for each hierarchical level, as described by Equation 3.13.

For each level k, the normalized probability function p_k is multiplied by the total number of nets at that level to compute the wire length distribution h_k for that specific level. This process is iterated from level 0 to level 5, and the final system-wide distribution is obtained by summing the distributions h_k across all levels. A flowchart of the process is shown in Figure 5.2.

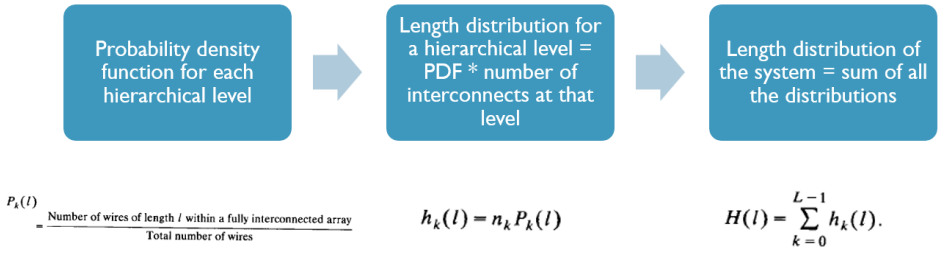
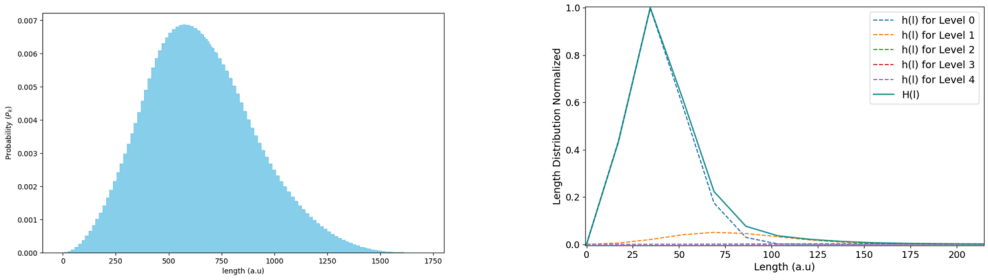


Figure 5.2: Donath's wire length distribution model flow chart

The resulting distributions were visualized using Matplotlib. A specific example of the probability density function (PDF) at level 4 is shown in Figure 5.3a. The total wire length distribution for the entire design is illustrated in Figure 5.3b.



(a) (b)

Figure 5.3: (a) Probability density function for k = 4 Level (b) Length distribution of hierarchical levels and total system distribution (zoomed in)

As an example, for the 64-bit ARM core design, the system-wide distribution was analyzed using a log-log plot. The slope of the distribution curve was extracted, as shown in Figure 5.4. A simple comparison with the actual slope obtained from the extracted wire length data (refer to Table 5.1) revealed significant discrepancies between the theoretical model and empirical data.

Detailed reasoning for this discrepancy, including potential issues with model assumptions and physical implementation constraints, will be discussed

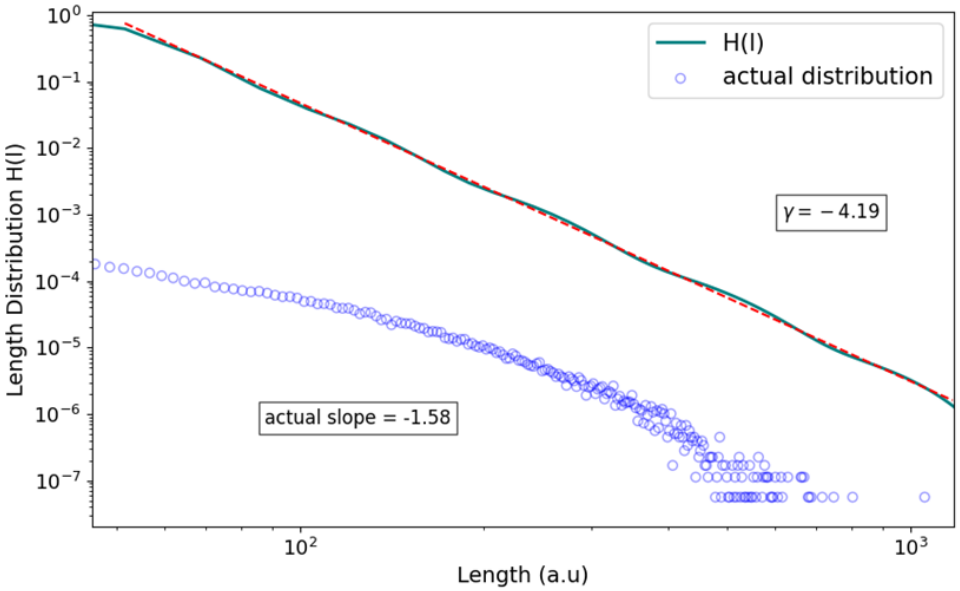


Figure 5.4: Donath's wire length distribution vs actual distribution for a 64-bit arm core with 700k instances (log-log)

in chapter 6.

5.3 Implementing Davis' Wire Length Distribution Model

In Chapter 3, we explained the core principles of Davis' wire length distribution model. Unlike Donath's model, Davis' approach decomposes wire length distribution into two separate components: the *structural distribution* and the *occupational probability*. The total wire length distribution is given as the product of these two components. The mathematical expressions for these distributions are provided in Equations 3.15 (structural distribution), 3.16 (occupational probability), and 3.18 (total wire length distribution).

5.3.1 Parameter Extraction and Preprocessing

Before implementing the model in Python, key parameters were extracted from Innovus using TCL scripts. These parameters include:

- **Total number of cells** in the design.
- **Average fanout** of nets.

- **Rent exponent**, which can be obtained using either of the two extraction methods described in Chapter 3.

Once extracted, the data was stored in a CSV file and subsequently loaded into Python using Pandas.

5.3.2 Implementation of the Model in Python

A function was developed in Python to compute the occupational probability based on Equation 3.16. This function takes two inputs: the Rent exponent and the total number of cells in the design. Similarly, another function was implemented for the structural distribution, following Equation 3.15. This function requires two parameters: the total number of cells and the length of a net.

