Universitat Politècnica de Catalunya
Departament de Llenguatges i Sistemes Informàtics
Màster en Computació

Tesi de Màster

Multi-clustering net model for VLSI placement

Estudiant: Andrey Ziyatdinov
Director: Jordi Cortadella

Data: September 7, 2008

Preface

This master's thesis falls into the field of computer science applied on microelectronics. The design process of Very-Large-Scale Integration (VLSI) circuits circuits is challenged by exponentially increasing integration densities and shrinking characteristic geometries on a chip. The wires, rather than devices, become the dominant factor in deciding the performance, power consumption, and reliabilities of VLSI systems.

Placement and routing are two steps that produce the physical layout based on the netlist information, and determine the performance of the circuit in terms of the length of wires. The study of this thesis is concerned about net models on wirelength estimation employed in placement. The final routed wirelength in the later routing phase is needed to be efficiently approximated in earlier placement phase. The accuracy and computational complexity of net models are factors to be considered in this work.

From computer science point of view, the minimum amount of wires required to interconnect n pins of a net is computed by construction of minimum Steiner tree which is a combination of two polynomial-time graph algorithms: shortest path and minimum spanning tree. However, the Steiner tree problem is NP-complete, and thus optimization of Steiner tree wirelength (StWL) cost regarded to be unpractical in placement.

The traditional approach is to employ Half-Perimeter Wirelength (HPWL) heuristic which estimates the netlength as half-perimeter of bounding box enclosing all the pins. One can easily see that HPWL gives the exact estimation for 2-pin and 3-pin nets, but the error in the case of multi-pin nets can be significant. The motivation to design a new net net model comes from necessity to improve the HPWL net model for multi-pin nets.

A novel clustering approach to the problem of netlegnth modeling is proposed in this thesis. A net is split into several subnets and the total HPWL of the subnets presents the wirelength. Clustering idea supposed supposed to work, because the HPWL measure is applied to subnets with smaller pincount than the original "unbroken" net. Moreover, the pins are clustered according to the density of their positions and resulted clusters assumed to go along contours of Steiner tree. The accuracy of the net model has been proved empirically and has shown the superior results in comparison with HPWL.

In terms of computational time, a effective multi-clustering algorithm is proposed for breaking the net into subnets. One of the main contributions of the thesis concludes in linear algorithmic complexity on the number of pins. The implementation of well-known k-means clustering approach is combined with local search on the optimal number k of clusters.

In experiments, the clustering algorithm has been used to build a new netlist where each original multi-pin net is substituted by subnets obtained by clustering. Since placement is typically proceeded in two global and detailed steps, the new netlist is constructed after global placement and passed to detailed placement. This experimental scheme has a practical advantages for physical design

community; our net model can be easily integrated and tested in any placement framework regardless of internal implementation of the placer.

The experiments have been run on the most recent circuit benchmarks containing up to 100 million of components and for three different type of placers. Obtained results have shown that our implementation of StWL cost outperforms the traditional HPWL-based approach in reduction of both wirelength and wire delays, with no or little additional CPU time. Thus, this thesis gives a positive answer to the key question in VLSI placement whether it is worth to replace the common HPWL measure by Steiner tree in netlength modeling.

Acknowledgements

I would like to thank Prof. Jordi Cortadella to the first order for invitation to join this project and guide me during all my study in Universitat Politècnica de Catalunya.

I must thank David Bañeres and Jordi Cortadella for their invaluable assistance in the work on our joint paper which will be published in October 2008.

Financial support for this work comes from CICYT TIN2007-66523, the UPC research scholarship and the grant from Intel Corporation. I gratefully acknowledge the contribution of these organizations.

Andrey Ziyatdinov

Abbreviations and Acronyms

VLSI	Very-Large-Scale Integration
CAD	Computer Aided Design
MST	Minimum Spanning Tree
SMT	Minimum Steiner Tree
RSMT	Rectilinear Steiner Minimum Tree
HPWL	Half-Perimeter Wirelength
StWL	Steiner tree wirelength
rWL	routed Wirelength
MCN	Multi-Clustering Net (model)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter starts with a short overview of the high-speed microelectronics and introduces some basic concepts of the design process in Section 1.1. Special attention is given to physical design phase which is presented in Section 1.2.

The research interest of the thesis is related with models for estimation the wirelength needed to interconnect circuit components on the physical layout. Such models applied in different levels of physical synthesis are briefly described in Section 1.3, that will be required to introduce the statement of the problem in Section 1.4.

1.1 Microelectronics

The phenomenal achievements have been happened in electronics over the past three decades, mainly because of the advent of Very-Large-Scale Integration (VLSI). The VLSI technology came with Computer Aided Design (CAD) that has been enabling the constant growth in the complexity and performance of integrated circuits. Many new applications and innovations powered by transistors continue coming in our daily lives and introduce new opportunities in high-performance computing, telecommunications and consumer electronics.

The number of circuit components in a chip has been rising according to Moore's Law which states doubling roughly every 18 months. Now days Intel Corp. enters billion-transistor era, releasing the "Penryn" processor on 45 nm technology with 820 million transistors [8].

To cope with the rapid and steady increase in circuit complexity, the *abstraction hierarchy* is traditionally used to split VLSI design cycle into separate tasks. The current design flow typically proceeds in the following sequence.

Behavioral level design creates the functional specification models of integrated circuit in terms of input, output and timing of each block, without describing its internal implementation. The area, power and other parameters are assigned to each block and should be maintained further. Such

high level of abstraction allows to employ sophisticated data and control representations for inspection the correct functioning of the circuit.

Logic design converts the behavioral specification into a register transfer level (RTL) description including the control flow, word widths, register allocation, arithmetic and logic operations. The logic design is simulated and tested with the goal of minimization on the number of Boolean expressions.

Circuit design transforms the logic expressions into a circuit representation with components like cells, macros, gates, transistors, and interconnections along them collected in a netlist. Circuit simulation is used to verify the correctness and timing of each component.

Physical design generates geometrical layout of the chip by representing circuit elements as rectangular shapes, and interconnections along them - as wires in multiple metal layers. Compact arrangement of area and accurate routing of wires evaluate the final performance of the circuit.

Therefore, the VLSI design process can be represented as transformation of data, where a VLSI chip can be viewed behaviorally as a system of functional restrictions, structurally as a family of logic gates, or physically as a population of rectangular cells interconnected by wires. However, the success of this approach strongly depends on correlation between abstract models at the higher level and physical implementation at the lower level.

In fact, today's nanometer-scale silicon complexity makes existing abstractions at earlier design stages largely incapable of simulating the performance, complexity and reliability of the interconnect. Typically, the hierarchal structure of the design flow is extended by *verification design stages*, and many iterations over the flow sequence are required to meet the design constraints. The principal objective of VLSI CAD tools is to minimize both the time of each iteration and the total number of iterations, thus reducing time-to-market.

Another critical factor in VLSI design process comes from the interconnect-to-device delay ratio, which is expressed in dominance the interconnect delay over the device delay. Thus, the VLSI system requires more time to send data from one chip component to another than to produce the data by devices. This ratio grows ever more problematic as design sizes increase. Due to rising interconnect delay, the *early physical design* starts beforehand in the design cycle, in order to get improved estimates of the performance of the circuit. As a result, high level representations of the design receive a feedback from physical implementation and identify some potential layout problems.

In conclusion, it is important to stress the crucial role of physical synthesis in the current deep submicron design. The performance of the circuit can be evaluated completely only in the last design phase, when circuit components are placed on the layout and interconnects along them are routed as wires. Additionally, reliable estimation of wire path and other physical effects definitely assist to fill the gap between physical implementations and abstract models in early design steps. In the future, such integration of physical design assumed to prevent

iterations of the design steps by replacing verification procedures in different design levels [9].

The overall scheme of the VLSI design cycle within verification and early physical design steps is presented in Appendix A. The physical design phase is described in the next section.

1.2 Physical Design

Following logic synthesis and circuit design, circuit components are extracted from a physical library and transformed into specific rectangular shapes of fixed dimensions. These components are referred to as modules or cells, and interconnections between them - as nets. The nets are collected in a netlist, and timing constraints on signal prorogation paths along nets are also specified. The output of the physical design stage is the layout of the circuit, where all the cells are positioned on the chip without overlapping and all the interconnection paths are completed.

Once the layout has been produced, one can ascertain the speed of the chip, its power consumption and other performance characteristics. Let us see how the performance constraints are formulated geometrically in physical design.

According to geometrical abstraction, circuit components are associated with rectangular shapes regardless logic function intended inside. Instead, a critical quantity controlling the performance is the amount of wire needed to interconnect cells. Timing delay over the signal path regarded to be strongly consistent with the path length: the shorter path the less time required for signal prorogation. Therefore, *the total wirelength* serves as a primary cost function in physical design that determines the performance speed of a chip.

However, the layout with short wirelength may have some local areas with high congestion of wires. Due to very high utilization ratio in many integrated circuits, congestion may happen in many areas of the chip, especially in areas with high connectivity of cells. Thus, *congestion* appeared to be another typical cost in physical design. Wirelength and congestion costs are dependent and should be improved concurrently in design flow.

The important characteristic of the layout is *routability*, that means the ability to pass wires along available routing channels. Accordingly, congestion considered to be its direct measure, whereas wirelength - a consistent approximation. The task of producing routable circuit with short wirelength is very complex and computationally demanding, and traditionally broken into two steps. [1]

Placement determines the positions of cells on the chip with minimum area arrangement and the shortest *estimated wirelength*. Due to the vast space of possible solutions, simple models of wirelength estimation are employed

[1] Indeed,integrated circuit is firstly partitioned into sub-circuits, referred to as modules. Then Floorplanning assigns the modules along the layout with optimal arrangement in terms of area and interconnections. The next two steps Placement and Routing complete the layout for each module.

to save runtime. Additionally, the measure of routability is selected as the total wirelength of interconnects, although some congestion-aware techniques are used in modern placers.

Routing receives the layout of cells with assigned positions that can not be changed. The objective of routing is to complete the interconnections between cells according to their positions and free available space by using *the shortest possible wirelength* measure. Routers take into account both the total wirelength for performance optimization and congestion for construction the physical path of wires on the layout.

The details of placement and routing algorithms will be given in the next chapter, while the remainder of this section is addressed to the problem of coherence between placement and routing.

Placement algorithms can afford to employ only estimation techniques of both wirelength and congestion because of the huge number of objects to be positioned from scratch. Contrary to placers, routers utilize exact algorithms to construct the shortest path of wires and evaluate congestion constructively.

For simplicity sake, placement can be viewed as a preparatory step for interconnect optimization performed in routing phase. Contrary to routers, placers dot not take into account such physical effects in interconnection path as detours of wires due to congestion or the number of bends, but employ abstract models for estimation the wirelength. The criteria to accuracy of these models is restricted by consistency with the routed wirelength.

The estimation of the routing cost is crucial during placement. On one hand, it is desirable to use models that provide an accurate estimation of the final wirelength. On the other hand, the models must be simple enough for the algorithms to have a manageable computational complexity. This trade-off is a continuous area of research in physical synthesis.

The thesis is dedicated to the problem of minimizing the total routed wirelength which is one of the fundamental goals in the VLSI placement stage. The next section introduces basic wirelength models implied during different phases of physical design. After defining the *net model* in terms of graphs, the description will be focused on accuracy and computational complexity of the models.

1.3 Wirelength Net Models

From computer science point of view, algorithms of physical design operates with connectivity graph based on the netlist information and project the graph onto 2D space satisfying predefined interconnect constraints. Each node of the graph a rectangular cell with width and height on the die. Edges of the graph have a hyperedge structure and express a interconnection a certain number of cells, referred to as terminals or pins. The number of pins corresponds to the net degree. Therefore, a *net model* is responsible to interconnect all the pins of the net using *minimum amount of wire*, which corresponds to graph problems including minimum spanning tree and minimum steiner tree.

Minimum Spanning Tree (MST) Given an edge-weighted graph $G = \{V, E\}$, select a subset of edges $E' \subseteq E$, such that E induces a tree and the total cost of edges is minimum over all such trees [4]. The weights are usually the length of edges. Either Prim's or Kruskal's algorithm provides the best complexity $O(n \cdot log(n))$ for the tree with n vertices. This approach can produce good wirelength estimation in reasonable amount of time.

Minimum Seiner Tree (SMT) Given an edge-weighted graph $G = \{V, E\}$ and a subset $D \subseteq V$, select a subset $V' \subseteq V$, such that $D \subseteq V'$ and V' induces a tree of minimum cost over all such trees [4]. The set D includes all pins of the net, and the set $V' - D$ contains additional *Steiner points*. It is easy to see that SMT is equivalent to MST if $D = V$. However, unlike MST, SMT and many its variant are NP-complete [10].

Rectilinear Seiner Minimum Tree (RSMT) A Steiner tree whose edges are constrained to be rectilinear is called a Rectilinear Steiner Tree (RST). A RSMT is a RST with minimum cost among all RSTs.

RSMT gives the exact wirelength estimation for routing of multi-pin net. This problem has traditionally been viewed as a Steiner tree problem [4]. Therefore, routers employ Steiner tree Wirelength (StWL) model based on RSMT construction. Before layout is completed by routing, the StWL model is also used in other physical design tools, for example, for timing analysis.

The wirelength based on MST is rarely utilized in physical design as a net model, because the MST length was proved by Hwang to be at most 3/2 times larger than the RSMT length. As a result of this theoretical bound, MST serves as either a fast estimation of RSMT, or a starting point to obtain RST by means of local modifications.

However, construction of RSMT is still too computationally expensive for placement algorithms, even using the most recent Steiner tree heuristics [11, 7]. A simple and efficient Bounding Box heuristic is preferred to MST in placement.

Half-Perimeter Wirelength (HPWL) This model is the most popular net model in placement and estimates the netlength by the half-perimeter of the bounding box of a net. It has been proven that this technique provides the optimal solution for 2-pin and 3-pin nets and a lower bound for nets with higher degree, with respect to StWL. The HPWL is very efficient, showing linear complexity $O(n)$ on net degree. However, it can significantly underestimate wirelength for multi-pin nets.

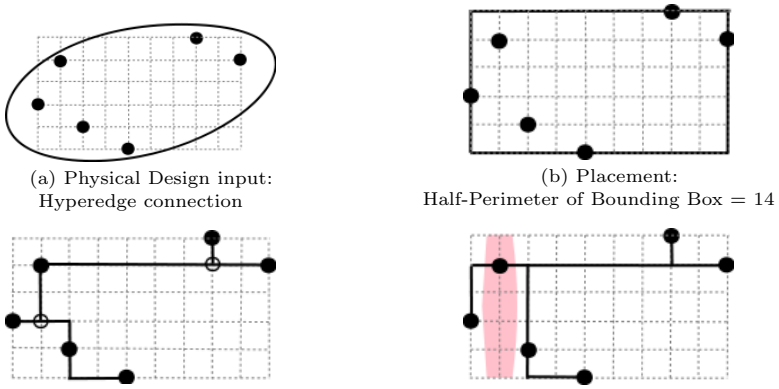
Weighted Wirelength (WWL) Cheng proposed a net weighting technique to scale up the HPWL estimation. The net weights are degree and perimeter dependent constants, which are experimentally determined. However, even for different nets with the same degree, the error in the HPWL estimation can be very different. It is impossible to derive a single net weight to accurately scale up the HPWL estimation for all nets.