5.3.2.1 Computing Structural Distribution

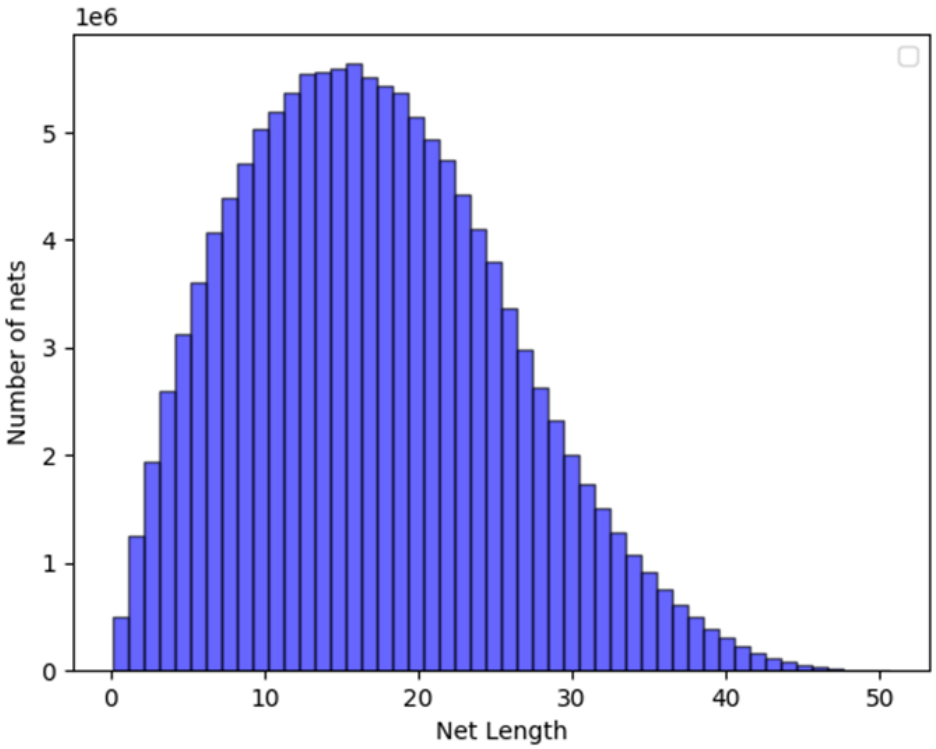


Figure 5.5: Structural distribution for M0 design with 16k instances

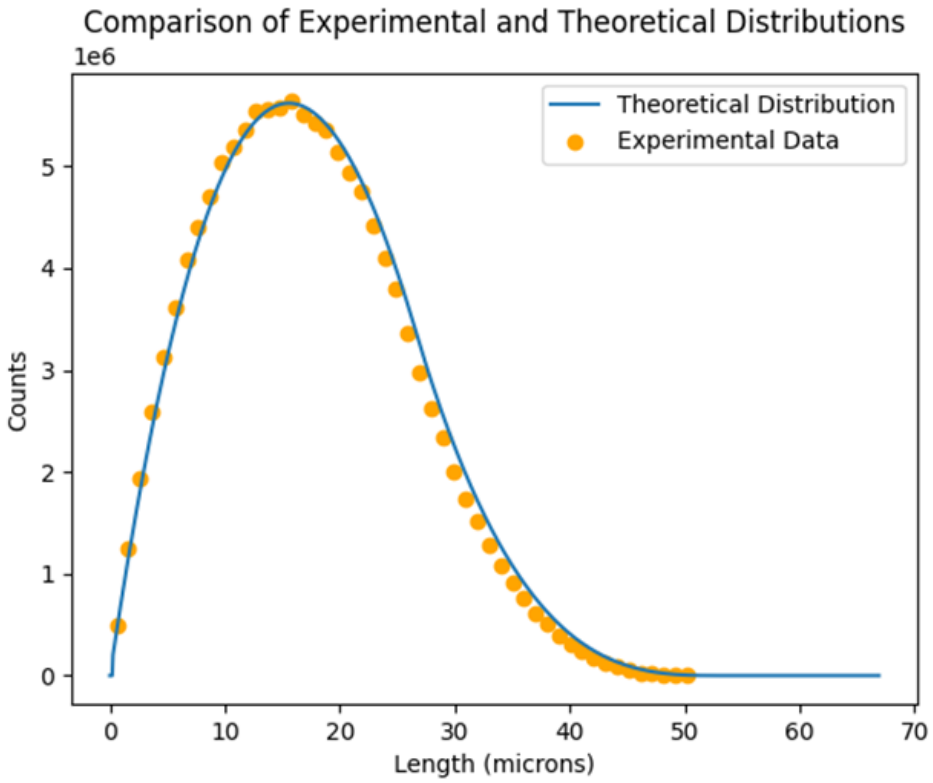


Figure 5.6: Site density function for M0 design and theoretical distribution for same number of cells

To further refine the model for specific designs, an alternative approach was developed to calculate the structural distribution empirically. In this approach, each cell was enumerated with every other cell in the design to determine interconnect distances. The steps involved in this computation are as follows:

1. A nested iteration was performed where each cell's Manhattan distance to every other cell was computed and stored.

2. To prevent redundant computations, once a cell's distances were calculated, it was removed from subsequent iterations.

3. This process continued until all cells were processed.

Figure 5.5 illustrates the structural distribution computed for an ARM M0 design with approximately 16k cells. As evident from Figure 5.6, the analytical function derived by Davis fits well over the computed distribution.

5.3.2.2 Computing the Final Distribution

A normalization function was also implemented, which takes the Rent exponent and total number of cells as input. A list of net lengths was defined, ranging from 0 to a maximum value of $2\sqrt{\text{total number of cells}}$. The following steps were carried out to compute the final wire length distribution:

1. For each net length, the values of the structural distribution and occupational probability were computed.
2. These values were multiplied together and stored in a list.
3. The final curve was plotted using Matplotlib.

Figure 5.7 shows the computed wire length distribution for the ARM M0 design. The log-log plot reveals that Davis' model provides a reasonable prediction for larger interconnect lengths. However, severe deviations are observed for shorter interconnects.

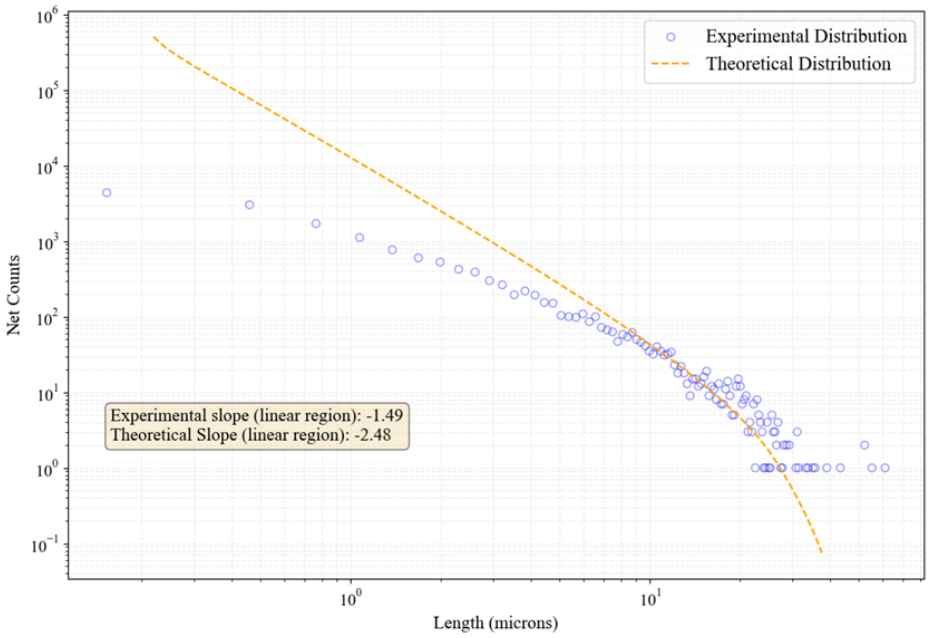


Figure 5.7: Experimental and theoretical wire length distributions for M0 design with 16k instances

One major source of discrepancy in short interconnect predictions stems from the way actual data is binned and plotted. The binning process can significantly impact both the slope and the maximum number of nets in

the probability density function (PDF). In contrast, the theoretical curve is independent of such extraction concerns. Hence, the following approaches will use CDFs instead.

5.4 Proposed Model for Wire Length Distribution

Following the methodology of previous sections, net length data was extracted from Innovus using a Tcl script and stored in a CSV file. This file was then processed in Python for further analysis. The cumulative distribution functions (CDFs) of the four designs considered throughout this study were computed and normalized, as shown in Figure 5.8.

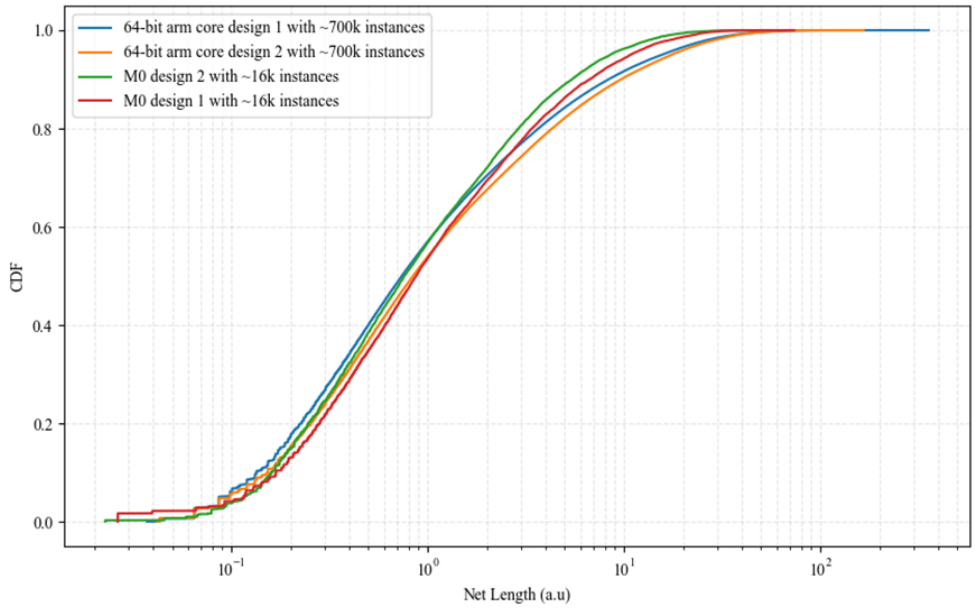


Figure 5.8: Cumulative distribution functions for different designs

From Figure 5.8, it can be observed that for shorter interconnects, variations among the different designs are minimal. However, as the interconnect length increases, noticeable differences emerge. This behavior highlights the need for accurate modeling, particularly for longer wire lengths.

5.4.1 Converting the Davis Model to CDF

Before constructing a new CDF-based model, the existing Davis probability density function (PDF) model was converted to a cumulative distribution function and normalized. The effects of varying Rent exponent p and the average number of terminals per block t were examined. The results of this analysis for the ARM M0 design are shown in Figure 5.9.

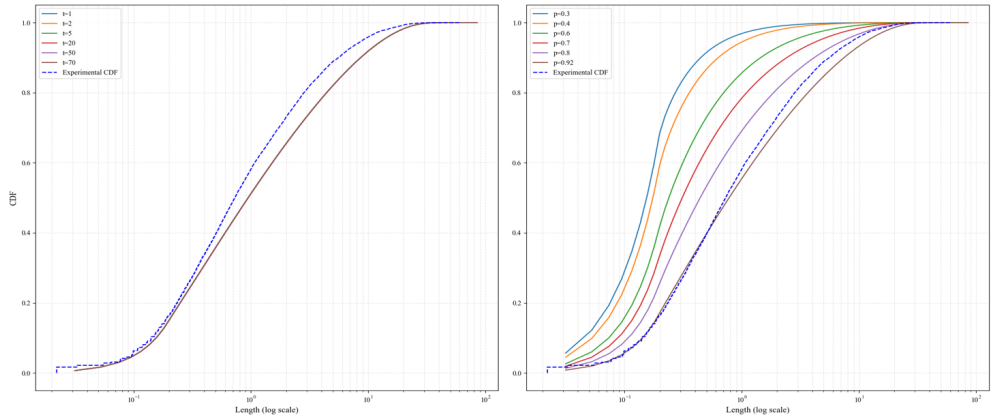


Figure 5.9: Experimental and theoretical CDFs for M0 design with 16k instances with varying t and p

The figure reveals that the shape of the CDF is primarily dependent on p and not on t. This is expected, as a normalized CDF only influences the shape of the distribution, while t affects the maximum value but not the functional form. This raises an important question: Is the extracted value of p incorrect, or is there an issue with the exponent $(2p - 4)$ in Davis' model?

Since p was extracted using multiple independent methods (the Growing Box Method and Recursive Splitting Method) and yielded consistent values, we can reasonably assume that our extracted p values are correct. On the other hand, in Davis' model, the total distribution is computed as the product of the structural distribution and the occupational probability. The structural distribution depends solely on the physical dimensions of the design and the total number of cells, making it independent of p. Therefore, any dependency on p must arise solely from the occupational probability, which is derived theoretically.

5.4.2 Refining the Model: Two-Part Optimization

Given these findings, the CDF modeling process was divided into two parts:

1. **Optimizing the Davis Model:** The first approach focuses on improving the Davis model by adjusting the exponent $(2p - 4)$ to better fit the empirical data for larger interconnects.

2. **Monte Carlo Simulations for Shorter Interconnects:** The Davis model is valid only for gate pitches of one and greater, meaning it does not accurately capture the behavior of very short interconnects. To address this limitation, Monte Carlo simulations will be employed to model short-range interconnect distributions more accurately.

These refinements will be explored in detail in the following sections.

5.5 Improving Davis Model – Longer Interconnects

For longer interconnects, the Davis model was improved by modifying the power law variable of length. After applying binomial expansion, the original model simplifies to the following equations:

$$\text{Region 1:} \quad 1 < \ell < \sqrt{N} \tag{5.1}$$

$$i(\ell) = \frac{\alpha k}{2}\Gamma \left(\frac{\ell^3}{3} - 2\sqrt{N}\ell^2 + 2N\ell \right) \ell^{2p-4} \tag{5.2}$$

$$\text{Region 2:} \quad \sqrt{N} \leq \ell < 2\sqrt{N} \tag{5.3}$$

$$i(\ell) = \frac{\alpha k}{6}\Gamma \left(2\sqrt{N} - \ell \right)^3 \ell^{2p-4} \tag{5.4}$$

Here, α is defined as $\frac{\text{fanout}}{\text{fanout}+1}$, k represents the number of pins, Γ is the normalization factor, N denotes the total number of gates, p is the Rent exponent, and ℓ is the net length.

5.5.1 Implementation and Theoretical CDF

The implementation was carried out in Python in a manner similar to the original Davis model. Additionally, a function was developed to compute the theoretical cumulative distribution function (CDF), which takes the net length and a new exponent (denoted as β in the implementation). To obtain the theoretical CDF, the built-in `quad` function was used to integrate the Davis probability density function (PDF).

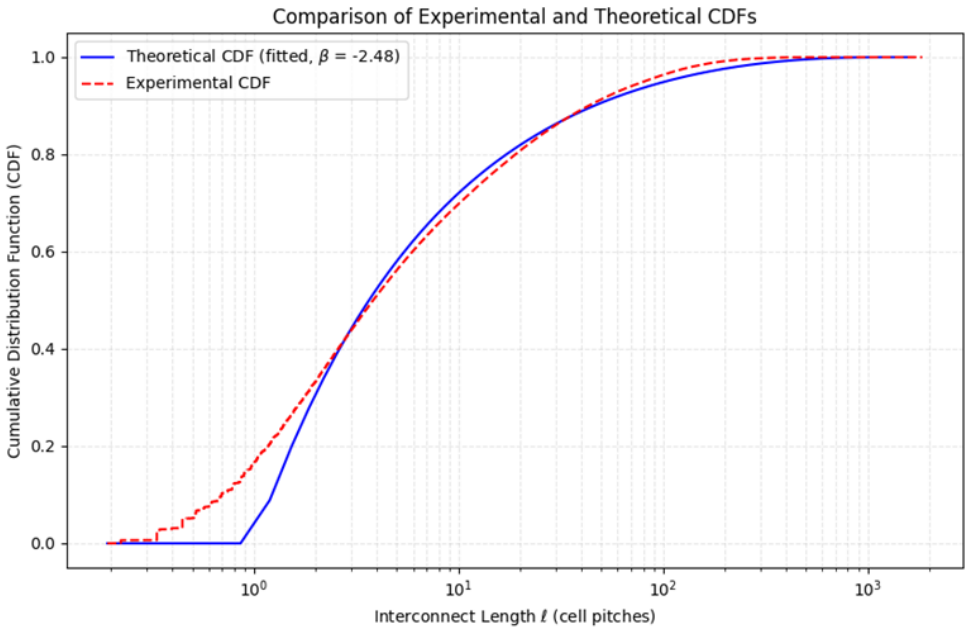


Figure 5.10: Improved Davis model CDF fit of a 64-bit arm core design with 700k instances

A wrapper function for `curve_fit` was also created, allowing for interpolation to match the size of the experimental data. Curve fitting was then performed by providing an initial guess for the β value, and the optimization process determined an optimal value for β.