State-of-the-art placers and routers traditionally approximate the wirelength with StWL and HPWL respectively. In placement, HPWL is the *first-choice* net model, because it much more faster in comparison with StWL and *empirically* proved to be well consistent with the final routed wirelength. The next example of wirelength estimation underlines the main drawback of the HPWL model: crucial underestimation for multi-pin nets.

Figure 1.1 presents the progression of net model usage in physical design on example of a high degree net with 6 pins. The net is represented as a hyperedge in the input of physical design cycle as shown in Figure 1.1(a). The HPWL estimation of the netlength in placement phase gives 14 units in Figure 1.1(b). More accurate and computationally expensive StWL model is used in routing phase and evaluates the length of 17 units in Figure 1.1(c). Both HPWL and StWL models approximate the final routed wirelength (rWL) in Figure 1.1(d), that is always concerned with error due detours in path because of wire congestion.

Experience in physical design community confirms the HPWL model to be a reasonable and efficient heuristic in placement [5, 12], although the netlength underestimation is significant for multi-pin nets. This inaccuracy of HPWL considered to be admitted, because the percentage of high degree nets in the netlist is typically low and includes around 20%. The research direction of this thesis goes against the main stream and explores the option of using the StWL cost in placement.

If the traditional HPWL net model is replaced by StWL, can the final wirelength be minimized, thus improving the performance of the chip? What kind of heuristic can fit the computational constraints in placement phase and increase the accuracy of estimation? What experimental scheme should validate our approach for different types of placement algorithms? All these questions will be considered in the next section.



(a) Physical Design input: Hyperedge connection

(b) Placement: Half-Perimeter of Bounding Box = 14

(c) Routing: Rectilinear Steiner minimal tree = 17 (Steiner points as white circles)

(d) Routing: Routed Wirelength = 18 (congested region in pink)

Figure 1.1: Wirelength estimation in physical design. Example of a net with 6 pins.

1.4 Statement of the Problem

This thesis is situated on the placement phase of the physical design flow. The strong interconnect issues needed to be efficiently modeled in early placement phase and reflects the importance of the problem. The motivation comes from the necessity to improve the traditional HPWL measure. The goal of the thesis is to design a net model for approximation StWL rather than HPWL. The model must be integrated in any available public placement tool and validated on the academical circuit benchmarks.

StWL cost in placement

The thesis is based on assumption that StWL evaluates the final routed wirelength better than HPWL, and our net model aims to approximate StWL. Optimization of the StWL cost rather than HPWL believed to improve the performance of the chip that will be verified by computing wire delays after placement is finished.

The thesis does not address the problem of congestion estimation during placement. This work examines only influence of the wirelength cost on placement results. The combination of modeling Steiner tree wirelength and congestion-aware technique assumed to consist a good framework for optimization the routed wirelength in future. Such perspectives for placement algorithms underline the importance of our research in StWL optimization.

Clustering approach

The contributions of this thesis is a novel clustering approach to evaluate the length of a net. A simple geometrical heuristic is used to increase the accuracy of HPWL, where bounding box that include pins of the net is divided into regions accordingly to density of pins positions. In terms of net model, splitting the net into several lower degree subnets, the total HPWL of the subnets estimates wirelength significantly better than HPWL of the original net.

To be competitive in CPU time with the HPWL model, an efficient clustering algorithm is proposed with complexity linear on the number of pins. As an advantage of our Steiner tree heuristic, it is *HPWL-based* in the sense that the length of the subnets is evaluated by HPWL. Thus, placer can continue employ well-developed methods on HPWL minimization.

Experimental scheme

Practically, the proposed approach can be integrated in any placement tool by transforming the circuit netlist between any consecutive placement stages accordingly to the new net model. The improvement in the final Steiner tree wirelength and wire delays have been tested for the most recent circuit benchmarks.

1.5 Description of the Chapters

Chapter 2 introduces some basics of the VLSI placement and routing stages. For each stage, formal definition of the problem and common algorithms are presented. These algorithmic details are necessary to introduce the new net model suitable for different placement approaches.

Chapter 3 explains the idea of clustering approach to netlength modeling and describes the designed algorithm. It also proves the linear complexity of the net model.

Chapter 4 presents experimental results on integration of our net model in placement framework. We track the reduction in StWL rather than traditional HPWL, and improvement in wire delays.

Finally, Chapter 5 presents the conclusions of this master's thesis and future research topics.

Chapter 2

Place-and-Route Basics

Placement and routing are the two steps that produce the physical layout of a circuit. The circuit in physical design is represented geometrically, with circuit elements as rectangular cells on the fixed die, and interconnections as net in the hyperedge format. According to such circuit abstraction, place-and-route methods typically employ geometrical-based algorithms and graph algorithms, in order to place the cells and optimize the interconnections, which in turn are presented as a connectivity graph.

The cost function of algorithms for physical layout incorporate factors responsible for the quality of the circuit, such as area, delay and routability. However, the wirelength cost is the main objective in placement and routing which regarded to be consistent with all the costs mentioned above. Special attention will be given to wirelength minimization and the difference of netlength modeling in placement and routing.

Although the research interest of the thesis falls into placement, the description of routing phase is also given. In the current physical design, placers have become more routability-oriented, and optimization of only the wirelength cost in placement is not sufficient for achievement the routable circuits. The contribution of optimization the wirelength cost to the final performance of the circuit will be on the focus of place-and-route description.

The goal of this Chapter is to overview basics of physical design which will be necessary further, for introduction the proposed net model and description the experimental framework. Sections 2.1 and 2.2 present placement and routing steps respectively.

The important material in Section 2.1 is concerned about description of contemporary algorithms used in placement. The design of the net model presented in this thesis is strongly correlated with algorithmic details and a particular implementation of placers. The next Section 2.1 gives the necessary background of placement phase in physical design.

2.1 Placement

Placement is a fundamental problem in physical design because of serious interconnect issues induced delay, routability, and noise especially in deep-submicron designs, which all have to be estimated or resolved in early placement phase. The placement problem has become very active in recent years, and many new academic placers for wirelength minimization were published since 2000. There are many other publications to handle timing, routability and power dissipation.

Placers determine the interconnection of circuit components to the first order. Consequently, routers receive fixed positions of cells and complete interconnections by constructing the exact wire passes. Thus, routers can not improve the wirelength dramatically due to fixed placement of cells.

Other factors underlying the importance of placement are concerned with some requirements to placement algorithms. Placement problem becomes significantly larger, and placers must handle circuits with up to 100 million of cells. Moreover, placement information is needed in early design stages, for example, in logic synthesis. Therefore, scalable and compact placement solutions are essential with *nearly linear complexity* of placement algorithm.

Two main performance cost functions are abstracted in placement. First, minimization of cycle time of the circuit is associated with the reduction the wirelength (estimated), that force cells connected with nets to be placed near each other. Typically, critical nets are considered notable in the netlist for high net weights. Second, placers must ensure the routability of the nets. Although short connections are advantageous for routing, the routing cost is contrary to the wirelength cost for local regions of the layout with high connectivity density of cells.

The next Section 2.1.1 presents the formal definition of the placement problem and put the stress on description of the cost functions.

2.1.1 Placement Problem Definition

Given a netlist and fixed-shape cells, find the exact location of the cells to minimize area and wirelength. The available layout region for placement is presented as rectangular space with terminal input/output cells fixed on the boundaries. The problem can be formulated as follows:

- **Input:**
  - Blocks (standard cells and macros) $B_1, B_2, ..., B_n$
  - Height and width for each rectangular block B_i
  - Nets $N_1, N_2, ...N_m$
- **Output:**
  - Coordinates (x_i, y_i) for each block B_i
  - No overlaps between blocks

  - The total wire length is minimized
  - The total area is minimized or given a fixed die
- **Other considerations**: timing, routability, clock, buffering and interaction with physical synthesis.

Minimization of total occupied area means tight packing of cells. The most critical objectives include wirelength and overlap costs. These two main cost functions are opposite in action, since the shortest wirelength provokes the collapse of the cells in the same location.

In the next Section, the placement algorithm will be considered only as a *wirelength-driven* approach. To take into account other issues like routability or timing, additional cost-aware technique are induced in placement. However, this topic is out of the scope in this thesis.

Design Style Specific

Geometrical design of the circuit can be generated by using different design styles which correspond to the physical implementation of rectangular shapes and channels for passing wires. In this thesis, the methodology of standard cells is considered, since it is very popular in the physical design and the most academical benchmark circuits are presented in this format.

The standard cells hold the same height and have to be placed in predefined rows. All rows are typically of the same width. Thus, circuit elements are ranged only in width when mapped from the physical library. The wires are physically routed along empty spaces over the standard cells rows. The standard cells are designed such a way that the power and ground nets run horizontally through the top and bottom of cells.

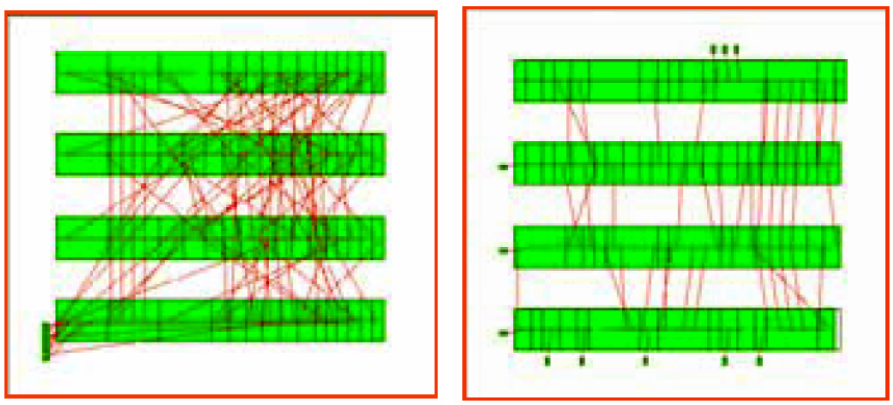
The physical properties of the standard cell design is critical for routing step, whereas placement considers design specifics only in the last phase when cells have to be fitted to rows exactly.

2.1.2 Global and Detailed Placement

Traditionally, placement is separated into two phases, global and detail placement. The mail goal of global placement is to distribute the cells evenly over the circuit layout, in order to minimize certain objectives such as wirelength. Overlaps of the cells are admitted and constrained lightly. The next Section 2.1.3 is focused on overview of global placement algorithms, since they perform the most of work in placement.

Detailed placement performs finer work on legalization of the cells fixing overlapping. The minor task of detail placer is further improvement in wirelength and, possibly, in timing and routability. However, detailed placers are restricted in runtime and can be requested only to assign the cells positions without overlaps.

Current detailed placement algorithms employ greedy heuristics on local perturbation of nets for better placement. The reduction in wirelength can achieve



(a) Bad Placement (b) Good Placement

Figure 2.1: (a) Bad and (b) Good Placement in terms of wirelength [1].

several percents in respect with wirelength after global placement. In this thesis, this ability of recent detailed placement tools is used to test the proposed net model, by changing input netlist and testing how detailed placer improves results due to these changes.

Before going to the next Section 2.1.3 describing placement algorithms, all information given above can be demonstrated in Figure 2.1. Standard cells design is depicted with green rows and cells tightly placed on them; wires are denoted with red lines. Consequently, the placement of cells is legalized and corresponds to the detailed placement. The Figure shows the crucial role of wirelength cost in producing the placement layout. Placers must tend to the good layout in Figure2.1(a) rather than the disordered layout in Figure2.1(b).

2.1.3 Placement Algorithms

Global placers regarded to be more complex and more important in placement. This Section overviews different approaches to perform global placement. The constant challenge of the algorithms is scalability. The problem size of the circuits is steadily increasing and getting closer to one billion of components, whereas complexity of the algorithms must be nearly linear.

Placement approaches can be different in implementation, but all have two main cost functions, wirelength of nets and overlaps of cells. The implementation of the algorithm must concern the way of expressing and optimization of these two costs.

The wirelength objective is traditionally approximated with HPWL measure. Since the HPWL minimization placement problem is NP-hard [13] and inapproximable [14], placers optimize HPWL heuristically by applying such methods as min-cut partitioning, quadratic or analytical solvers, or simulated annealing. All the methods are described in the next Subsections.

In order to produce routable designs, placers typically combine HPWL cost optimization with different congestion-aware techniques, for example, [15, 16,

Algorithm 1 Simulated-Annealing Placement Algorithm

Require: A circuit
Ensure: Placement
1: Initialize temperature: T
2: Initialize placement: P (randomly)
3: **while** $T < T_{final}$ **do**
4: **while** little overlapping **do**
5: Set new placement: P_{new} = PERTRUB(P)
6: Compute changes in the cost: δC = COST(P_{new}) - COST(P)
7: **if** $\delta C < 0$ **then**
8: $P = P_{new}$
9: **else**
10: **if** $RANDOM(0,1) > e^{\delta C/T}$ **then**
11: $P = P_{new}$
12: **end if**
13: **end if**
14: T = SHEDULE(T)
15: **end while**
16: **end while**
17: **return** Placement P

17]. Since the approach presented in this thesis is based on minimization of StWL, such techniques are not presented in this work, but the most recent overview can be found in [18].

The further description of the algorithms is accompanied with correspondent net models on wirelength estimation, which become the important part of the algorithm. Most of the models tend to approximate the HPWL metric, contrary to the new net model of the StWL cost proposed in the thesis. However, the approaches on the netlength modeling introduced in this Section present the background for the new net model, which will be necessary in the next Chapters.

Simulated-Annealing Placement

Simulated-annealing approach is widely used for circuit placement. This algorithm simulates *natural phenomenon* of annealing process in crystals and *proved* to find the global optimum if there is no limit in runtime. In practice, the main problem of these algorithm is CPU time constraints. First efficient implementation was proposed in TimberWolf 3.2 placer [19], which has become a classical implementation of the Simulated-Annealing approach.

The outline of the TimberWolf algorithm is presented in Algorithm 1. The placement is firstly generated randomly and then improved iteratively with decreasing temperature. The state of the system, which is placement of the circuit, is changed by perturbation function `PERTRUB`. The cost function `COST` estimates the quality of the placement for the given temperature. The placement is updated in two cases, if the improvement in cost is positive (line 8) or the system is not cold enough to accept the random move (line 11).

The contribution of TimberWolf implementation consists in a good choice of functions `PERTRUB` and `SHEDULE` which simulate freezing of the placement state-based system. Particularly, the perturbation includes tree types of operations: move a circuit block, interchange between blocks and interchange in orientation for a block.

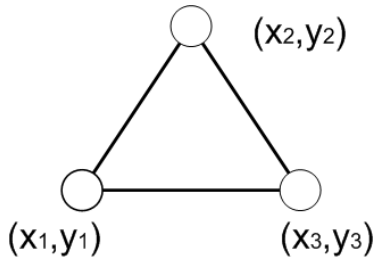


Figure 2.2: Decomposition of a tree-pin net to two-pin connections.

The `COST` function incorporates typical placement objectives of wirelength, overlap penalization and timing violations. The wirelength is traditionally expressed with the HPWL model, although it can be any net model which fits to runtime bounds.

$$WL^{SA} = HPWL \tag{2.1}$$

Analytical Placement

The analytical approach employs methods of mathematical programming to solve the placement problem. The cost of the algorithm is split into two wirelength and overlap components as for other placers, but both cost functions must be smoothly convex and continuously differentiable, in order to use differential calculus.

The overlap component can be modeled by any heuristic that penalize the collapse of cells. Typically, the bell function is used for this. The other component of wirelength supposed to approximate the HPWL measure, which is not a convex function. The great finding here is a Naylor function presents an elegant way of expressing the wirelength by sum of log-exponential expressions.

Given a net with pin coordinates $\{(x_1, y_1), (x_2, y_2) \dots (x_k, y_k)\}$, the wirelength objective is

$$WL^{AP}(\alpha) = \alpha{\cdot}(\ln(\sum e^{x_i/\alpha}){+}\ln(\sum e^{-x_i/\alpha})){+}\alpha{\cdot}(\ln(\sum e^{y_i/\alpha}){+}\ln(\sum e^{-y_i/\alpha})) \tag{2.2}$$

The function $WL^{AP}(\alpha)$ converges to HPWL as α converges to 0.

Non-linear mathematical methods on optimization the differentiable cost function are used to cope with the optimization task. The most famous representatives of analytical placement are mPL [20] and Aplace [16].

Quadratic Placement

Quadratic placers also belong to the group of analytical approaches, because they use quadratic wirelength objective. Such abstraction gives inaccurate estimate, but, instead, the wirelength can be minimized very efficiently by solving a system of linear equations.

Quadratic placers can operate *only* with two-pin nets, so each net (originally presented in the netlist as a hyperedge) is decomposed into a set of two-pin

Algorithm 2 Quadratic Placement Algorithm

Require: A circuit
Ensure: Placement
1: Convert hyperedges in the netlist to 2-pin connections
2: Compute wirelength: $WL = \sum_i WL^q(i)\ \forall$ nets i
3: Placement $P \leftarrow$ System of linear equation derived from WL
4: **while** little overlapping **do**
5: Compute wirelength: $WL = \sum_i WL^q(i)\ \forall$ nets i
6: Add overlap constraints (also quadratic): $WL + Overlap$
7: Placement $P \leftarrow$ System of linear equation derived $WL + Overlap$
8: **end while**
9: **return** Placement P

connections. Figure 2.2 presents a net with three pins, and quadratic placement approach computes the netlength as:

$$WL^q = \sum_{i=1, j \neq i}^{3} w_{ij} \cdot [(x_i - x_j)^2 + (y_i - y_j)^2] \tag{2.3}$$

where w_{ij} are net weights. The adaptation to the HPWL metric consists in proper calculating of net weights w_{ij}.

However, what happen if the net has more than three pins? Special net models for hyperedge decomposition problem are presented further. Before this description, the sketch of the quadratic placement is depicted in Algorithm 2.

The initial placement can be computed regardless of any overlap constraints (lines 1-3) if the circuit includes fixed pads in the netlist, which form the numerical vector for the system of linear equation. The obtained initial placement is full of overlapping. To spread cells over the layout, quadratic forces are induced to remove overlaps, and the placement is recomputed in the **while** loop.

The quadratic approach is very popular in physical design. Sine the connectivity matrix in the system of linear equation is sparse, the calculation of placement solution can be performed efficiently by mathematical solver of system of linear equation, for example, LASPack package [21].

In the next paragraphs, some net models for quadratic placement are presented, that differ in the way of multi-pin net decomposition and calculating net weights for simulating the HPWL measure.

Clique and Star models. We have seen how to translate a net with three pins to a set of two-pin connections in Figure 2.2. Now we examine the general case of a net with k pins and weight W. The clique and the star net models are traditional for quadratic placement. The first model replaces a net by $k(k-1)/2$ two-pin nets forming a clique. The second introduces a fake star pin and yields k two-pin nets. Figure 2.3 illustrates the case when $k = 5$.

The quadratic wirelength for a k-pin net is expressed as follows:

$$WL^{clique} = \sum_{i=1, j \neq i}^{k} w^{clique} \cdot [(x_i - x_j)^2 + (y_i - y_j)^2] \tag{2.4}$$

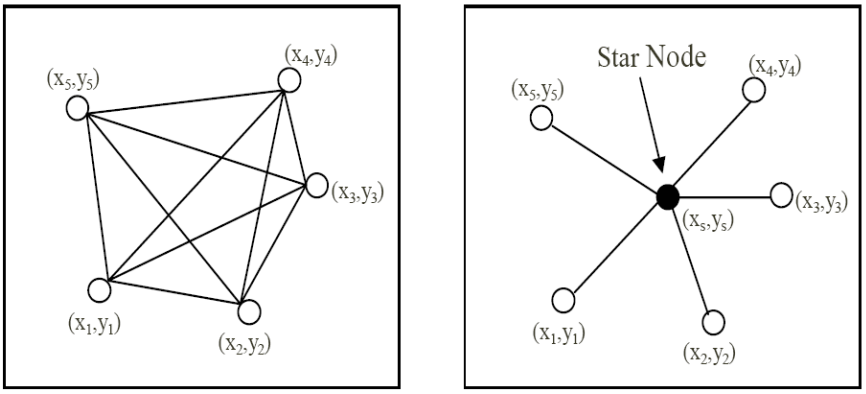


Figure 2.3: Clique and Star net models [2] for 5-pin net.

$$WL^{star} = \sum_{i=1}^{k} w^{star} \cdot [(x_i - x_{star})^2 + (y_i - y_{star})^2] \tag{2.5}$$

One can readily see that the star node is a midpoint of all k pins, and the two models are equivalent [2] when the weights are related like $w^{star} = k \cdot w^{clique}$.

The star model is preferred for multi-pin nets, since the number of two-pin connections is equal to k, rather than $k(k-1)/2$ for the clique model (quadratic dependency on k).

A quadratic cost function advantages the placement algorithm to be fast and effective. A number of net weighting heuristics were proposed to adjust quadratic wirelength to the realistic *linear* wirelength. Factor $2/k$ is to adapt the total net weight to the number of edges in a spanning tree connecting all pins [22]. The additional net weight λ can be used to linearize the quadratic length [23]. The most recent and successful technique to overcome quadratic nature was proposed within the next net model.

Linear Bounding Box model. The authors in [3] showed that error in wirelength estimation by the clique model can reach 150% for randomly generated

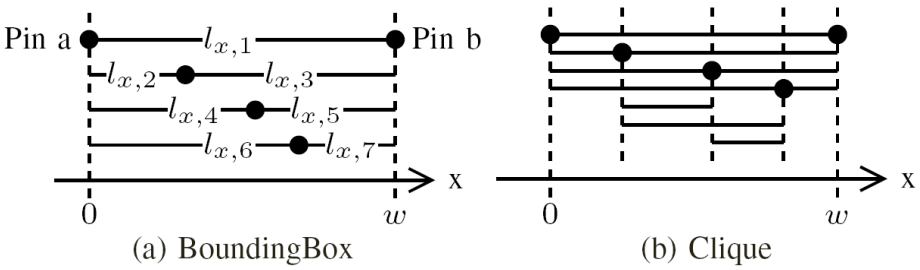


Figure 2.4: Bounding Box and Clique net models [3] for 5-pin net in x-direction.

nets. They picked the HPWL metric and proposed a Bounding Box net model, which is linear rather than quadratic and *exactly* equivalent to HPWL.

In the Bounding Box model, a hyperedge in the netlist is not transformed into all possible two-point connections, as it is performed by the clique model. Only a few characteristic pins are selected, as illustrated in Figure 2.4 a. Two boundary pins a and b have lowest and highest X coordinates respectively. These two pins are connected with each other, and all remaining $k-2$ pins of the net are connected with both outer pins a and b. That results in total number of two-pin connections equal to $1+2(k-2)$, which is linear on the number of pins k as for the start model.

The quadratic wirelength for the decomposed net is:

$$WL_x^{bb} = \sum_{i=1}^{1+2(k-2)} w_{x,i}^{bb} \cdot {l_{x,i}}^2 \tag{2.6}$$

Calculations are the same for Y direction. The linearization of the length WL_x^{bb} is achieved by assignment of weights selected like $w_{x,i}^{bb} = 1/[(k-1) \cdot l_{x,i}]$. Finally, the WL_x^{bb} is *exactly equivalent* to the HPWL of the net.

Partition-based Placement

Partition-based placement algorithms decompose a given placement problem to smaller subproblems by subdividing the placement region. Circuit cells are assigned to subregions cutting the netlist hypergraph connections [24]. Such min-cut placers generally use either bisection or quadrisection to divide the placement area and netlist. Fiduccia-Mattheyses heuristic and derivatives [25, 26] are typically used for operations on the netlist. Additionally, some quadratic placement and geometric partitioning methods [27] can be also utilized.

Figure 2.5 shows an example of partitioning processing from initial random placement and two consistent bisections. The accompanying procedure in circuit partition is *terminal propagation* [28] where nodes external to the regions being partitioned are propagated as fixed terminals to them. Consequently, movable cells are positioned closer to their terminals in partitions, hence reducing wirelength.

The min-cut cost serves as an objective of the algorithm, so there is no implicit net model like in quadratic placement framework. The main challenge is to associate the min-cut cost with a selected netlength metric by performing a weighted min-cut. Further, we overview some a method for proper calculation of weights, in order to make equal min-cut cost to HWPL.

Weighted min-cut for HPWL. For each net in each partitioning block, one must calculate the cost of all pins on the net being placed in partition 1 (w_1), the cost of all pins on the net being placed in partition 2 (w_2) and the cost of all pins on the net being split between partitions 1 and 2 (w_{12}). For simplicity sake,

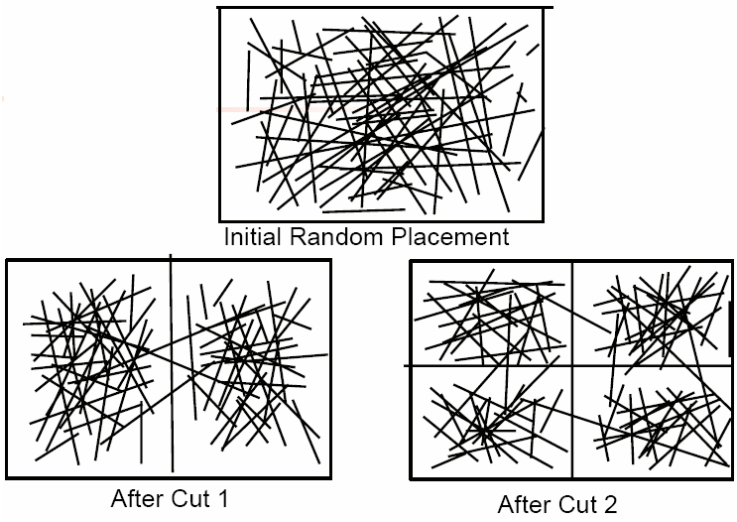


Figure 2.5: Min-cut placement approach [1].

we address a 3-case analysis [29], which minimize total HPWL during min-cut placement.

Up to two hyperedges can be created in the partitioning block, one with weight $|w_1 - w_2|$ and the other with weight $w_{12} - max(w_1, w_2)$. The only assumption is that $w_{12} \geq max(w_1, w_2)$. The HPWL of the set of pins necessary to calculate w_{12} is at least as large as that of w_1 and w_2 since it contains an additional points - the centers of two partitioning blocks. More details are supported in [29].

In addition, we mention the authors of [30], who introduced a new terminal propagation technique that allows the partitioner for better mapping net-cut to HPWL.

Conclusion

Having in mind that the new thesis' net model is oriented in approximation of StWL cost, we should answer the question whether the contemporary net models described above are able to simulate StWL. Such modeling is possible with the only condition that one needs to build Steiner tree, in order to know the subject of simulation. Contrary to the HPWL bounding box, the construction of Steiner tree is NP cost problem. Some heuristic is required for either estimation the StWL or emulating the paths of Steiner tree.

To evaluate the StWL cost, MST or even HPWL are typically used, but the results are not satisfactory. The most recent heuristic in Steiner tree construction [7, 11] regarded to be still computationally expensive in placement [31]. Therefore, the new clustering approach to netlength modeling proposed in this thesis can fill the gap on StWL minimization in placement.

The next Chapter directly addresses the new net model, whereas this Chapter ends up with some short introduction to routing problem.

2.2 Routing

Contrary to placement, routing is a well studied problem, and several hundred articles have been published about all of its aspects. Since almost all problems in routing are computationally hard, the researches have focused on heuristic algorithms. Complete routing of all the connections cannot be guaranteed in many cases because of hard physical interconnect issues. As a result, techniques as rip-up and re-route are employed to removes some complex connections and re-routes them in a different order.

2.2.1 Routing Problem Definition

Given a placement of circuit cells and a number of available metal layers, find a valid pattern of horizontal and vertical wires that connect the pins of the nets. The wirelength is estimated with StWL, but the netlength can deviate the paths of Steiner tree due to congestion of wires.

Formulation of the routing problem is as follows:

- **Input:**
  - Netlist $N_1, N_2, ...N_m$
  - Timing budgets for, typically, critical nets
  - Location of blocks $B_1, B_2, ..., B_n$ (determined from placement step)
  - rectangular shapes for each block B_i
- **Output:**
  - Geometric layout of all nets.
- **Objectives:**
  - Minimize the total wire length, the number of vias, or just completing all connections without increasing the chip area.
  - Each net meets its timing budget.

The traditional approach to routing divides it into two phases. The first stage, called global routing, assigns a list of routing regions for each net without specifying the actual geometric layout of wires. The second stage, called detailed routing, determines the exact route and layers for each net.

Conclusion

This chapter has presented placement and routing problems in physical design showing the difference in interconnect optimization at each step. Placer can afford only estimate the wirelength, and the netlength modeling problem is translated into the task of approximation HPWL or StWL. In routing, physical

effect of wires are considered which are congestion of wires and consequent bends in wire paths. The other cost is the number of vias of wires that must be considered and minimized.

Therefore, in order take into account complete routing cost in placement, placer must simulate the behavior of router which is not possible. From this point of view, the criteria of minimal StWL after placement seems to be reasonable for validation the net model proposed in the thesis.