As an example, Figure 5.10 shows the improved Davis CDF for the 64-bit ARM core design.

5.6 Variable Cell Size Model – Shorter Interconnects

As discussed earlier, all wire length distribution models presented thus far assume uniform cell spacing, identical cell sizes, and a single pin located at the center of each cell, as illustrated in Figure 5.11. However, real-world designs often feature variable core sizes (total area/total cell area), diverse cell dimensions, and non-uniform pin placements across the layout.

To better capture these variations, this section focuses on modeling the 64-bit ARM core design. In particular, the goal is to analyze short interconnects around gate pitch one using a Monte Carlo simulation. The simulation is based on the following assumptions:

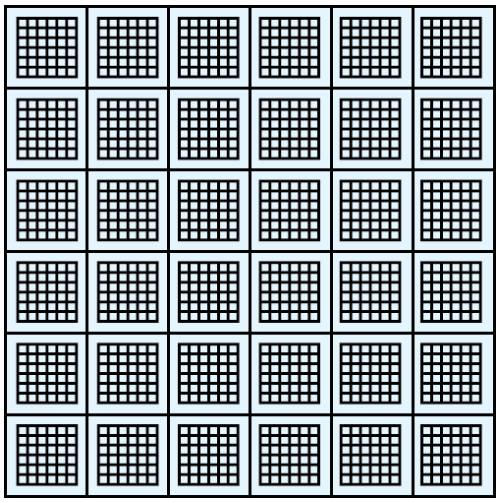


Figure 5.11: Equidistant Manhattan Model

1. The cell height is fixed at 0.09.
2. The cell width follows a uniform distribution among four possible values: W, $1.5W$, $2W$, and $2.5W$, where $W = 0.09$.
3. Pin locations are uniformly distributed within the cell (i.e., they can be placed anywhere inside the cell boundaries).
4. There is no vertical spacing between rows; the second row begins immediately after the first row ends.
5. Core utilization (set to 0.8 in this case) determines the total available spacing in the layout.
6. Horizontal spacing between adjacent cells is randomly partitioned among the four surrounding cells. For instance, if the total spacing is 1, then cell b_1 may receive a spacing $s_1 = 0.261$, cell b_2 a spacing $s_2 = 0.429$, and cell b_3 a spacing $s_3 = 0.310$, ensuring that all values sum to 1 in each iteration.
7. Each respective cell spacing is further randomly divided between the left and right sides of the cell. For example, if $s_1 = 0.261$, the left side of b_1 might have 0.157 spacing while the right side has 0.104 spacing, maintaining the sum of 0.261.

The experimental setup is depicted in Figure 5.12. The simulations were performed in Python, running for 10,000 iterations.

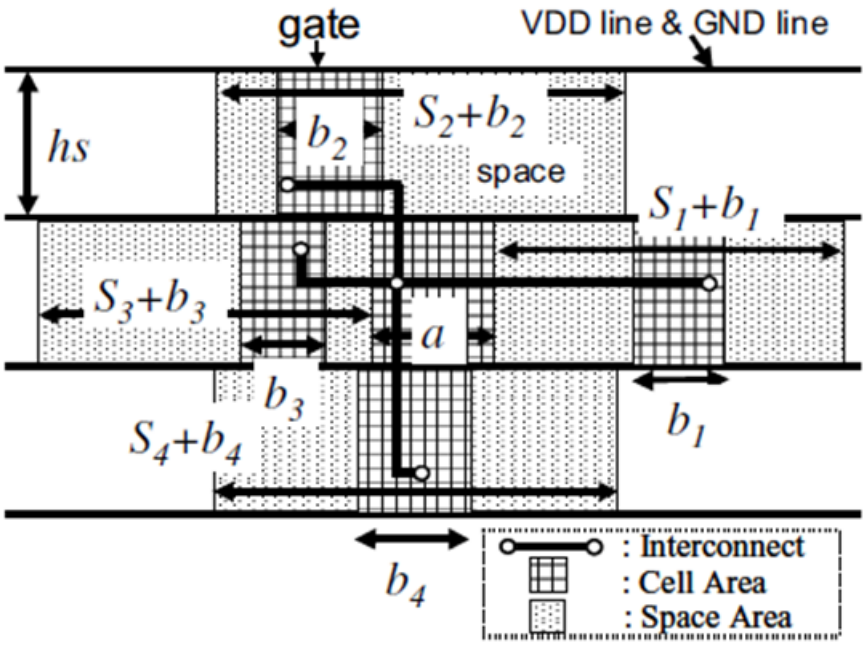


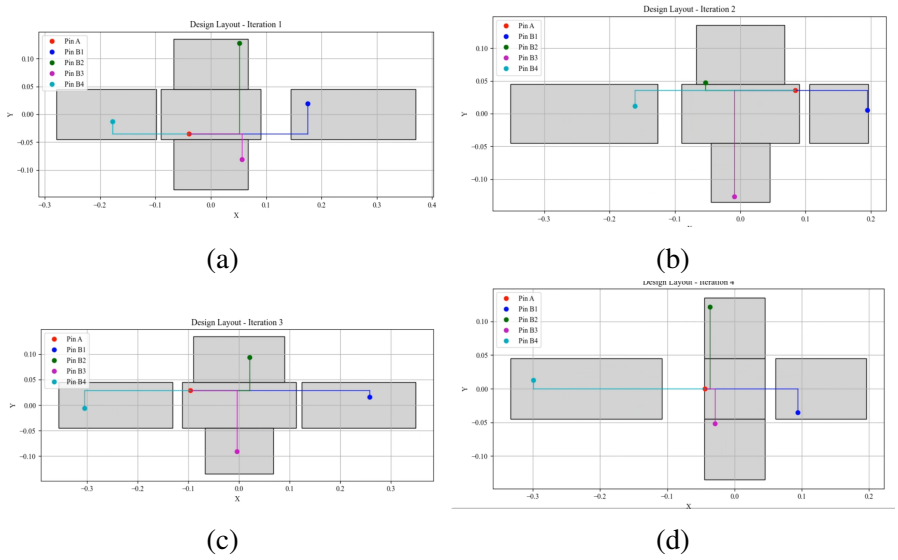
Figure 5.12: Proposed variable cell size model [20]

5.6.1 Monte Carlo Simulation

To model the variability in cell widths and pin placements, a Monte Carlo approach was adopted. The key steps in the implementation are as follows:

- A function was developed to assign random widths to cells, selecting from the predefined set of possible widths.
- The total layout width was computed, considering both the assigned cell widths and the additional spacing dictated by the core utilization factor.
- The total spacing was randomly distributed among the surrounding cells, ensuring that the sum remained consistent across iterations.
- Within each cell, pin locations were randomly generated using a uniform distribution to reflect realistic variability.
- Absolute pin positions in the layout were determined by accounting for both assigned cell widths and allocated spacing.
- Finally, Manhattan distances between the central cell pin and surrounding cell pins were calculated and stored for further analysis.

The simulation results are visualized in Figure 5.13, illustrating the spatial distribution of the first few iterations. The probability density function (PDF) of the setup, depicted in Figure 5.12, highlights the variable nature of gate pitch as opposed to the constant gate pitch assumption in older models.



(a) (b)

(c) (d)

Figure 5.13: First 4 Monte Carlo iterations of the proposed model

To further analyze the interconnect distribution, the cumulative distribution function (CDF) was computed. The CDF for net lengths up to gate pitch 2 is shown in Figure 5.15. The figure reveals that the probability distribution of shorter interconnects exhibits significant variability, reinforcing the importance of incorporating a variable cell size model in wire length predictions.

These results demonstrate that incorporating variable cell sizes into wire length modeling provides a more accurate representation of realistic circuit layouts. By changing the assumptions of cell height and cell width distribution, we can modify this model for other technology nodes. The next section will discuss further refinements and practical implications of this approach.

5.7 Combined CDF Model

In this section, the Modified Davis model for longer interconnects and the Variable Cell Size model for shorter interconnects are combined to create a unified wire length distribution model. The merging point for these two

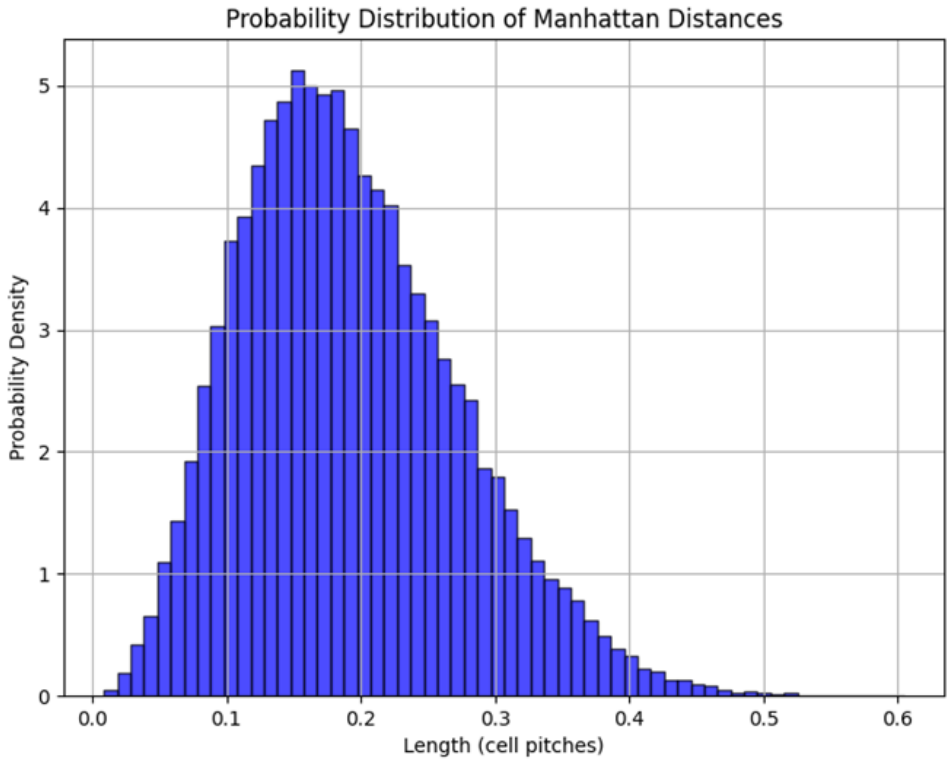


Figure 5.14: PDF of the proposed variable cell size model

models is set at gate pitch 2. To ensure a smooth transition between the two distributions, the cumulative distribution functions (CDFs) of both models are aligned at this point.

The implementation follows these steps:

- The CDF from the Variable Cell Size model is computed for interconnect lengths up to gate pitch 2.
- The CDF from the Modified Davis model is calculated for interconnect lengths beyond gate pitch 2.
- The transition point at gate pitch 2 is identified in both models.
- The CDFs are connected at this point, ensuring continuity and preserving the overall distribution shape.

By combining these models, a more comprehensive and accurate representation of interconnect length distribution is obtained. The fit between the two models was evaluated using the coefficient of determination (R^2),

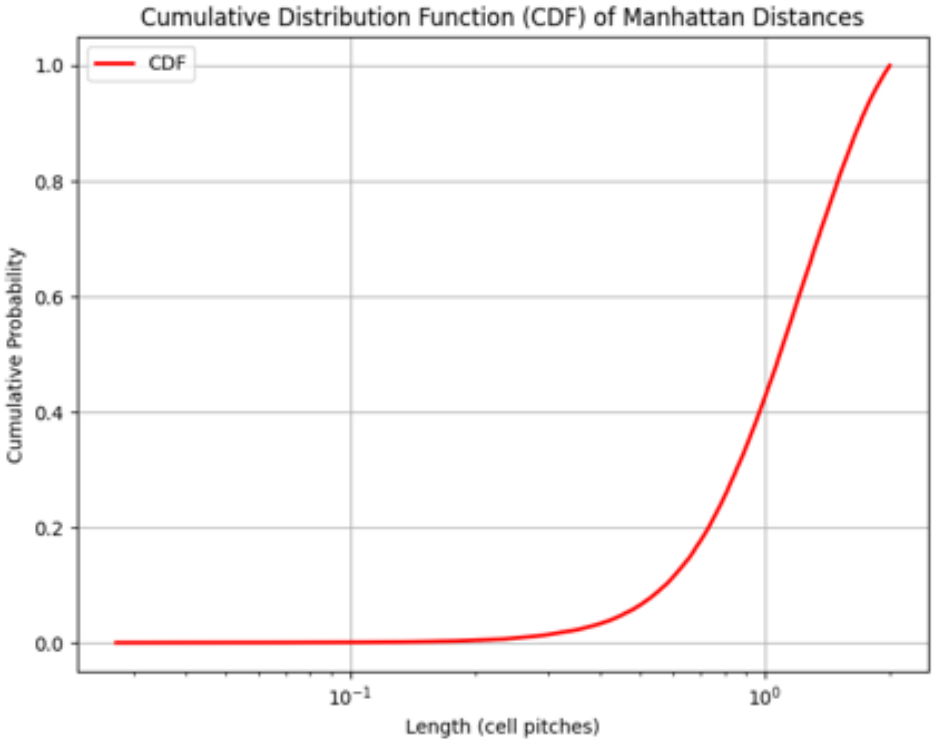


Figure 5.15: CDF of the proposed variable cell size model

which was found to be 0.98, indicating a strong correlation and a smooth connection between the short and long interconnect distributions.

For the 64-bit ARM core design, the combined CDF is shown in Figure 5.16. The figure demonstrates that the integration of the two models provides a seamless transition from short to long interconnects while maintaining accuracy.

This combined model provides a more realistic representation of wire length distributions in modern VLSI designs, as it incorporates both the variability in shorter interconnects due to cell size differences and the statistical behavior of longer interconnects modeled by the Modified Davis approach.