Chapter 3

Multi-Clustering Net Model

This chapter starts with overview of Multi-Clustering Net (*MCN*) model proposed in the thesis. Section 3.1 presents the motivation of the new model and demonstrates the reasoning of our clustering approach to approximate the Steiner tree measure. Some basic concepts of designed algorithm and experimental scheme are also shortly introduced.

The implementation of the net model is based on k-means clustering algorithm [32] which is described in Section 3.2. This method came from data mining and has been adopted for our needs of geometrical clustering of net pins. It is able to produce clustering very fast for a given number k of clusters.

The whole clustering algorithm runs k-means implementation iteratively looking for the optimal number of clusters k. Correspondent local search algorithm and score function are described in Section 3.3.

The clustering algorithm is applied to a multi-pin net, which pins are connected by hyperedge as defined in the input netlist. The output of the algorithm is a union of subnets, also in the hyperedge format, that not only preserves connectivity of all the pins, but also improves interconnection by reflecting geometrical positions of the pins. The final Section 3.4 explains the way of representing the subnets in the output netlist.

3.1 Overview

The thesis addresses the problem of wirelength evaluation for multi-pin nets (more than three pins) in placement. The traditional HPWL model is adequate for nets with two or three pins, but it can crucially underestimate wirelength for nets with more pins. To overcome the deviation, a high degree net is broken into several subnets by the clustering approach described further. In order to estimate the netlength, the HPWL measure is applied to a resulting union of the subnets.

On the *MCN* model, we assume that the pins with closest position form the subnets. Furthermore, the Steiner tree of the original net is likely to be within

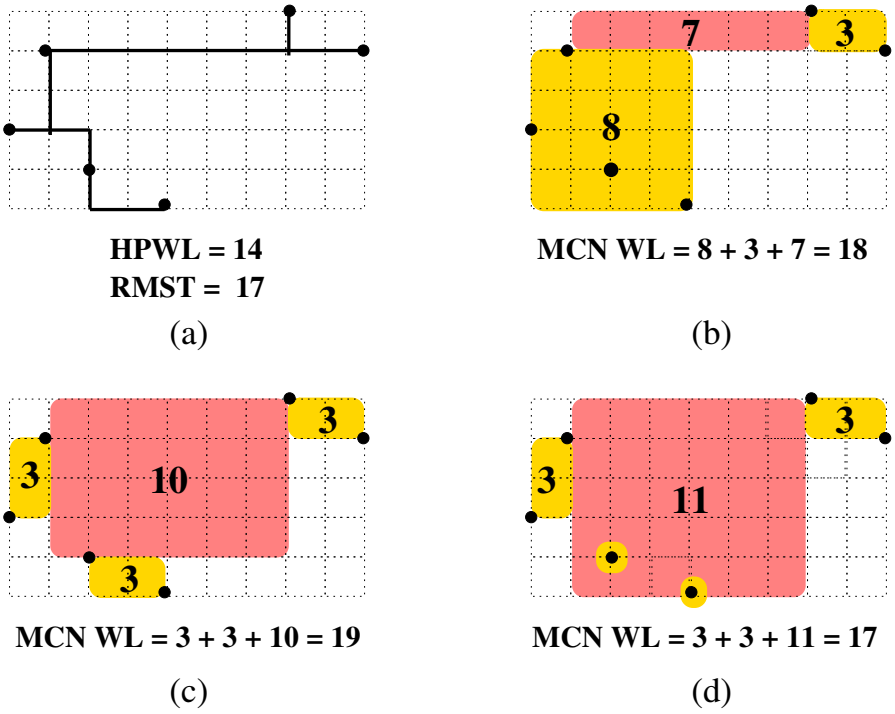


Figure 3.1: Example of wirelength estimation for 6-pin net: (a) traditional HPWL and RSMT measures; (b,c,d) Multi-Clustering Net model.

the HPWL bounding boxes of the subnets. *The sum of the HPWL of the subnets* gives more accurate estimation, mainly because the subnets have smaller pincount than the original net. Our empirical results demonstrate the efficiency of the *MCN* heuristic in StWL approximation and prove more precision with regard to the HPWL. The next example confirms the statements discussed above.

Motivation Example

Figure 3.1 illustrates how the wirelength can be estimated for a net with 6 pins. Figure 3.1(a) depicts two traditional measures RSMT and HPWL applied to the net. The RSMT gives the exact value of 17 length units, whereas the fast bounding box heuristic of HPWL concludes 14 units and undervalues the netlength. One can easily see that pins placed inside the bounding box do not contribute to the netlength estimation, and the only pins that determine the length are positioned on the boundaries.

In order to take into account "internal" pins and simulate "internal" routes of Steiner tree, the net is split into subnets (light rectangles) and one additional subnet (dark rectangle) is introduced to connect them, as shown in Figures 3.1(b,c,d). The subnets are selected with the main purpose to span regions with the most density of pins. The implementation of pins clustering to subnets will be presented in the next Section 3.2. Furthermore, we employ the traditional HPWL measure to each subnet, although other metrics are possible.

Figure 3.1(b) presents grouping the net into three subnets, and the netlength of 18 units reasonably approximates the exact Steiner tree length of 17 units. The next Figures 3.1(c,d) depict the *MCN* approach for four and five subnets respectively. Moreover, the *MCN* model is able to reach the exact RSMT wirelength in the last case.

In result, each net model on Figures 3.1(b,c,d) is capable to approximate StWL much more accurately than HPWL. However, how many subnets are necessary to produce an optimum solution: three, four or five? This issue is tackled in the next subsection.

Designed Algorithm

The *MCN* model aims at obtaining a better approximation of the netlength for a given net. The accuracy of the wirelength estimation is increased by splitting the original net into several subnets.

In this thesis, an optimization algorithm has been designed to explore the best number of subnets. Internally, the problem is simplified to cluster the pins into *subsets* by closest pins position. Given the number of subsets k, all the pins of the net are divided into subsets by the well-known k-means algorithm [32]. The best configuration of subsets is selected based on a local search algorithm. Iteratively, the increasing number of subsets is explored and the best one is chosen. Sections 3.3 presents all the algorithmic details further.

When describing the algorithm, two terms *subsets* and *subnets* are widely used in this work. In order to be clear, the formal difference between them is as following. When pins clustering is performed, clusters are also referred to as subsets of pins. Subsets mean only geometrical regions where the pins are grouped. Once pins are assigned to subsets, the connectivity inside subsets and along themselves can be assigned and expressed in terms of subnets. For example, the subnets depicted in Figures 3.1(b,c,d) as light rectangles also present the subsets, but the subnet of dark rectangle does not.

In this work, subnets are typically represented as hyperedges or bounding boxes of the HPWL measure. The construction of the subnets is straightforward from the subsets. The only problem lies in the way of connecting the subsets along each other without inducing any new pins like the fake star-node in Star net model. The proposed method of construction the subnets is described in Section 3.4.

Experimental scheme

The experimental framework is an important issue in the thesis for validation the *MCN* ideas, especially because the authors do not have a placement tool, where the model can be induced and tested. However, there are many academical placers available for the physical design community. These placers typically operates in two steps global and detailed, and the input information for each phase is the netlist of a circuit.

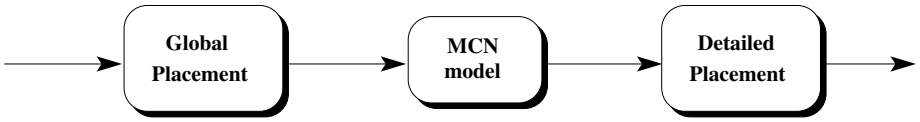


Figure 3.2: Placement flow with the *MCN* model.

Due to separability of placement flow into two steps, one can avoid integration of the net model in particular implementation of the placer. Instead, the netlist, which contains all interconnection information, can be changed according to the *MCN* algorithm. Therefore, the effect of the *MCN* approach on real placement algorithms is examined by transforming a circuit netlist between global and detailed placement steps.

The experimental scheme is based on the outline of Figure 3.2. The layout produced by global placement is captured, the new modified netlist is built and passed to detailed placement. In other words, the *MCN* algorithm transforms the input netlist to the output netlist based on the input placement information.

Although all nets in the netlist are represented in hyperedge format, detailed placers typically employ the HPWL measure to nets. Thus, if original multi-pin net is replaced by union of subnets in the netlist according to the *MCN* net model, the HPWL applied to the subnets exactly corresponds to the *MCN* wirelength representation.

In conclusion, it is important to underline that the new net model has been design, in order to be introduced in the placement framework for improvement the wirelength expressed in StWL rather than HPWL. The placement results obtained by following the scheme on Figure 3.2 must be compared with the common two-phase placement flow without the block of the *MCN* model. These results are presented in the next Chapter 4.

Moreover, the proposed approach to modeling StWL can not be considered as a pure Steiner tree heuristic with computation of Steiner points and tree paths. Instead, the *MCN* net model is practically oriented to simulate the StWL cost in placement by means of fast clustering and HPWL heuristics. Consequently, the accuracy of the net model can be proved only empirically.

3.2 K-means algorithm

The k-means clustering algorithm [32] is commonly used in data mining where efficient algorithms were proposed to process large quantity of data [33]. The clustering is stated as *Classification problem* for multivariate observations. Each observation or data point is described with m variables, and the task is to group data points such a way that similar data points are joined together.

The k-means algorithm is formulated as follows. Given a set of n data points and an integer k, determine a set of k points, referred to as *centroids* of clusters, such that the *squared* Euclidean distance from each data point to its nearest centroids is minimized. The dimension of data space is determined by the

Algorithm 3 K-means Algorithm

Require: A set of n points and the number of clusters k
Ensure: Clustering of points into k subsets
1: Initialize centroids of k clusters.
2: **repeat**
3: Each point finds out which centroid it is closest to.
4: Each cluster finds the centroid of the points it owns and jumps there.
5: **until** no jumps
6: **return** k subsets of points

number of variables that is equal to m, and the Euclidean distance is computed in m-dimensional space.

The general description of k-means algorithm is presented in Algorithm 3. Initially, k points are chosen as the potential centers of the clusters in an appropriate way. The main part of computations is performed inside the loop, where each of n points is assigned to the closest cluster. Then all centroids of k clusters are updated as the center of gravity of the points of the cluster. The calculation stops when a stable configuration is reached for all the centroids.

Such approach of iterative adaptation of centroids of clusters appeared to be very efficient. The complexity of the algorithm is $O(kni)$, where k is the number of clusters, n is the number of points to be clustered, and i the number of iterations to converge. In our case, k is the number of subsets of pins and n is the total number of pins of the net, which both are typically small. Experimentally, the algorithm converges very fast when n is small, thus showing linear complexity on n.

In terms of clustering quality, the k-means algorithm does not guarantee the optimal solution. However, the obtained results considered to be reliable and very close to the optimum. Since the algorithm is iterative in nature, its evolution highly depends on the step of defining the first positions of centroids (line 1 in Algorithm 3). The choice of the best initial configuration is out of the scope of this thesis and can be found in the literature [32, 33]. The common approach of randomization of the centroids is used for the *MCN* net model, where k pins of a net are arbitrarily selected as initial centroids.

When the number of clusters is unknown, k-means algorithm typically is combined with some heuristic, which introduces a cost function for estimation the clustering quality and explores the best number of clusters interactively. Such approach with wirelength-based cost function and local search algorithm has been implemented in this work and can be found in the next Section 3.3. This Section ends up with a small example which demonstrates the processing and the power of the k-means algorithm.

Demonstration example

Figure 3.3 presents an example of the k-means algorithm when two clusters ($k = 2$) are sought. A net with eight pins is depicted in Figure 3.3(a). The algorithm starts in Figure 3.3(b) with random selection of two pins A and B as the initial positions of centroids of clusters. Consequently, all the pins are

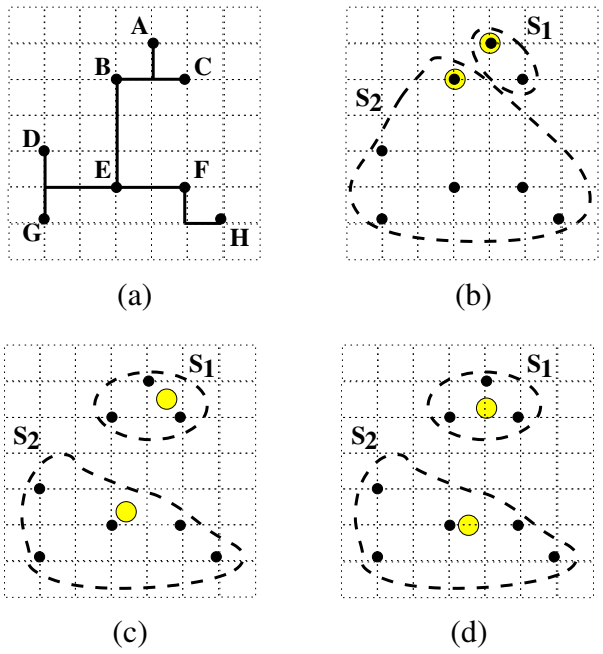


Figure 3.3: K-means algorithm for $k = 2$: (a) example of a net with 8 pins, (b,c,d) evolution of the algorithm.

assigned to two subsets S_1 and S_2 which is closest to, as shown with shadowed circles. The classification of pins to the subsets looks like $S_1 = \{A, C\}$ and $S_2 = \{B, D, E, F, G, H\}$.

Figure 3.3(c) depicts the second iteration of the algorithm, when the centroids have been re-computed and some pins change the subsets. Particularly, pin B jumps from S_2 to S_1 subset, because the centroid of S_1 has become closer than the other. Figure 3.3(d) depicts the convergence when the centroids are re-computed again and neither pin moves.

The final classification $S_1 = \{A, B, C\}$ and $S_2 = \{D, E, F, G, H\}$ can be predicted, since the example of the net is quite small and the pins are clearly separated to top and bottom regions. On the other part, two initial centroids A and B were selected rather close to each other and in the same top region, that could affect the final solution and bring clustering to local optima. However, the following evolution of the algorithm demonstrated the power of k-means approach to produce expected results for the given example. In practice, the random selection of initial centroids proved to work reasonably for the most cases.

In conclusion of this Section, it is necessary to mention some extensions of the k-means algorithm where the number of clusters k is not given and has to be evaluated. The most famous algorithm is called ISODATA [32] which determines the best number k internally by running the k-means algorithm several times. The ISODATA algorithm is discarded for application in this thesis, because of

Algorithm 4 Clustering Algorithm

Require: A net N
Ensure: A group of subsets of pins
1: Cost of clustering: $Cost \Leftarrow$ Cost of net N
2: Number of subsets: $k \Leftarrow 2$
3: **repeat**
4: centroids of subsets: $\{C_1, \ldots, C_k\} \Leftarrow k$ random points
5: **while** changes in $\{S_1, \ldots, S_k\}$ **do**
6: $\forall i \in \{1, \ldots, k\}$, $S_i \Leftarrow$ Pins of net N closer to C_i
7: $\forall i \in \{1, \ldots, k\}$, $C_i \Leftarrow$ centroid of S_i
8: **end while**
9: $Cost \Leftarrow$ Cost of the k subsets $\{S_1, \ldots, S_k\}$
10: $k \Leftarrow k + 1$
11: **until** no improvement in $Cost$
12: **return** $\{S_1, \ldots, S_k\}$

very strong computational demands.