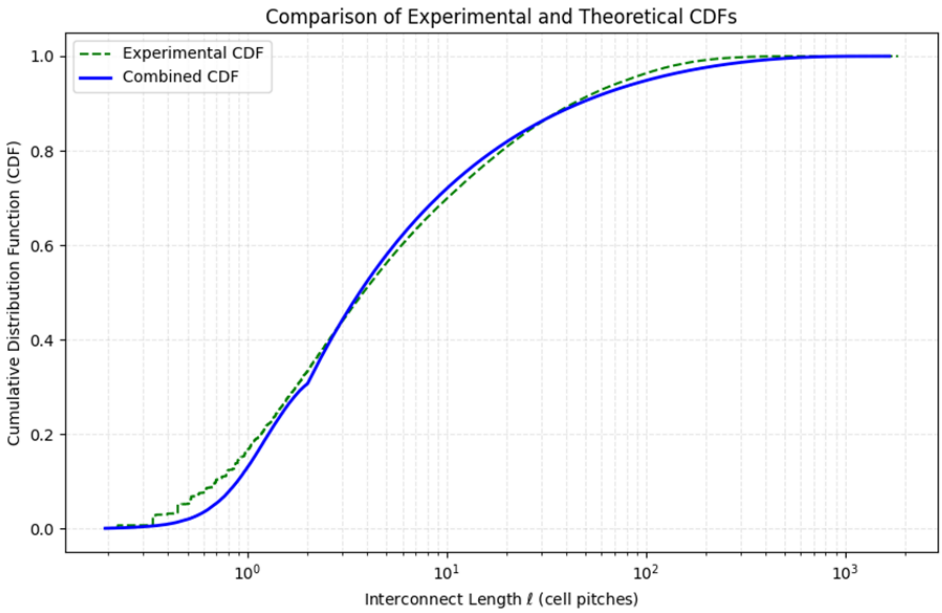


Figure 5.16: Combined CDF model fit of a 64 bit arm core design with 700k instances

Chapter 6

Discussion

In this chapter, the key findings of the research are analyzed, their implications are discussed, and potential limitations and future research directions are outlined. The discussion revisits the accuracy and applicability of the proposed model, its advantages over traditional methods, and its potential impact on early-stage design decisions in physical design. In addition, the assumptions, constraints, and trade-offs involved in the modeling process are critically examined.

6.1 Comparison of Rent's Coefficient Extraction Techniques

Two methods, the *Growing Box Method* and the *Recursive Splitting Method*, were employed to estimate the Rent exponent. Both methods are based on the same fundamental assumptions for counting the number of crossing nets and the number of cells within a bounded region. The extracted Rent exponent, as shown in Figures 4.1 and 4.4, remains nearly identical between the two methods, differing only in the second decimal place. This similarity confirms the robustness of the extraction methodology irrespective of the chosen approach.

Despite producing comparable results, these methods differ significantly in implementation and computational efficiency. The Growing Box Method is fully implemented in TCL, making it independent of external scripting languages like Python. However, this comes at a cost—its computation time is substantially higher compared to the Recursive Splitting Method. In contrast, the Recursive Splitting Method leverages Python for loading the design data

and performing computations, leading to a significant reduction in execution time. The computation times for both methods are presented in Tables 4.1 and 4.2.

One major drawback of the Growing Box Method is its inherently sequential approach, where a single box expands from the center outward. This process could be optimized by introducing multiple growing boxes that originate from different corners of the chip and expand towards the center or opposing edges. Such a modification would not only reduce computation time but also increase the number of data points for regions containing a high density of cells and nets. By covering these regions multiple times, the extraction process would yield a more accurate representation of Rent's coefficient distribution.

Moreover, the Recursive Splitting Method, while computationally efficient, inherently partitions the design in a hierarchical manner, which may introduce minor inaccuracies in highly irregular placements. Further optimizations, such as dynamic binning strategies or parallel processing, could further enhance both accuracy and speed.

6.2 Limitations of Traditional Wire Length Estimation Methods

In the past, classical models such as Donath's and Davis' methods have been widely used to predict interconnect distributions. However, when applied to modern ARM-based designs, these models exhibit significant discrepancies from actual wire length distributions, highlighting their limitations in the context of advanced CMOS nodes.

6.2.1 Discrepancies in Donath's Model

The implementation of Donath's model for the analyzed ARM designs revealed a major mismatch between the theoretical and actual wire length distributions. As shown in Figure 5.4 , there is no significant overlap between the predicted and observed distributions, with the slopes differing significantly. More critically, the implemented model did not adhere to the theoretical relation proposed by Donath, where the slope γ of the wire length distribution is given by:

$$\gamma = 3 - 2p \tag{6.1}$$

This deviation suggests several potential sources of error. One key factor is that Donath's model assumes a hierarchical placement of cells from a netlist, iteratively cutting them into smaller modules. However, in this study, wire length estimation was performed on an already placed design, where the placement was dictated by a PnR flow rather than a hierarchical partitioning approach. This fundamental difference likely led to uneven module sizes across hierarchical levels, which could have distorted the distribution.

In hierarchical placement, the lowest-level module ideally contains a single cell, minimizing the number of nets being cut into subnets. In contrast, PnR prioritizes minimizing wire lengths and optimizing routing efficiency, which may alter the subnet distribution. Additionally, it is possible that Donath's probability density function is outdated or less accurate for newer designs with modern routing methodologies. The observed mismatch could be a result of one or a combination of these factors, making Donath's model less suitable for contemporary CMOS designs.

6.2.2 Inaccuracies in Davis' Model

Davis' stochastic wire length model showed better agreement with actual distributions for longer interconnects, but significant deviations were observed for shorter interconnects. In certain regions, the predicted number of shorter interconnects was off by a factor of 10^2, leading to a substantial overestimation of total net counts. This suggests that the underlying assumptions about short interconnect distribution do not align with modern physical design practices.

One possible explanation is that Davis' model assumes a linearly increasing probability distribution (on a log scale) for interconnects at shorter distances, whereas real designs are constrained by placement density, routing congestion, and pin access limitations, which alter the actual interconnect distribution. The model inherently predicts a much higher number of short interconnects than observed, suggesting that adjustments to the probability function are necessary. An additional challenge affecting the accuracy of wire length estimation is the binning problem, which is further elaborated in the next section, as it plays a crucial role in the observed discrepancies between theoretical predictions and experimental results.

6.3 The Binning Problem

In the literature on wire length distribution modeling, probability density functions (PDFs) are the standard approach. From Donath to Davis, most

implementations focus on modeling wire length distributions through PDFs. However, a critical issue arises—there is little to no discussion on how binning is performed in these models. Key questions such as whether the binning is linear or logarithmic, how bin sizes are chosen, and what effect different binning strategies have on the results remain unanswered.

This lack of clarity is problematic because theoretical models have no dependency on experimental binning, whereas empirical data must be processed through a binning scheme to generate a PDF. This discrepancy introduces a potential source of error when comparing theoretical predictions with real-world data. One possible way to address this issue would be to extract the structural distribution experimentally, as demonstrated in the subsection 5.3.2.1, and separately extract the probability distribution from the physical architecture. However, there is currently no known method for experimentally deriving the probability distribution. This is because the probability distribution is theoretically derived based on the *law of conservation of terminals* [5] and Rent's Rule, rather than being directly observable in the structure. As a result, the validity of the occupation probability function for new designs and process nodes remains uncertain.

Another major concern is that the choice of binning parameters—such as the number of bins and their scaling—can significantly impact the computed slope and the maximum number of nets in the PDF. In extreme cases, a poorly fitting model could be artificially corrected by adjusting the binning scheme to match the expected slope. This raises concerns about the reliability of PDF-based methods. As an example, Figure 6.1 shows PDFs computed for the ARM M0 design using different numbers of bins. The substantial variations in slope highlight the sensitivity of results to binning choices.

To mitigate this issue, the focus was shifted from PDFs to cumulative distribution functions (CDFs). Unlike PDFs, CDFs do not require binning, making them a more robust alternative for comparing experimental and theoretical results. Starting from the next section, all wire length distribution models will be analyzed using CDFs instead of PDFs.