Another approach consists in estimation of density distribution of data points and prediction the number of clusters according to the density map. This heuristic can provide different results depending on the given level of density. However, the density approach can be tested and adopted for the *MCN* model in the future work.

In this thesis, we have chosen the classical way of determine the number of clusters by iterations of the k-means algorithm. The description of the designed implementation is presented in the next Section 3.3.

3.3 Clustering algorithm

The clustering algorithm is presented in Algorithm 4. The task is to group pins of the targeted net into subsets with the closest positions. The cost of the algorithm is a distribution of subsets balanced in terms of pins density. This geometrical-based approach does not construct interconnection paths like Steiner tree or even HPWL bounding box. Instead, the goal of clustering algorithm can be viewed as reduction of problem size in netlength modeling expressed in splitting the original net into subnets with smaller pincount.

The algorithm assumes the targeted net as the initial solution of one cluster. Then it iteratively explores several clustering by incrementing the number of possible subsets in the outermost **repeat** loop. The k-means algorithm is applied in the innermost **while** loop to obtain the new clustering of the pins. The calculations stop when there is no improvement in the cost, that corresponds to *local search algorithm* manner.

A drawback of the k-means algorithm is that the number of clusters k must be supplied as an input parameter. In other words, the number of subsets will depend on the particular placement of net pins and can not be derived only from net degree. To evaluate the clustering quality, we define a cost function that aims at minimizing the total inter- and intra-clustering variance:

$$Cost = \underbrace{\sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - C_i)^2}_{inter-clustering} + \underbrace{\sum_{C_i \in C} (C_i - C_T)^2}_{intra-clustering}$$

where k is the number of clusters, C_i is the centroid of the cluster S_i, and C_T is the center of gravity of the centroids of the clusters.

The devision the cost into two components is common for clustering problem. The inter-clustering variance penalizes far-away points and leads to more compact clusters, whereas the intra-clustering variance constraints large connections among clusters. The quadratic function emphasizes the penalty effect.

To compute the inter-clustering variance, the Star model [2] assumed to estimate the distance along pins where the centroid is the star node. The star model is also applied to calculate the intra-clustering variance. In this case, the center of gravity C_T of the centroids of the subsets corresponds to the star-node.

In terms of data mining, star-node corresponds to mean point of a given set of points, that is the centroid of subset in our case. The distance to the mean point is a typical score for clustering. Moreover, the proposed cost function can be an advantage in application with the Star net model in quadratic placement.

Contrary to typical examples in processing large amount of data, the number of net pins is comparable with the number of clusters. Cost function can even be calculated for the number of clusters equal to the net degree. One can easily see that the cost function gives the same value for two boundary cases when $k = 1$ and $k = n$. Additionally, the inter-clustering variance is going down when k is increasing, because pins are joint to more clusters and more closer to the cluster centroid. The behavior of intra-clustering variance is opposite, and the function is increasing with more clusters used in the algorithm. The difference in two variances will be observed clearly on demonstration example described in the next Subsection.

Therefore, the cost function of clustering score assumed to have "smooth path" to a global minimum which appears at the point of contradiction between two variance components. This property validates the usage of the local search algorithm in the clustering algorithm. Moreover, we experimentally observed no significantly improvement on further exploration in larger number of subsets after a worst solution is found.

In terms of performance, the complexity considered to be *linear* with regard to the number of pins due to the small number of k-means algorithm runs and the small number of explored subnets. The number of subnets does not exceed 6 in practice and is equal to 3-4 in average on the experiments. The following example presents typical run of the clustering algorithm where net is finally split into 3 subnets.

Demonstration example

The *MCN* model operates in two independent steps, in order to spit a net into subnets. First, pins of the net are grouped into *subsets* with the closest positions. The clustering algorithm takes into account only placement of pins and does not estimate the wirelength of path along pins. Second, connectivity of pins is assigned when the pins have been divided into subsets, thus forming the *subnet* connections.

Figures 3.4(a,b,c,d) show an example of the clustering algorithm during three iterations. The centroids of the subsets are labeled with C_i where i is the number of the subsets. C_T corresponds to the center of gravity of the centroids. The two elements in the cost correspond to the inter- (light color) and intra-clustering (dark color) variance respectively.

The original net is assumed as the initial solution in Figure 3.4(a). Figure 3.4(b) corresponds to the clustering into two subsets derived from the example shown in Fig. 3.3. The algorithm stops when $k = 4$ in Figure 3.4(d) because the cost is not improved. Finally, the solution with 3 subsets reported on Fig. 3.4(c) is selected.

The last Figure 3.4(e) presents subnets, which are derived from the subsets of pins obtained by the clustering algorithm. In this thesis, hyperedge is used to

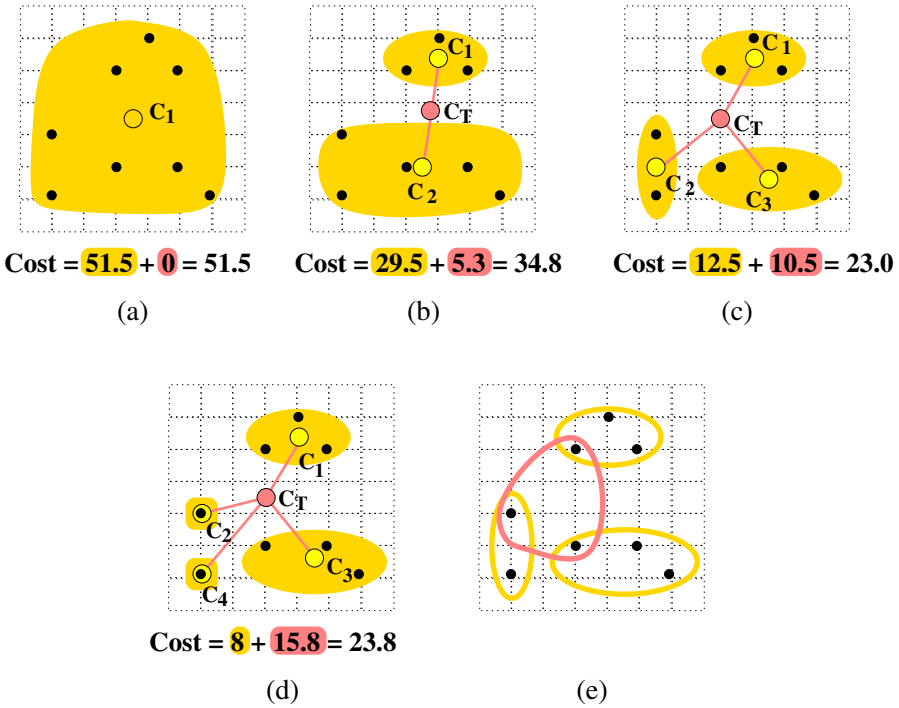


Figure 3.4: Clustering algorithm: (a) example of a net with 8 pins. Clustering pins into (b) 2 subsets, (c) 3 subsets and (d) 4 subsets. (e) Resulting interconnection of subsets with hyperedges for (c) solution.

connect pins inside the subsets (light curves) rather than star or clique model. The subsets are also connected by hyperedge (dark curve).

The details of construction of subnets are described in the next Section 3.4. Special attention is given to the issue of interconnecting pins without inducing any additional fake pins.

3.4 Construction of subnets

The *MCN* model is integrated in placement flow such a way that it changes the netlist between global and detailed placement steps. Following the clustering algorithm applied to nets in the input netlist, each of these nets must be replaced by correspondent group of subnets in the output netlist. Thus, construction of subnets is responsible for the format used to represent the subnets in the netlist.

Typically, all nets are written in the netlist as hyperedges. Although hyperedge can be converted to the clique or star connection, the hyperedge format is always preferable to express the interconnection for placers, because it gives total freedom to the posterior placement steps for conversion the hyperedge to any model on wirelength estimation.

Figure 3.5 introduces to the problem of subnet assignment when the pins clustering into subnets has been performed for a net with 9 pins. The picture on the top depicts tree subsets of pins which are needed to be interconnected. If only the pins in the subnets are connected, the original hyperedge connection of all 9 pins will be lost. Thus, connectivity along the subnets is also required.

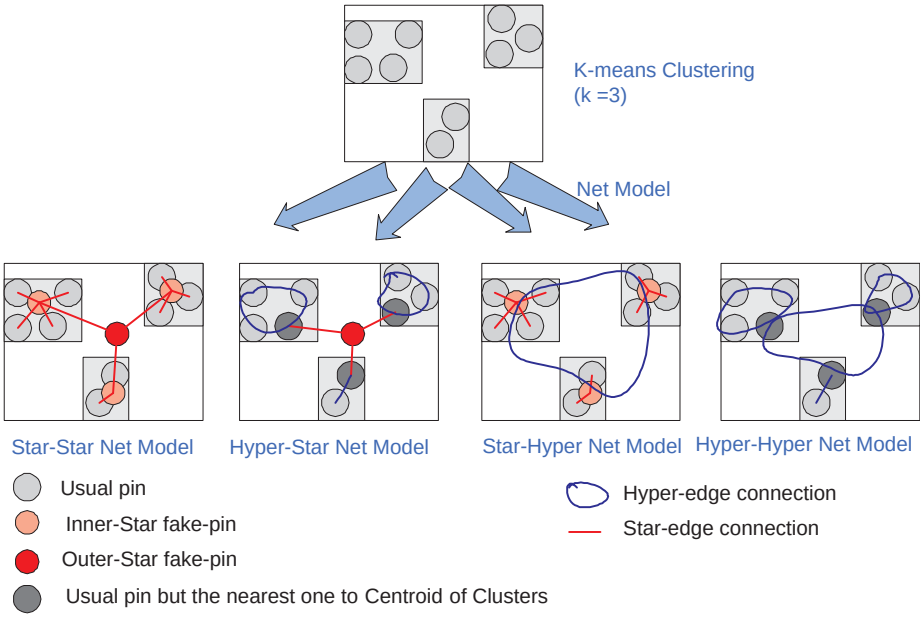


Figure 3.5: Construction of subnets on example of a net with 9 pins.

Four pictures on the bottom of Figure 3.5 show different combinations of interconnection the pins. One of the method is to induce a fake-pin for connecting either pins in the subnets or the subnets themselves. Such approach has the same disadvantage like the Star net model, it is necessary to introduce additional pin which should be considered by the placement algorithm later on in an appropriate way.

The method used in this work is to compute for every subset a pin which is the nearest to the centroid of the subsets. The last of four pictures presents the resulted interconnection denoted as *Hyper-Hyper Net Model*.

This method does not work if spatial information about the pins is not available. But the *MCN* net model is aware of placement of pins and, thus, the proposed approach is a reasonable for our case. Moreover, the nearest pins can be easily calculated if the information after the run of the clustering algorithm is accumulated.

Therefore, the construction of the subnets from the clustering solution of the subsets proceeds in two steps:

- The pins of each subset are connected with a hyperedge.
- For each cluster, the closest pin to the center of gravity C_T is selected. The set of closest pins are interconnected with another hyperedge.

Assuming that a clustering into k subsets has been performed, $k+1$ new subnets are created and neither additional fake-pin is introduced.

Conclusion

This Chapter has introduced basic ideas, algorithmic details and implementation of the *MCN* approach to the netlength modeling which are necessary for the next Chapter 4. Experimental framework and obtained results are presented there.

Further, the *MCN* model can be viewed from practical point of view, that is a "black box" tool for converting the input netlist to the output netlist in placement flow. In result, a proper placement step receives the modified netlist where nets are represented such a way that their estimation of the wirelength is improved.

Chapter 4

Experimental Results

This Chapter start with description of experimental framework used in the thesis. Section 4.1 explains how the *MCN* model can be introduced and tested in placement flow. Academical resources used in this work are also presented: several placement tools, three benchmark suits of circuits and a special tool for evaluation StWL.

Before experiments on optimization StWL in placement with the *MCN* model, this model is compared with HPWL in terms of accuracy in wirelength estimation in Section 4.2. The wirelength of *already produced placement* layout is estimated with both new *MCN* and traditional HPWL metrics. As a result important for further experiments, the *MCN* model empirically proved to approximate StWL significantly better than HPWL.

Based on results obtained in Seciton 4.2, the next two Sections 4.3 and 4.4 demonstrate how to *produce better placement* with shorter wirelength and delays by employing the *MCN* model in placement algorithms. The improvement is tracked for different placers to show generality of our net model, and for different benchmark suits to take into account some practical specificities.

Section 4.4 presents an extension of the basic experimental scheme. Since the *MCN* approach improves the netlength estimation by reflecting the positions of net pins, detailed placer can be run in a loop and adopt the *MCN* model to the produced placement iteratively. Such approach may be practical, because the runtime of detailed placer in placement flow is relatively small.

4.1 Experimental Framework

To test the application of our *MCN* model in existing placers, the net model is integrated in placement flow between two placement steps. This intermediate block is represented as a tool between global and detailed placers, which task is to modify the netlist based on the global placement layout and transfer the new netlist to detailed placement. Consequently, all positive benefits compared to the common placement flow are produced because of the netlist modifications.

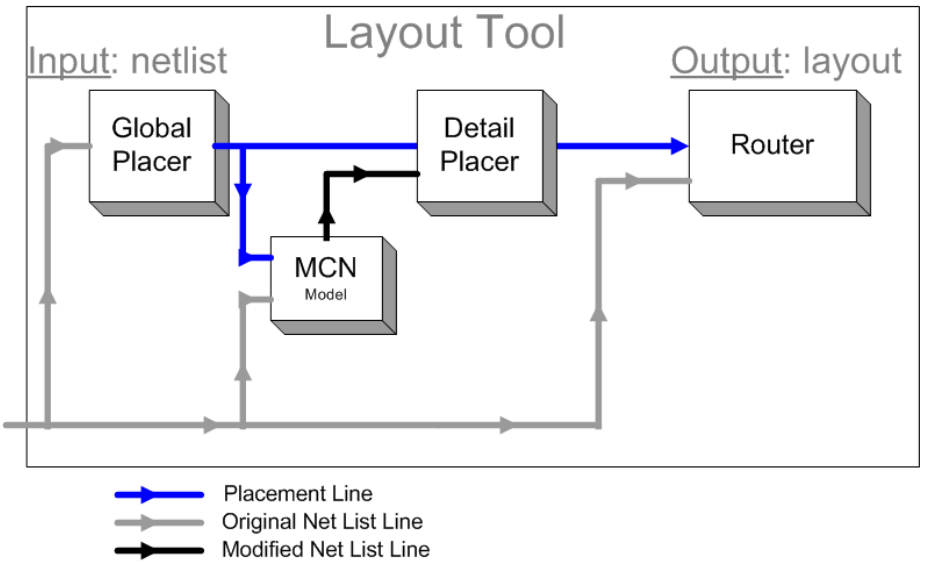


Figure 4.1: Experimental scheme of testing the *MCN* model in placement flow.