6.4 Evaluation of the Proposed Wire Length Model

The proposed wire length model introduces a two-part optimization to improve the accuracy of interconnect length distribution predictions for advanced CMOS nodes. The first approach refines Davis' model by modifying the

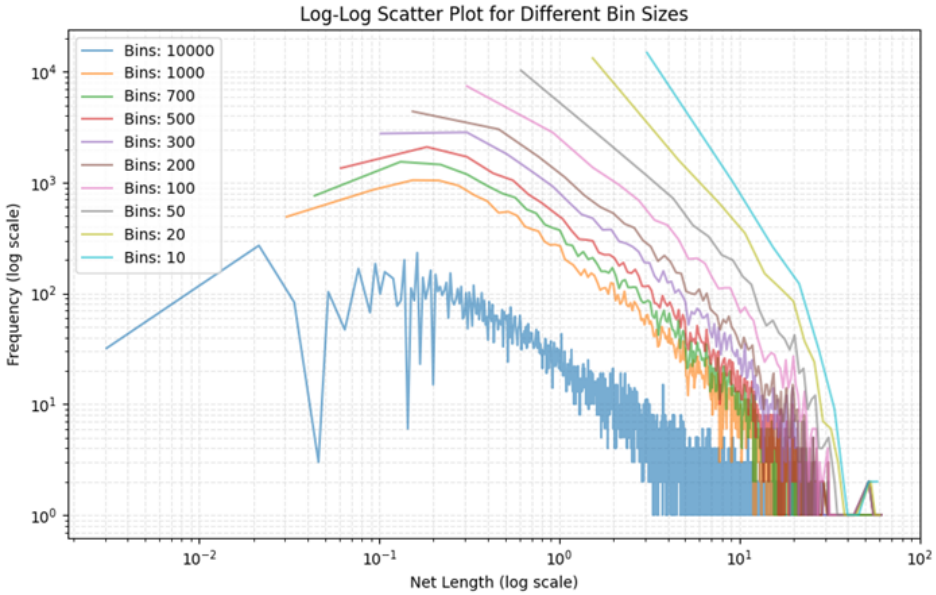


Figure 6.1: Net count vs length for M0 design with 16k instances and multiple number of bins

exponent to better fit empirical data for longer interconnects. The second approach, namely *Variable Cell Size Model* addresses the shortcomings of Davis' model for very short interconnects, where it is no longer valid, by employing Monte Carlo simulations to capture short-range interconnect distributions more accurately.

6.4.1 Performance of the Improved Davis Model

For longer interconnects, the adjusted exponent significantly improved the agreement between the theoretical and observed distributions. As shown in the Figure 5.10, the fitted model performs exceptionally well for cell pitches greater than 2, aligning closely with the empirical data. However, a fundamental limitation remains—Davis' model is inherently valid only for cell pitches greater than 1.

One primary reason for this limitation is the set of assumptions underlying Davis' model. It assumes a rigid Manhattan grid where all cells are equidistant, have uniform height and width, and always have pins located at their centers. In practical applications, these assumptions do not hold, as real-world physical designs exhibit variability in cell dimensions, placement density, and pin locations. This discrepancy leads to deviations between

the theoretical predictions and actual wire length distributions. Despite this limitation, the optimized model provides a significant improvement over the original formulation.

6.4.2 Effectiveness of Variable Cell Size Model

To address the inaccuracy for short interconnects, a Monte Carlo simulation was implemented to model short-range interconnect distributions more precisely. Figure 5.16 indicates that this method significantly improved the overall fit in the short interconnect region. However, visible discrepancies in this region suggest that certain assumptions in the Monte Carlo setup may have been overly idealized.

One possible explanation for this deviation is the assumed distributions of cell widths and pin locations. While these assumptions were reasonable approximations, it is possible that an unaccounted-for variable or an overlooked distribution parameter significantly influences short-range interconnect behavior. Additionally, increasing the number of Monte Carlo iterations beyond 10,000 did not yield noticeable improvements in the distribution, suggesting that the limiting factor lies in the model assumptions rather than statistical noise.

6.4.3 Overall Model Fit and Potential Refinements

The combined model, incorporating both the improved Davis model for long interconnects and the Monte Carlo simulation for short interconnects, demonstrated a high degree of accuracy. The final fit achieved an R^2 value of 0.984, indicating strong correlation with empirical data. However, one noticeable issue was the transition point between the short and long interconnect models. A visible interchange exists where the Monte Carlo-based distribution merges with the Davis-based model at cell pitch 2.

One potential refinement could involve applying a smoothing function to seamlessly blend the two models and eliminate the discontinuity. However, such an adjustment may compromise the validity of the model, as it would introduce an artificial modification rather than being derived from first principles.

Chapter 7

Conclusions and Future Work

This chapter summarizes the key findings, sheds light on key insights gained, and outlines potential future directions.

7.1 Conclusions

The results demonstrated that traditional wire length estimation models, such as those proposed by Donath and Davis, exhibit significant discrepancies when applied to modern physical designs. Donath's model, in particular, showed no overlap between theoretical and observed distributions, largely due to differences in placement methodology—hierarchical placement versus physical design-optimized placement. Davis' model performed better but was limited to interconnects with lengths greater than one gate pitch, leading to substantial deviations in the short interconnect region.

The results showed that the proposed model significantly improved wire length estimation accuracy, achieving an R^2 value of 0.984. The model successfully captured the trends observed in real designs and corrected for the systematic errors seen in classical methods. However, minor discrepancies remained, particularly in the transition between the short and long interconnect models. A possible refinement could involve applying a smoothing function to ensure a seamless transition, but this would need to be done cautiously to preserve the model's validity.

7.2 Insights and Lessons Learned

Throughout this research, several key insights were gained:

- Rent's rule remains a powerful tool for wire length estimation, but the way Rent's exponent is extracted significantly impacts model accuracy. The growing box and recursive splitting methods provided similar results, but the recursive approach was computationally more efficient.
- The assumptions in classical models, such as uniform cell size, equidistant placement, and centered pin locations, do not hold in advanced CMOS nodes. This highlights the need for updated modeling approaches that account for placement density, variable cell sizes, and realistic pin distributions.
- Binning in PDFs introduces significant variability in the estimated wire length distributions. Shifting to CDFs eliminated this issue, leading to a more stable and reliable model.
- The Monte Carlo approach effectively captured short interconnect behavior, but further refinements are needed to address minor discrepancies, potentially by refining the assumed distributions of cell widths and pin locations.

7.3 Future Work

While this research significantly improves wire length estimation for advanced CMOS nodes, several avenues remain open for future work.

7.3.1 Developing a Purely Theoretical Wire Length Estimation Model

One of the major limitations of this study is that it relies on post-placement data, making it a posteriori in nature. A significant step forward would be to develop a purely theoretical, *a priori* wire length estimation model. This could be achieved by starting from the circuit graph instead of the post-placement data. Partitioning techniques for the circuit, like `ratio cut` [21] and `hMeTis` [22] can be explored for this purpose.

7.3.2 Refining Short Interconnect Modeling

While the Monte Carlo simulation improved accuracy in the short interconnect region, further refinements are needed. Future research could explore:

- More realistic cell width and pin location distributions based on empirical data from real cell libraries.
- Machine learning-based approaches to predict short interconnect distributions more efficiently.

7.3.3 Improving the Transition Between Short and Long Interconnect Models

The current model shows a visible transition point between the short and long interconnect models at a gate pitch of 2. Future work could:

- Explore smoothing techniques to ensure a seamless transition while preserving model accuracy.
- Investigate hybrid statistical models that blend Monte Carlo simulations with analytical models more smoothly.

7.3.4 Generalization to Other Architectures

This study was conducted on ARM-based designs, but the methodology could be extended to:

- Other processor architectures, such as RISC-V or custom ASIC designs.
- Different technology nodes.
- Heterogeneous integration scenarios.