The *MCN* block in placement flow is viewed as a stand-alone tool which changes the netlist reflecting the placement of net pins. On the one hand, the *MCN* approach is accomplished on the high level of transition of the netlist which is a input/output quantity between placement steps. On the other hand, the integration of the *MCN* model into low level implementation of either global or detailed placer is not possible in this work, because academical placers are available as binary rather than open-source tools.

Therefore, the model can be tested regardless of a particular implementation of a placer. Considering this feature as an advantage, the net model has been tested for different placement tools according to a simple experimental scheme.

Experimental Scheme

The experimental scheme is depicted in Figure 4.1. The Figure presents physical design cycle with both placement and routing phases, where placement phase is divided into global and detailed and the additional block with the *MCN* model is introduced. For simplicity sake, the presented *Layout Tool* produces the output physical layout of a circuit based on the input netlist.

The effect of the *MCN model* block can be viewed through data path lines of the netlist. Contrary to *Global Placer* and *Router*, *Detailed Placer* operates with the modified netlist. Thus, the detailed placer is forced to employ the *MCN* model on wirelength estimation, because of the different input netlist.

Indeed, the detailed placement can be divided further into some internal blocks which correspond to iterative steps of the algorithm. Consequently, the *MCN* model may improve the wirelength estimation between these consistent steps by reflecting changes in placement from iteration to iteration. Such refinement of

the scheme can be performed in future work when designed our own placement tool. In this thesis, we rely on the fact that detailed placer performs minor changes in placement, as discussed in Section 2.1.2, and the layout received from global placer captures *relative positions* of net pins consistently enough with further placement changes.

Placement Tools

According the experimental scheme described above in Figure 4.1, global and detailed placer are represented as "black boxes", and the *MCN* model communicates with them through input/output lines of placement and netlist. However, some algorithmic and implementation basics of the examined placement tools are important to present for further explanation of the final results.

Three placement tools are used in this thesis, and all placers operates in two global and detailed phases. Moreover, each global placer differs in type of employed placement algorithm, which are described in details in Section 2.1.2.

mPL 6 [20] employs analytical placement algorithm combined with hierarchical multi-level framework. For examined circuits, this placer has been able to produce the best results.

FastPlace 3 [34] belongs to the class of quadratic placers and shows the best runtime along all placers. Additionally, its detailed placement algorithm has been demonstrated to be the most powerful along all detailed placers in [35].

Capo 10.5 [31] is a partition-based placer oriented to routability optimization. This tool utilize CPU time expensive methods on greedy swapping of cells for the detailed placement.

These three placers have been run to demonstrate the performance of the *MCN* model on the main experiments in Section 4.3.1, where obtained results expected to be consistent for all the placers, in order to prove the generality of the *MCN* model.

Placement Benchmarks

The current benchmarks used in physical design community for placement are typically presented in BookShelf format [36]. All circuits correspond to design style of the standard cells, as presented in Section 2.1.1. In this thesis, three benchmark suites are tested with the purpose to validate the *MCN* approach more absolutely. Figures of some circuits can be found in Appendix B, while the description is listed below.

ISPD05 circuits [5] are the most recent benchmarks with sizes ranging from 210 thousand to 2.1 million objects. The circuit elements are mostly standard cells, but there is sufficient quantity of macro blocks, that reflects

the presence of IP blocks in the current VLSI circuits. The goal of testing ISPD05 circuits is to present the efficiency of the *MCN* approach on the state-of-the-art benchmarks.

PEKU circuits [6] belong to the class of artificial benchmarks designed with a priori known optimal wirelength. These circuits were firstly presented in [6], in order to show incapability of placers to reach the bound with optimal wirelength. However, the property important for our experiments consists in the large number of multi-pin nets in the circuit netlist. The objective of experiments with PEKU benchmarks is to show superior results of the *MCN* model, because of orientation of the net model to high-degree nets.

ISCAS99 circuits [37] are quite old examples with the small number of cells, but only these benchmarks contain information for computing the wire delays. This feature of ISCA99 circuits is used to show that reduction in wirelength due to the *MCN* approach results in shorter wire delays.

The improvement in wirelength is reported for ISPD05 and PEKU circuits in Section 4.1, whereas the complete set of results in reduction of wirelengh and wire delays is given for ISCA99 benchmarks in Section 4.4.

Evaluator of Steiner tree Wirelength

The experimental objective is to produce placement layout with shorter wirelength expressed in StWL rather that HPWL. Contrary to HPWL, computation of the total StWL of nets can be computationally expensive due to NP cost of Steiner tree construction.

The software GeoSteiner [38] for exact computation RSMT in not practical for the examined circuits with up to 100 million of cells. The most recent heuristics FastSteiner [7] and FLUTE [11] approximates RSMT with some error in comparison with GeoSteiner, but in significantly less runtime. Typical error for instances of VLSI circuits concludes less than 1% that is admissible for our experiments. These two heuristics regarded to be equivalent [31], but the choice has been fixed upon FastSteiner in this thesis, because of its open source distribution.

In further Sections, StWL is referred to as the wirelength measured by FastSteiner package. Consequently, the next Section 4.2 compares the traditional HPWL net model and the new *MCN* model with respect to StWL.

4.2 Accuracy of Wirelength Estimation

When describing the *MCN* approach in Section 3.1, the improved wirelength estimation by the new net model was demonstrated on the small example of the net with 6 pins, depicted in Figure 3.1. However, placement tools operate with circuits containing millions of nets, and the accuracy in estimation of the total netlength of the circuit is of interest in practice. The goal of this Section is to

ISPD05 bench.	% nets > 3 pins	Wirelength error (%)			
		all nets		nets >3 pins	
		HPWL	*MCN*	HPWL	*MCN*
adaptec1	32	-3.73	0.45	-11.46	1.37
adaptec2	23	-2.90	0.19	-12.54	0.81
adaptec3	24	-2.62	0.25	-10.62	1.00
adaptec4	21	-2.19	0.22	-10.07	1.02
bigblue1	27	-3.36	0.32	-12.18	1.14
bigblue2	20	-2.38	0.19	-11.50	0.93
bigblue3	18	-1.94	0.12	-10.44	0.65
bigblue4	22	-2.55	0.17	-11.25	0.75
Norm.		**-2.71%**	**0.24%**	**-11.26%**	**0.96%**

Table 4.1: Average error in wirelength estimation of HPWL and *MCN* models in respect with FastSteiner [7] heuristic.

compare the accuracy of HPWL and *MCN* net models with regard to StWL on examples of the VLSI circuits.

To show that the *MCN* model outperforms the traditional HPWL metric, the experiments have been performed on the ISPD05 circuits placed by mPL 6, although other benchmark suits or placers may be selected. The results are summarized in Table 4.1 where the *relative error* in approximation of StWL, average along all the nets, is presented in percents.

Important factor affecting the accuracy of estimates is a percentage of multi-pin nets (more than 3 pins) in the circuit netlist. This number is reported in the second column of the Table. Comparing circuits with different influence of multi-pin nets, one can see that circuits with more number of high-degree nets in the netlist have worse estimation of StWL, for instance, `adaptec1` benchmark.

The average error is computed for all nets and for nets with more than three pins separately. The numbers correspondent to *MCN* and HPWL models are denoted in the Table as `HPWL` and *`MCN`*, respectively. The HPWL underestimates the wirelength that is shown with negative numbers. Contrary to HPWL, the *MCN* approach increases the accuracy consistently for all the circuits and improves the estimation from -2.71% to 0.24% for all nets in respect with HPWL.

For both net models the lost of precision comes from multi-pin nets, as presented in the last two columns of the Table. However, the percentage of these high-degree nets in the circuit netlist account for $20 - 30\%$, in average 25%. This factor of 25% or $1/4$ is confirmed by normalized numbers in the last row of the Table. Small amount of multi-pin nets produces a significant increment of inaccuracy of -11.26% for HPWL measure, but the error computed for all nets is only -2.71% that is about 4 times less. The same tendency is observed for numbers correspondent to the *MCN* net model.

Such property of the netlist for VLSI circuit instances (most of the nets have 2-3 pins) allows HPWL to be efficient measure in placement regardless the crucial underestimation for multi-pin nets. However, the results presented in the Table 4.1 show that the *MCN* model is able to estimate the same netlength with considerably smaller error of 0.24% in average, in comparison with the HPWL error of -2.71%.

The results obtained in this Section justify the *MCN* heuristic for the opti-

mization of the StWL in placement. The rest Sections of this Chapters present experiments where placers produce layouts with shorter wirelength and wire delays by replacing the HPWL by the StWL cost (expressed by the *MCN* model).

4.3 Improvement in Wirelength

This Section reports about the main experimental results achieved in this work. The tangible improvement in the wirelength with no or little CPU time overhead according the scheme presented in Figure 4.1 validates the *MCN* approach proposed in the thesis. The obtained results are collected in Tables 4.2 and 4.3 organized in the same manner. Fields of the tables are introduced before presenting the results.

Notations in Tables

For each circuit the number of cells and average net degree are shown in the columns called as `# nodes` and `Avr. ND` respectively. This statistic reflects the size of the circuit and the netlist structure in terms of presence of multi-pin nets. Particularly, the percentage of high-degree nets can serve as a predictor of possible StWL improvement.

When stating the improvement in wirelength, wire delays or CPU time, two placement flows are compared. First is the traditional flow with two global and detailed placement steps, referenced in the tables as `GP+DP`. Second is the new flow with our intermediate step (exactly depicted in in Figure 4.1), referenced in the tables as `MCN`.

The wirelength reported in the tables is evaluated in both HPWL and StWL format. For each circuit the wirelength is given in absolute numbers of dimensionless units of BookShelf format [36], in order one can compare the numbers obtained in our experiments with numbers of others placers and StWL optimization strategies. The correspondent columns are referred to as `HPWL` and `StWL` respectively.

The column `CPU Ratio` presents the overhead in runtime of the two placement flows described above. The numbers less than 1.00 mean that the *MCN* flow requires less time than the traditional flow. The total runtime includes the CPU time spend by global placer, *MCN* tool on construction the new netlist and detailed placer. Since the runtime of global placer is equal for both placement flows, the difference arise due to *MCN* tool and detailed placer run. The first component is additive and always leads to more time for the new placement flow, whereas the second depends on implementation of detailed placement algorithm.

In practice, the contribution on the total CPU time of the *MCN* netlist construction observed to be *meaningless*. This experimental artifact validates the reasoning for the design of the clustering algorithm in Section 3.3, where the k-means algorithm is iterated and the solution for the *MCN* model is selected according to the local search algorithm. In other words, the *MCN* computation

ISPD05 bench.	Statistics		mPL 6				
	#nodes ($\times 10^5$)	Avr. ND	HPWL ($\times 10^8$)		StWL ($\times 10^8$)		CPU Ratio
			GP+DP	*MCN*	GP+DP	*MCN*	
adaptec1	2.1	4.3	0.78	0.79	0.88	0.86	1.00
adaptec2	2.6	4.0	0.92	0.93	1.07	1.03	1.00
adaptec3	4.5	4.0	2.14	2.16	2.40	2.32	1.00
adaptec4	5.0	3.7	1.94	1.95	2.13	2.06	0.99
bigblue1	2.8	4.0	0.97	0.99	1.11	1.07	1.00
bigblue2	5.6	3.7	1.52	1.53	1.75	1.70	0.96
bigblue3	11.0	3.4	3.44	3.48	4.07	3.95	0.98
bigblue4	22.8	4.0	8.30	8.37	9.41	9.15	0.95
Norm.			**1.000**	**1.013**	**1.000**	**0.970**	**0.98**

ISPD05 bench.	FastPlace 3				
	HPWL ($\times 10^8$)		StWL ($\times 10^8$)		CPU Ratio
	GP+DP	*MCN*	GP+DP	*MCN*	
adaptec1	0.80	0.83	0.90	0.89	1.01
adaptec2	0.94	0.97	1.10	1.08	0.96
adaptec3	2.14	2.20	2.39	2.37	0.93
adaptec4	2.01	2.06	2.19	2.18	0.95
bigblue1	0.98	1.02	1.12	1.11	0.97
bigblue2	1.56	1.59	1.78	1.76	0.99
bigblue3	3.77	3.78	4.36	4.28	0.94
bigblue4	8.60	8.62	9.58	9.41	0.94
Norm.	**1.000**	**1.024**	**1.000**	**0.988**	**0.98**

ISPD05 bench.	Capo 10.5				
	HPWL ($\times 10^8$)		StWL ($\times 10^8$)		CPU Ratio
	GP+DP	*MCN*	GP+DP	*MCN*	
adaptec1	0.88	0.89	0.98	0.97	1.13
adaptec2	0.99	1.00	1.15	1.13	1.13
adaptec3	2.44	2.50	2.62	2.58	1.15
adaptec4	2.16	2.19	2.36	2.34	1.17
bigblue1	1.08	1.09	1.22	1.20	1.15
bigblue2	1.62	1.64	1.85	1.83	1.14
bigblue3	4.30	4.35	4.99	4.91	1.14
bigblue4	9.74	9.80	10.8	10.6	1.15
Norm.	**1.000**	**1.011**	**1.000**	**0.985**	**1.14**

Table 4.2: *MCN* approach on ISPD05 circuits.

is allowed to spend more CPU time than a single run of the k-means algorithm, nevertheless the netlist information of millions of nets is processed.

Every table of results in this Section contain the last row called `Norm.` which corresponds to the normalized sum measure. The average numbers in two certain columns of the `GP+DP` and `MCN` flows are calculated for all the circuits. The normalized value for the `MCN` column is computed by devision the second number to the first, whereas the value of 1.00 is given to the `GP+DP` column. Values less than 1.00 in `MCN` columns show improvement either in wirelength or CPU time of the *MCN* approach in comparison with the traditional flow.

The next two Subsections 4.3.1 and 4.3.2 present the tables of results for academical ISPD05 benchmark suite and artificial PEKU benchmark suite respectively.