7.4 Final Remarks

This research successfully enhanced wire length estimation for advanced CMOS nodes by addressing the limitations of classical models and incorporating empirical refinements. The proposed hybrid approach provides a more accurate representation of interconnect distributions, with strong validation against real-world ARM-based designs. While there are still areas for improvement, the insights gained lay the foundation for further advancements in theoretical interconnect modeling, bridging the gap between early-stage estimation and modern physical design constraints. Future work in this area could lead to more predictive and efficient design methodologies, ultimately benefiting the semiconductor industry as it continues to push the boundaries of scaling and integration.

References

[1] M. Pedram and B. Preas, "Accurate prediction of physical design characteristics for random logic," in *Proceedings of the 1989 IEEE International Conference on Computer Design: VLSI in Computers & Processors*. IEEE, 1989. doi: 10.1109/ICCD.1989.167415 pp. 100–108. [Page 2.]

[2] B. Landman and R. Russo, "On a pin versus block relationship for partitions of logic graphs," *IEEE Transactions on Computers*, vol. C-20, no. 12, pp. 1469–1479, 1971. doi: 10.1109/T-C.1971.223159 [Pages 3 and 15.]

[3] W. Donath, "Placement and average interconnection lengths of computer logic," *IEEE Transactions on Circuits and Systems*, vol. 26, no. 4, pp. 272–277, 1979. doi: 10.1109/TCS.1979.1084635 [Pages 3, 15, and 20.]

[4] W. E. Donath, "Wire length distribution for placements of computer logic," *IBM J. Res. Dev.*, vol. 25, no. 2–3, p. 152–155, Mar. 1981. doi: 10.1147/rd.252.0152. [Online]. Available: https://doi.org/10.1147/rd.252.0152 [Pages 3, 15, and 20.]

[5] J. Davis, V. De, and J. Meindl, "A stochastic wire-length distribution for gigascale integration (gsi). i. derivation and validation," *IEEE Transactions on Electron Devices*, vol. 45, no. 3, pp. 580–589, 1998. doi: 10.1109/16.661219 [Pages 3, 23, 25, and 54.]

[6] D. Stroobandt and H. Van Marck, "Efficient representation of interconnection length distributions using generating polynomials," in *Proceedings of the 2000 International Workshop on System-Level Interconnect Prediction*, ser. SLIP '00. New York, NY, USA: Association for Computing Machinery, 2000. doi: 10.1145/333032.333039. ISBN 1581132492 p. 99–105. [Online]. Available: https://doi.org/10.1145/333032.333039 [Page 3.]

[7] D. Prasad, S. Sinha, B. Cline, S. Moore, and A. Naeemi, "Accurate processor-level wirelength distribution model for technology pathfinding using a modernized interpretation of rent's rule," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, 2018. doi: 10.1109/DAC.2018.8465803 pp. 1–6. [Pages 3 and 30.]

[8] D. Yakimets, M. G. Bardon, D. Jang, P. Schuddinck, Y. Sherazi, P. Weckx, K. Miyaguchi, B. Parvais, P. Raghavan, A. Spessot, D. Verkest, and A. Mocuta, "Power aware finfet and lateral nanosheet fet targeting for 3nm cmos technology," in *2017 IEEE International Electron Devices Meeting (IEDM)*, 2017. doi: 10.1109/IEDM.2017.8268429 pp. 20.4.1–20.4.4. [Page 4.]

[9] A. Farokhnejad, S. S. Sahoo, D. Abdi, J. Cousins, A. Dutta, G. Mirabelli, V. Sankatali, L. Verschueren, S. Yang, O. Zografos, J. Myers, P. Weckx, M. G. Bardon, G. Hellings, and J. Ryckaert, "N2 nanosheet pathfinding-pdk (p-pdktm) including back-side pdn," in *2024 IEEE European Solid-State Electronics Research Conference (ESSERC)*, 2024. doi: 10.1109/ESSERC62670.2024.10719536 pp. 17–20. [Page 4.]

[10] H. Kükner, G. Mirabelli, S. Yang, L. Verschueren, J. Bömmels, J. Y. Lin, D. Abdi, A. Farokhnejad, A. Zografos, N. Horiguchi, M. G. Bardon, G. Hellings, and J. Ryckaert, "Double-row cfet: Design technology co-optimization for area efficient a7 technology node," in *2024 IEEE International Electron Devices Meeting (IEDM)*, 2024. doi: 10.1109/IEDM50854.2024.10873524 pp. 1–4. [Page 4.]

[11] N. Sherwani, *Algorithms for VLSI Physical Design Automation*. Springer, 1995. [Page 7.]

[12] D. Gajski, "Guest editors' introduction: New vlsi tools," *Computer*, vol. 16, pp. 11–14, 1983. [Online]. Available: https://api.semanticscholar.org/CorpusID:16669482 [Page 8.]

[13] H. Veendrick, "Very-large-scale integration (vlsi) and asics," in *Nanometer CMOS ICs: From Basics to ASICs*, latest ed. Berlin, Heidelberg: Springer, 2025, pp. 349–409. ISBN 978-3031297458 [Page 8.]

[14] D. Stroobandt and J. V. Campenhout, "Accurate interconnection length estimations for predictions early in the design cycle," *VLSI Design*, vol. 10, no. 1, p. 081698, 1999. doi: https://doi.org/10.1155/1999/81698.

[Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/1999/81698 [Page 11.]

[15] L. Hagen, A. Kahng, F. Kurdahi, and C. Ramachandran, "On the intrinsic rent parameter and spectra-based partitioning methodologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 1, pp. 27–37, 1994. doi: 10.1109/43.273752 [Page 17.]

[16] H. M. Ozaktas, "Paradigms of connectivity for computer circuits and networks," *Optical Engineering*, vol. 31, no. 7, pp. 1563 – 1567, 1992. doi: 10.1117/12.57685. [Online]. Available: https://doi.org/10.1117/12.57685 [Page 17.]

[17] D. Stroobandt, "Interconnect research influenced," *IEEE Solid-State Circuits Magazine*, vol. 2, no. 1, pp. 21–27, 2010. doi: 10.1109/MSSC.2009.935293 [Page 21.]

[18] J. Cotter and P. Christie, "The analytical form of the length distribution function for computer interconnection," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 3, pp. 317–320, 1991. doi: 10.1109/31.101325 [Pages 21, 22, and 24.]

[19] D. Stroobandt, "A priori system-level interconnect prediction: Rent's rule and wire length distribution models," in *Proceedings of the 2001 International Workshop on System-Level Interconnect Prediction*, ser. SLIP '01. New York, NY, USA: Association for Computing Machinery, 2001. doi: 10.1145/368640.368645. ISBN 1581133154 p. 3–21. [Online]. Available: https://doi.org/10.1145/368640.368645 [Page 25.]

[20] H. Nakashima, J. Inoue, K. Okada, and K. Masu, "Ulsi interconnect length distribution model considering core utilization," in *Proceedings of the Conference on Design, Automation and Test in Europe - Volume 2*, ser. DATE '04. USA: IEEE Computer Society, 2004. ISBN 0769520855 p. 21210. [Page 46.]

[21] Y.-C. Wei and C.-K. Cheng, "Ratio cut partitioning for hierarchical designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 7, pp. 911–921, 1991. doi: 10.1109/43.87601 [Page 58.]

[22] G. Karypis and V. Kumar, “Hmetis: a hypergraph partitioning package,” *ACM Transactions on Architecture and Code Optimization*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:59699713 [Page 58.]

TRITA – EECS-EX-2025:92
Stockholm, Sverige 2025

www.kth.se