4.3.1 Experiments on ISPD05 circuits

The table 4.2 presents the results obtained on the most recent ISPD05 benchmarks. The ISPD05 circuit includes circuits with number of cells up to 2 million of cells as depicted in the second column of the table. Additionally, the circuits

PEKU bench.	Statistics		mPL 6				
	#nodes ($\times 10^5$)	Avr. ND	HPWL ($\times 10^6$)		StWL ($\times 10^6$)		CPU Ratio
			GP+DP	*MCN*	GP+DP	*MCN*	
dp01	1.3	111.7	0.91	0.92	2.96	2.62	1.00
dp05	2.9	167.5	1.95	1.96	7.41	6.59	1.01
dp10	7.0	261.7	5.62	5.52	24.77	23.76	1.04
dp15	16.3	401.5	11.99	11.77	69.92	65.23	0.98
dp18	21.2	458.3	16.19	14.43	99.48	86.71	1.00
Norm.			**1.000**	**0.974**	**1.000**	**0.908**	**1.01**

Table 4.3: *MCN* approach on PEKU circuits.

have the considerable number of multi-pin nets as confirms the average net degree in the third column.

The main goal of these experiments is to demonstrate the performance of the *MCN* approach on wirelength reduction for the state-of-the-art circuits in the current physical design. Three placers mPL6 [20], FastPlace 3 [34] and Capo 10.5 [31][1] have been run and finally shown the similar results.

The total HPWL reasonably increases for all placers and circuits as shown in the last row `Norm.`, because the *MCN* model targets to simulate StWL rather than HPWL. Consequently, each net is split into subnets according to the *MCN* algorithm, that force pins to be compactly arranged inside subnets, but not in the HPWL bounding box.

The main achievement is reduction in StWL which expressed in normalized sum by 3.0%, 1.2% and 1.5% for mPL6, FastPlace and Capo respectively. These results corroborate the *MCN* approach for wirelength optimization in the current placement framework, nevertheless the percentage of high-degree nets of the circuits is small and detailed placement algorithm (which process the *MCN* netlist) is not so powerful as global placement.

In terms of CPU time, the runtime concerned with the *MCN* approach is better by 2% in average for both mPL and FastPlace. But there is significant overhead of 14% for Capo. As discussed above in the previous Subsection, the detailed placement algorithm of Capo assumed to be more sensitive to the increment on the number of nets due to net *MCN* netlist transformation rather than the reduction in the average net degree. However, the runtime of constructing the *MCN* netlist is less than 1% of overall placer CPU time for Capo as well as for mPL or FastPlace.

In order to improve results of reduction in wirelength, the performance of the model is examined on the PEKU circuits with the large number of multi-pin nets in the next Section 4.3.2. These artificial benchmarks contain only multi-pin nets, whereas the ISPD05 circuits tend to have low-pin nets.

4.3.2 Experiments on PEKU circuits

The table 4.3 reports the results on experiments with PEKU circuits. These benchmarks are artificial and have been designed like a grid of multi-pin nets crossing each other. The number of cell belongs to a wide range with the maximum of 2 million, as depicted in the second column. Big numbers in the second column show that PEKU circuits have very high-degree nets in the netlist, and, thus, the *MCN* approach expected to show superior reduction in StWL.

The mPL6 [20] placement tool is selected as global and detailed placer, because it can work on PEKU benchmarks. FastPlace3 [34] fails on some circuits, and Capo requires to much computational time.

The *MCN* strategy leads to superior results in StWL reduction which is improved by 9.2% at the expense of 1% CPU time increase, as depited in the last row of normalized sum. Additionally, the improvement in HPWL is observed for some circuits, that underline more drawbacks of the HPWL measure applied to multi-pin nets rather than positive correlation between StWL and HPWL.

However, the results presented in this Subsection can not be considered as a strong argument for the *MCN* model to apply in real placement framework, since typical academical and industrial circuits contain mainly low-pin nets. To sum up experiments on ISPD05 and PEKU performed in this Section, the main results are related with tangible reduction of StWL on ISPD05 circuits, whereas the numbers obtained for PEKU benchmarks are intended for the demonstration purpose.

4.4 Improvement in Wire Delays

This Section extends the experiments carried out in the previous Section 4.3. The experimental objective is to examined the influence of wirelength reduction on wire delays. We can figure out the correlation between the wirelength cost function applied in physcial design and the delay cost function which directly defines the speed of the circuit. Thus, the *MCN* model will be validated for optimization the real performance of the circuit.

These experiments require a step of *technology mapping* for transforming the logic of the circuit to the layout representation in the BookShelf format. The $0.13\mu m$ *vxlib ALLIANCE* library [39] has been used for technology mapping. The technological parameters have been scaled down for the different technologies (65nm and 32nm), using the *Predictive Technology Model* [40]. For instance, the wire capacitance and resistance for 65nm are $2.71\Omega/\mu m$ and $0.19fF/\mu m$, respectively, that approximately correspond to M2/M3 metal layers of the 65nm technology described in [41].

[1]ROOSTER feature is disabled in Capo, because ROOSTER is aimed strictly for routability, and comparison in the wirelength is not fair(according to the personal reference to the authors).

ISCAS99 bench.	Statistics		Wirelength		
	#nodes ($\times 10^5$)	Avr. ND	StWL ($\times 10^6$) GP+DP	StWL ($\times 10^6$) *MCN*	CPU Ratio
b14_1	4.6	3.0	2.85	2.72	0.96
b15_1	7.3	3.2	4.81	4.69	0.98
b17_1	22.5	3.2	14.61	14.30	0.97
b20_1	8.9	3.1	5.77	5.65	0.98
b21_1	9.1	3.1	5.85	5.69	0.96
b22_1	13.4	3.1	7.41	7.25	0.97
s13207	2.7	2.8	1.49	1.38	0.99
s15850	10.7	2.9	1.95	1.88	0.97
s38584	10.0	2.9	6.00	5.94	0.99
Norm.			**1.000**	**0.969**	**0.98**

ISCAS99 bench.	65 nm				32 nm			
	WNS ($\times 10^3$ ps)		TNS ($\times 10^6$ ps)		WNS ($\times 10^3$) ps		TNS ($\times 10^6$ ps)	
	GP+DP	*MCN*	GP+DP	*MCN*	GP+DP	*MCN*	GP+DP	*MCN*
b14_1	5.71	5.39	1.12	1.08	7.62	7.01	1.31	1.26
b15_1	6.85	6.69	1.93	1.89	8.33	8.03	2.36	2.27
b17_1	7.18	6.82	6.37	6.22	9.32	8.39	7.74	7.58
b20_1	8.04	8.05	2.56	2.51	10.27	9.12	2.99	2.87
b21_1	8.21	8.08	2.67	2.66	9.66	9.38	3.14	3.10
b22_1	9.53	9.60	4.35	4.27	10.59	10.61	5.03	4.94
s13207	2.88	2.74	0.44	0.43	3.01	0.95	0.50	0.49
s15850	3.94	3.85	1.20	1.22	9.58	9.39	5.50	4.93
s38584	0.99	5.01	4.17	3.63	13.96	13.57	9.24	7.36
Norm.	**1.000**	**0.974**	**1.000**	**0.972**	**1.000**	**0.950**	**1.000**	**0.944**

Table 4.4: *MCN* approach on ISCAS99 circuits.

The initial circuits have been obtained by using the tree-height reduction technique `speed_up` [42] and the tree-mapping algorithm in the SIS tool [43]. A square layout with 25% whitespace has been created, with the terminals uniformly distributed around the bounding box.

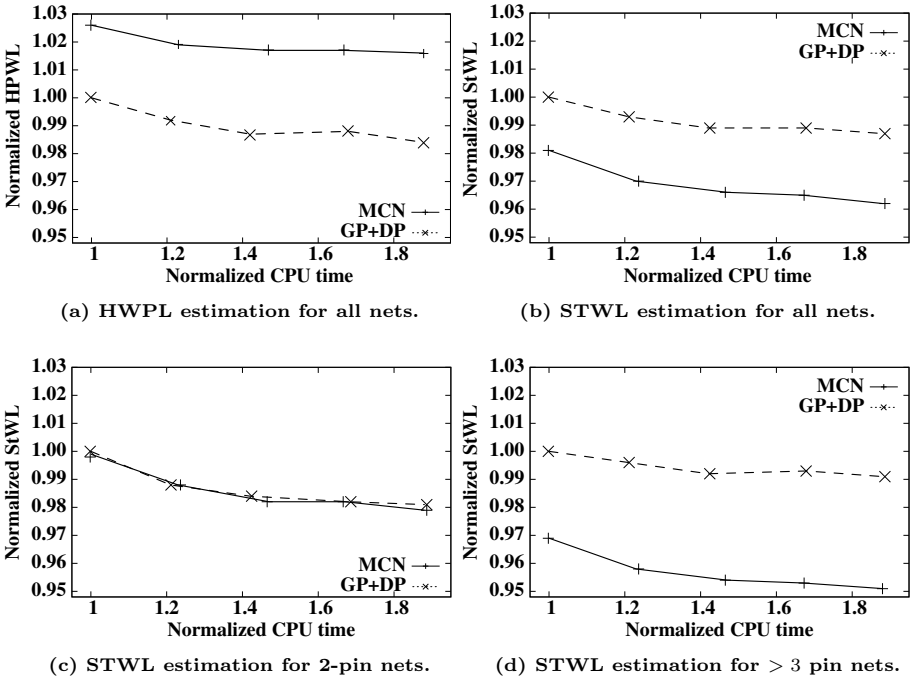
4.4.1 Experiments on ISCA99 circuits

The ISCAS99 benchmarks allow to track improvement not only in StWL but also in delays. We selected the largest circuits and used FastPlace [34] as a placement tool. Table 4.4 summarizes the results in the same manner as the previous tables, but two additional values of the worst negative slack `WNS` and the total negative slack `TNS` are also reported . These numbers are estimated with the FastSteiner [7] package to build the pass of wires.

The same tendency is observed in wirelength and runtime improvement with respect to the previous experiments on ISPD05 circuits. The StWL is reduced by 3.1%, as reported in the last row of normalized sum.

The delays are presented in the Table for 65nm and 32nm technologies separately. Although the *MCN* model is not a delay-oriented approach, the improvement in wirelength is also reflected in delays. The improvement in delays is reported in two `WNS` and `TNS` columns of the Table.

As an important result of this Section, the reduction of delay in future semiconductor technologies (from 2.8% of improvement in 65nm to 5.6% in 32nm) confirms the increasing relevance of interconnect optimization due to the dominant role of wire delays in the physical design cycle.



(a) HWPL estimation for all nets.

(b) STWL estimation for all nets.

(c) STWL estimation for 2-pin nets.

(d) STWL estimation for > 3 pin nets.

Figure 4.2: Iterative detailed placement on `adaptec1` circuit.

4.5 Iterative Detailed Placer

This Section presents one possible improvement of the scheme described in Fig. 4.1consists on the idea of feedback line (not shown in the scheme). We can produce the *MCN* netlist and run detailed placer iteratively. The input placement for building the netlist is the detailed placement produced in the previous iteration. The proposed iterative approach is reasonable, because detailed placement contributes a small portion in overall CPU time compared to the global placement.

Figure 4.2 presents the results for *adaptec1* circuit during 5 runs of the detailed placer mPL6 [20]. We compare the iterative *MCN* approach (denoted as *MCN*) with an iterative scheme of the traditional flow (denoted as GP+DP). Figure 4.2(a) shows the expected HPWL reduction due to the iterative application of detailed placer. However, the *MCN* approach gives superior results in StWL reduction because of employing the *MCN* model as plotted in Fig. 4.2(b).

Figures 4.2(c,d) present the main feature of the *MCN* approach. The StWL is measured separately for low degree and multi-pin nets. The wirelength of low degree nets for both *MCN* and GP+DP placement flows is being reduced at the same ratio from iteration to iteration as shown on Fig. 4.2(c). However, the *MCN* approach can improve the wirelength for high degree nets significantly, plotted in Fig. 4.2(d).

Conclusion

The conclusions concerned with the experiments performed in this Chapter are moved to the final Chapter 5, where the direction line of future work is also discussed.

Chapter 5

Conclusions

5.1 Results

The experimental results presented in the previous Chapter 4 validate the application of the proposed *MCN* model for reduction the StWL cost. The placement framework includes the most recent ISPD05 benchmarks and placement tools. The achieved improvement of $2-3\%$ in wirelength is a typycal number for demonstration of effectiveness of some optimization technique in placement. The experiments on ISCA99 circuits complete the results for one of the most important cost function in physical design - wire delays.

Two series of additional experiments have been run to show superior improvements in StWL for some specific experimental needs. The reduction by 9% has been obtained for the artificial PEKU circuits with very high-degree nets, that corroborate the *MCN* model for handling multi-pin nets. Another type of experiments on iterating the detailed placer have shown some perspectives on using the *MCN* approach in placement algorithm, which are more powerful than detailed placer.

The work presented in this thesis has been accepted as a regular paper [44] to the IFIP International Conference on Very Large Scale Integration and will be published in October 2008.

Contribution

The new clustering approach for better wirelength modeling in placement has been proposed. We experimentally proved that our *MCN* model approximates the Steiner tree wirelength more accurately than the traditional HPWL model. Circuits with shorter wirelength and delays have been produced on the explored placement flows. The main contribution of our work is formulated by the following statements.

Clustering approach. A clustering technique for the netlength modeling problem is introduced, and an efficient implementation with linear complex-

ity on the number of pins is presented. When a net is split into several lower degree subnets, the total HPWL of the subnets approximates RSMT length significantly better than HPWL of the original net.

Practical StWL minimization. By transforming the netlist between global and detailed placement stages accordingly to the *MCN* model, the total StWL of all nets is improved with no or little penalty in runtime. The proposed approach neither depends on any placement algorithm or RSMT heuristic. In practice, it can be applied between any consecutive placement steps.

5.2 Future Work

The *MCN* approach has been tested in placement framework and proceeded in detailed placement phase. As future work, the net model can be integrated in global placer to reach stronger results in wirelength optimization. Such approach seems to be promising, because the *MCN* model is HPWL-based and global placement algorithms include a variety of well-developed techniques on minimization of the HPWL cost.

The goal of optimization StWL can not be considered as decisive in physical design cycle, since the final objective reflecting the performance of the produced layout is routed wirelength. Thus, the approach in StWL minimization designed in this thesis has to be combined with some techniques on estimation the routing cost in placement.

In terms of implementation of the *MCN* clustering algorithm, it can be based on ISODATA algorithm or implemented with precomputing the number of clusters according to density distribution of pins. Additionally, the cost function of the clustering can include more parameters relevant for the quality of clustering.

Finally, another line of investigation is to design a delay-aware clustering model, in order to group critical pins to the same subnet. This new model applied in placement expected to reduce delays on critical paths considerably, since most timing violations come from critical long wires, which typically are multi-pin nets.

Appendix A

VLSI design cycle

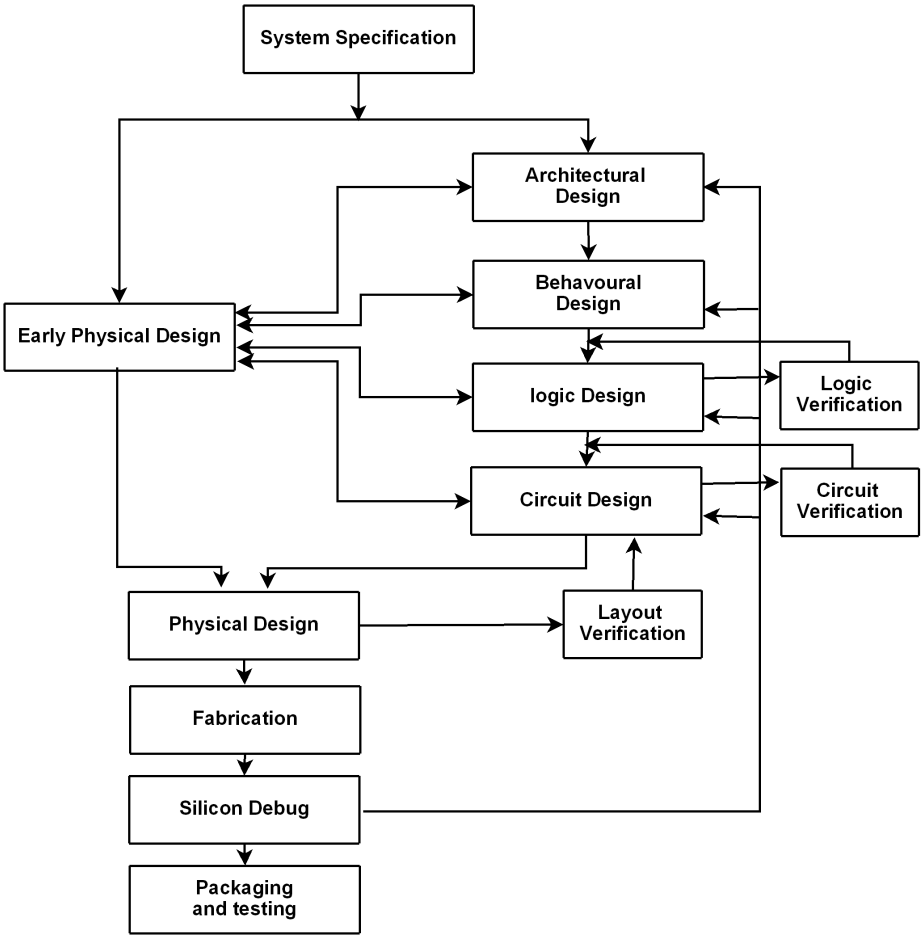


Figure A.1: The scheme of VLSI design cycle [4].

Appendix B

Figures of some benchmarks

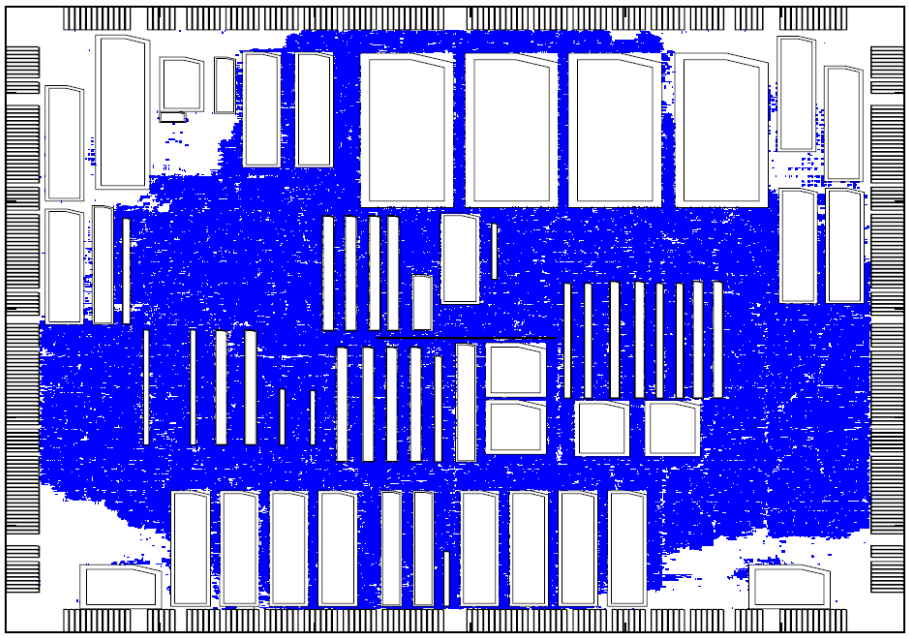


Figure B.1: The layout of a circuit from ISPD05 benchmark suite with standard cells (depicted with blue) and macro blocks [5]. `adaptec1` circuit with 211447 cells and 221142 nets.

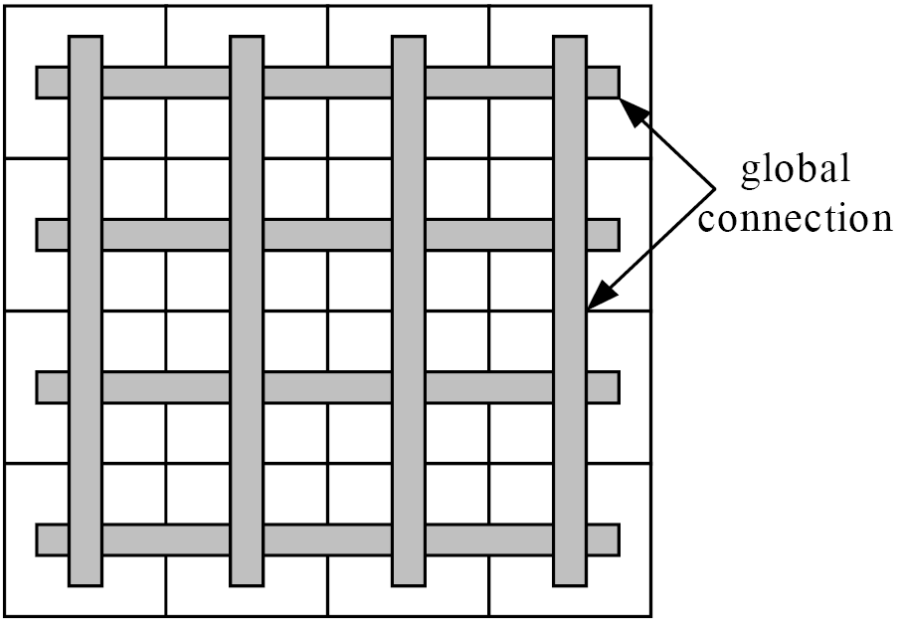


Figure B.2: A grid approach of artificial PEKU benchmarks to simulate nets with many pins [6].

Bibliography

[1] Sachin Sapatnekar. Class: Physical design for vlsi circuits. `http://www.lsi.upc.edu/sachin/`.

[2] Natarajan Viswanathan and Chris Chong-Nuen Chu. FastPlace: efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model. In *ISPD '04: Proceedings of the 2004 international symposium on Physical design*, pages 26–33, New York, NY, USA, 2004. ACM.

[3] Peter Spindler and Frank M. Johannes. Fast and robust quadratic placement combined with an exact linear net model. In *ICCAD '06: Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design*, pages 179–186, New York, NY, USA, 2006. ACM Press.

[4] Sherwani N. *Algorithms for VLSI physical design automation*. 1995.

[5] Gi-Joon Nam, Charles J. Alpert, Paul Villarrubia, Bruce Winter, and Mehmet Yildiz. The ISPD2005 placement contest and benchmark suite. In *ISPD '05: Proceedings of the 2005 international symposium on Physical design*, pages 216–220, New York, NY, USA, 2005. ACM.

[6] Jason Cong, Tim Kong, Joseph R. Shinnerl, Min Xie, and Xin Yuan. Large-scale circuit placement: Gap and promise. In *ICCAD '03: Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*, page 883, Washington, DC, USA, 2003. IEEE Computer Society.

[7] Andrew B. Kahng, Ion I. Măndoiu, and Alexander Z. Zelikovsky. Highly scalable algorithms for rectilinear and octilinear steiner trees. In *ASPDAC: Proceedings of the 2003 conference on Asia South Pacific design automation*, pages 827–833, New York, NY, USA, 2003. ACM.

[8] Intel Corp. 60 years of the transistor: 1947 - 2007. `http://www.intel.com/technology/timeline.pdf`.

[9] Andrew B. Kahng. A roadmap and vision for physical design. In *ISPD '02: Proceedings of the 2002 international symposium on Physical design*, pages 112–117, New York, NY, USA, 2002. ACM.

[10] M. R. Garey and D. S. Johnson. The rectilinear Steiner problem is NP-Complete. In *SIAM Journal of Applied Mathematics*, pages 826–834, 1977.

[11] C. Chu. FLUTE: fast lookup table based wirelength estimation technique. In *ICCAD '04: Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, pages 696–701, Washington, DC, USA, 2004. IEEE Computer Society.

[12] Andrew B. Kahng and Sherief Reda. A tale of two nets: studies of wirelength progression in physical design. In *SLIP '06: Proceedings of the 2006 international workshop on System-level interconnect prediction*, pages 17–24, New York, NY, USA, 2006. ACM.

[13] Sartaj Sahni and Teofilo Gonzalez. P-Complete approximation problems. *J. ACM*, 23(3):555–565, 1976.

[14] M. Queyranne. Performance ratio of polynomial heuristics for triangle inequality quadratic assignment problem. *Operations Research Letters*, 4:231–342, 1986.

[15] Chen Li, Min Xie, Cheng-Kok Koh, J. Cong, and P. H. Madden. Routability-driven placement and white space allocation. In *ICCAD '04: Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, pages 394–401, Washington, DC, USA, 2004. IEEE Computer Society.

[16] Andrew B. Kahng, Sherief Reda, and Qinke Wang. Aplace: a general analytic placement framework. In *ISPD '05: Proceedings of the 2005 international symposium on Physical design*, pages 233–235, New York, NY, USA, 2005. ACM.

[17] Peter Spindler and Frank M. Johannes. Fast and accurate routing demand estimation for efficient routability-driven placement. In *DATE '07: Proceedings of the conference on Design, automation and test in Europe*, pages 1226–1231, San Jose, CA, USA, 2007. EDA Consortium.

[18] Taraneh Taghavi, Foad Dabiri, Ani Nahapetian, and Majid Sarrafzadeh. Tutorial on congestion prediction. In *SLIP '07: Proceedings of the 2007 international workshop on System level interconnect prediction*, pages 15–24, New York, NY, USA, 2007. ACM Press.

[19] Wern-Jieh Sun and Carl Sechen. Efficient and effective placement for very large circuits. In *ICCAD '93: Proceedings of the 1993 IEEE/ACM international conference on Computer-aided design*, pages 170–177, Los Alamitos, CA, USA, 1993. IEEE Computer Society Press.

[20] T. F. Chan, J. Cong, J. R. Shinnerl, K. Sze, and M. Xie. mPL6: enhanced multilevel mixed-size placement. In *ISPD '06: Proceedings of the 2006 international symposium on Physical design*, pages 212–214, New York, NY, USA, 2006. ACM.

[21] Laspack package.

[22] F. M. Johannes J. M. Kleinhans, G. Sigl and K. J. Antreich. Gordian: Vlsi placement by quadratic programming and slicing optimization. In *IEEE Transactions on Computer-Aided Design of Circuits and Systems*, pages 356–365, 1991.

[23] Georg Sigl, Konrad Doll, and Frank M. Johannes. Analytical placement: A linear or a quadratic objective function? In *DAC '91: Proceedings of the 28th conference on ACM/IEEE design automation*, pages 427–432, New York, NY, USA, 1991. ACM.

[24] Andrew E. Caldwell, Andrew B. Kahng, and Igor L. Markov. Can recursive bisection alone produce routable placements? In *DAC '00: Proceedings of the 37th conference on Design automation*, pages 477–482, New York, NY, USA, 2000. ACM.

[25] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *DAC '82: Proceedings of the 19th conference on Design automation*, pages 175–181, Piscataway, NJ, USA, 1982. IEEE Press.

[26] Andrew E. Caldwell, Andrew B. Kahng, and Igor L. Markov. Design and implementation of move-based heuristics for vlsi hypergraph partitioning. *J. Exp. Algorithmics*, 5:5, 2000.

[27] Ulrich Brenner and André Rohe. An effective congestion driven placement framework. In *ISPD '02: Proceedings of the 2002 international symposium on Physical design*, pages 6–11, New York, NY, USA, 2002. ACM.

[28] A. E. Dunlop and B. W. Kernighan. A procedure for placement of standard-cell vlsi circuits. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 92–98, 1985.

[29] Y. W. Chang T. C. Chen and S. C. Lin. Imf: Interconnect-driven multilevel floorplanning for large-scale building-module designs. In *ICCAD*, pages 159–164, 2005.

[30] N. Selvakkumaran and G. Karypis. Technical report, university of minnesota. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2004.

[31] Jarrod A. Roy, James F. Lu, and Igor L. Markov. Seeing the forest and the trees: Steiner wirelength optimization in placement. In *ISPD '06: Proceedings of the 2006 international symposium on Physical design*, pages 78–85, New York, NY, USA, 2006. ACM Press.

[32] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symp. on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, 1967. University of California Press.

[33] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):881–892, 2002.

[34] N. Viswanathan, Min Pan, and C. Chu. FastPlace 3.0: A fast multilevel quadratic placement algorithm with placement congestion controlr. In *ISPD '04: Proceedings of the 2004 international symposium on Physical design*, pages 135–140, 2007.

[35] Min Pan, N. Viswanathan, and C. Chu. An efficient and effective detailed placement algorithm. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 48–55, Washington, DC, USA, 2005. IEEE Computer Society.

[36] GSRC/BookShelf format.
vlsicad.ucsd.edu/GSRC/bookshelf/Slots/Placement/plFormats.html.

[37] ISCAS99 circuits.
http://www.intusoft.com/benchmarks.htm.

[38] GeoSteiner software for computing Steiner trees.
http://www.diku.dk/geosteiner/.

[39] Alliance library.
www.vlsitechnology.org/html/vx_description.html.

[40] Interconnection calculator.
www.eas.asu.edu/~ptm/cgi-bin/interconnect/local.cgi.

[41] P. Bai, C. Auth, and et al. *A 65nm Logic Technology Featuring 35nm Gate Lengths, Enhanced Channel Strain, 8 Cu Interconnect Layers, Low-k ILD and 0.57μm^2 SRAM Cell.* Intel Developer Forum, August 2005.

[42] K.J. Singh, A.R. Wang, R.K. Brayton, and A. Sangiovanni-Vincentelli. Timing optimization of combinational logic. In *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pages 282–285, November 1988.

[43] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli. SIS: A system for sequential circuit synthesis. Technical report, U.C. Berkeley, May 1992.

[44] Andrey Ziyatdinov, David Baneres, and Jordi Cortadella. Multi-clustering net model for placement algorithms. In *IFIP International Conference on Very Large Scale Integration*, to be published in October 2008.